GENERATING MARKOV RANDOM FIELD IMAGE ANALYSIS SYSTEMS FROM EXAMPLES

by

Davin Milun

A dissertation submitted to the
Faculty of the Graduate School of
State University of New York at Buffalo
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

May 1995

Abstract

This thesis develops image interpretation systems from training sets of images. The system learns Markov random field parameters by sampling from this training set. Specialized MRFs can model both the image and the noise processes. The method has been tested in the domains of binary image reconstruction, edge detector enhancement and edge detection.

Capturing the noise process within the MRF parameters is achieved by sampling neighborhoods in the original images together with the corresponding neighborhoods in the noisy images, and so creating a probability density function over pairs of neighborhoods across both images. This technique is that of Double Neighborhood MRF (DNMRF). This technique can even capture the noise model in situations where the noise is correlated.

The task that the DNMRF method performs, and the domain in which it operates, can be changed by simply providing an appropriate set of training images. The collection of domains in which this system will work are those problems where the task to be performed is localized in nature, has a binary desired result, and has an available set of training images.

The required size of the training set is studied. For standard MRFs, only 16 images of size 64×64 are required to be sampled. The performance of DNMRFs continue to improve even as the sample size increases about 256 images.

The method uses optimization to interpret images. The optimizers used are the ICM gradient descent method, and simulated annealing. The sparseness of sampled MRF distributions is ameliorated by a technique developed by Hancock and Kittler.

Of note are the domains of edge detection and edge detector enhancement of JPEG compressed images. In the case of edge detection the DNMRF method, when operating on binary images, performed as well as, or even better than, other well known edge detectors, such as Sobel and Canny. In the case of edge enhancement of JPEG compressed images, the method greatly improved the output of the edge detector.

Acknowledgments

I would like to thank my committee, Dr. David B. Sher, Dr. Deborah K. Walters and Dr. Sargur N. Srihari. Special thanks go to Dr. Sher, for his advice, direction and insight in the formation of my thesis topic, and who continued to advise me even after he left Buffalo. Also to Dr. Walters, who, despite a very busy schedule, joined my committee and became a co-chair in the past six months—without her advice, input and motivation, I would not have completed my thesis.

Thanks also go to Dr. Eugene Kleinberg, for the comments he contributed early on in the thesis process. And thanks too, to my outside reader, Dr. Raj Acharya.

Many thanks go to Helene Kershner and Kulbir Arora, for their help, words of encouragement, and goading to actually achieve my goals. Also to Ellie Benzel, the graduate secretary, in her role as deadline keeper, and caring supporter. Thanks too to the rest of the computer science department secretaries and faculty for their concern and support.

Thanks to Harry Delano, the most considerate and flexible of bosses, for the support and time that he has given me to allow me to get my thesis done. And to Ken Smith, my office mate, co-worker and friend, for his caring and nagging, and for picking up the slack at work while I was otherwise occupied these past few weeks.

Thanks also go to all my friends, the csgeeks in particular, for their support, taunting and nagging.

And to my parents, for their love, support and proofreading. One could not ask for nicer, better and more caring parents.

And last, but definitely not least, thanks and love go to my fiancee, Elyse Podnos, for all her love, care and support, especially during the last few weeks before my defense.

Contents

Abstract **ii**

Acknowledgments **iii**

1 Introduction **1**

1.1 Overview 1

1.2 Markov Random Fields 2

1.3 Double Neighborhood MRFs (DNMRFs) 4

1.4 Overview of the thesis 6

2 Previous work **7**

2.1 Edge Relaxation and Binary Image Reconstruction 7

2.1.1 Markov Random Field Models 8

2.1.2 Probabilistic Relaxation 12

2.2 Edge Detection 13

3 Improving sampled probability density functions **15**

3.1 Introduction 15

3.2 Theoretical Implementation 16

3.3 Hancock and Kittler's technique 19

3.4 Results 21

3.5 Conclusions 27

4 Binary Image Enhancement — DNMRFs **28**

4.1 Introduction 28

4.2 Theoretical Implementation 29

4.3 Results 31

4.4 Conclusions 34

5 How much training data is required **35**

5.1 Introduction 35

5.2 Single field MRFs 35

5.3 Double neighborhood MRFs 38

5.4 Summary 39

6 Edge detection enhancement — DNMRFs **42**

6.1 Introduction 42

6.2 Measures of goodness 43

6.3 Results on images with Gaussian noise 43

6.4 Compression noise 48

6.5 Results with compression noise 48

6.6 Results on real images 50

7 Taking Advantage of Rotational Invariance **53**

7.1 Introduction 53

7.2 Rotational Invariance 53

7.3 Rotationally Invariant Domains 56

7.4 Improving Distributions through Rotational Invariance 58

7.5 Reducing the Size of Distributions 58

8 Edge detection **60**

8.1 Introduction 60

8.2 Training 60

8.3 Measures of correctness 61

8.4 Results 62

8.5 Real Images 63

8.6 Conclusions 71

9 Sampling generates different distributions **72**

9.1 Introduction 72

9.2 Distribution varies between image classes 72

9.3 Conclusions 74

10 Directions for future research **75**

11 Conclusion **78**

Bibliography **80**

Chapter 1

Introduction

1.1 Overview

The goal of this dissertation is to develop a general system for solving certain classes of low level vision problems.

Computer vision can be broken down into various parts, or levels. At the lower levels are those tasks that operate at the pixel level with the images—tasks such as noise removal, image enhancement and edge detection. At the higher levels are those tasks which work with more symbolic information—tasks such as image segmentation, and object recognition. This thesis aims at solving tasks from the lower levels.

The standard paradigm for developing an image processing system is to develop a complex and specific system to solve problem at hand. In contrast, the general system described in this thesis is able to learn how to solve any task from within some subdomain of problems, by using training images from the domain of that task. The subdomain of problems in which this system will work are those problems where the task to be performed is localized in nature, has a binary desired result, and has an available set of training images.

This general system is driven by parameters. However, in contrast to many other systems, the parameters which define how it operates do not need to be explicitly provided. Rather, it acquires these parameters from a set of training images. The system can be adapted to applications simply by providing an appropriate set of training images. In general, the system will only work in domains where the operation to be performed is local in nature. And in its current implementation, the system is limited to those domains where the "ground truth" is binary.

The basic method of acquiring these parameters is to "learn" them by counting the frequency of occurrences of each possible labeling of small sections (called *neighborhoods*) of the image. In the simplest case, just count the occurrences of each neighborhood in the training set. Then, to perform the task, update pixels in the image so as to leave more of the common neighborhoods, and fewer of the less common neighborhoods, guided by the initial image and some general knowledge about the domain. This technique is effectively that of Markov random fields (MRFs).

There are some problems with standard MRF methods. They usually require an explicitly specified noise model. And they operate with some restrictive model for the neighborhood probabilities. These problems imply that a specialized system needs to be built for each domain in which MRFs are being used. What this thesis sets out to do is to generalize the MRF scheme, so that one method will work for a variety of domains.

1.2 Markov Random Fields

A Markov random field (MRF) is a set of sites, with a certain probability structure to the possible labelings of those sites. That probability structure defines the dependencies between the labelings. In particular, for a labeling of the sites to form an MRF, it is required that the labeling of a particular site be dependent only on the labelings of some neighborhood of that site, and not on the labelings of the entire set of sites. The original definition of an

$$\boxed{\begin{matrix} I & I & I \\ I & I & I \\ I & I & I \end{matrix}} : 0.21667 \boxed{\begin{matrix} I & I & B \\ I & I & B \\ I & I & B \end{matrix}} : 0.11250 \boxed{\begin{matrix} B & I & I \\ B & I & I \\ B & I & I \end{matrix}} : 0.11806$$
$$\boxed{\begin{matrix} B & B & B \\ I & I & I \\ I & I & I \end{matrix}} : 0.09722 \boxed{\begin{matrix} I & I & I \\ B & B & B \\ I & I & I \end{matrix}} : 0.07777 \boxed{\begin{matrix} B & B & B \\ B & B & B \\ B & B & B \end{matrix}} : 0.00000$$
$$\boxed{\begin{matrix} I & I & I \\ I & I & B \\ I & I & B \end{matrix}} : 0.00277 \boxed{\begin{matrix} I & B & B \\ I & B & I \\ B & B & I \end{matrix}} : 0.00139 \boxed{\begin{matrix} I & I & B \\ B & B & B \\ I & I & I \end{matrix}} : 0.01111$$

Figure 1.1: Examples of neighborhoods with probabilities. (These are probabilities from an image of overlapping, upright rectangles.)

MRF is due to Dobruschin [Dobruschin, 1968; Geman and Geman, 1984].

In the domain of image processing, consider those sites to be the pixels of a two dimensional grid. Then the Markov random field describes the local structure of an image by defining a neighborhood system on the pixels of the image and a probability density function over the labelings of those pixels.

For example, consider an image where all pixels are labeled either as containing an object boundary B or lying entirely in the interior of an object I. The Markov random field for this example describes the probability of various local structures of I's and B's—in particular, figure 1.1 shows some possible examples of local neighborhood probabilities.

MRFs were introduced into common usage for image processing by Cross and Jain [Cross and Jain, 1983], Besag [Besag, 1986] and Geman and Geman [Geman and Geman, 1984]. The concept is to have a distribution over neighborhoods, and to use that distribution to solve images, using a variety of methods. (See chapter 2: Previous Work). However, MRFs impose one restriction on the distribution: it must obey the Markov property. What this means is that the conditional probability of a given pixel labeling, given only the labeling of its neighborhood, is the same as the conditional probability given the labeling of the entire image. This means that all relevant information is locally contained, within the neighborhood.

One way of representing such distributions is with cliques. A clique is a totally connected subgraph. The clique distribution method essentially assigns probabilities to each of all the possible labelings of all the cliques that fit within the neighborhood structure. Distributions stored as cliques are very compact, and also, by the Hammersley-Clifford theorem [Besag, 1974], the Markov property is guaranteed to hold. One can convert between clique representation and regular representation in a variety of ways [Swain, 1990; Sher *et al.*, 1993]. The system described in this thesis does not use cliques to store the distribution.

This thesis develops and explores methods of extending and improving MRFs for use as a general image processing system, and gives examples in the areas of image reconstruction, edge detector enhancement and edge detection.

1.3 Double Neighborhood MRFs (DNMRFs)

To solve the problems described earlier, this thesis defines a technique called double neighborhood MRFs (DNMRFs). They are MRFs which code information about the given images as well as the "ground truth" images from the training set.

In the literature, for example [Geman and Geman, 1984; Blake, 1989; Szeliski, 1989; Swain, 1990; Milun and Sher, 1991; Sher, 1991c], researchers dealing with MRFs have generally assumed that the following three items are available to the optimizer[1]:

Item 1: The image, as it was scanned/sampled

Item 2: The MRF probability density function

Item 3: A model for the noise in the image

[1] "Optimizer" refers generally to whatever algorithm is being used to process the image, such as simulated annealing or iterated conditional modes (ICM) [Besag, 1986].

Since item 1 is what the image processing system is given to work with, it requires no further study. Therefore much of the research has been concentrated on generating good models and representation schemes for item 2 (see chapter 2). As for item 3, it has usually been assumed that

Assumption 1: The noise has a simple model, such as independent bit errors

Assumption 2: The noise model is available to the optimizer

However, assumptions 1 and 2 are not accurate for real world applications. Assumption 1 is not necessarily true, and is definitely false in the case of binary images generated by edge detectors, as their errors are almost always correlated in some way. Assumption 2 is actually a very non-intuitive assumption: if one is given an image, it is an extremely non-trivial task to obtain from that image a model of the noise, so that it can be given to the optimizer.

The problem with assumption 1 was solved by modifying item 2. By sampling neighborhoods from noisy images together with their corresponding true images, the method encodes the noise model right into the MRF probability density function, forming a DNMRF. This neighborhood structure allows for an encoding of a general model of the noise, and provides a natural means of estimating the noise model.

Consider, for example, the output of an edge detector. Even state of the art edge detectors will not give exactly the correct edges of an image (assuming that there even exists such a concept)—there might be gaps in the edges, spurious edges, etc. However, assume that one is given the true edges of the image. If one incorporates information from both the noisy and true edges, one can train the system to correct errors.

This thesis demonstrates that this method is a useful and viable method for performing some set of image processing tasks.

1.4 Overview of the thesis

Chapter 2 reviews some of the work done in similar areas by other researchers. Various incarnations of the method have been used successfully to perform various tasks: image enhancement (chapters 3 and 4), edge detector enhancement (chapter 6), and edge detection (chapter 8). The number of training images required was studied, both in the simple MRF case, and for DNMRFs (chapter 5). In much of the work a technique of Hancock and Kittler's [Hancock and Kittler, 1990c] is used to smooth the sampled distributions. Chapter 3 contains a description of the implementation of their technique. Chapter 7 demonstrates a technique whereby the size of the distribution can be reduced by taking rotational invariance into account. Chapter 9 demonstrates that changing the training set indeed changes the domain in which the method operates. Finally, chapter 10 discusses some possible directions of future research based on the ideas contained in this thesis.

Chapter 2

Previous work

This section reviews the most relevant research in edge relaxation and binary image reconstruction. It demonstrates the relationship between previous work in these areas and this thesis. Also, even though this thesis does not set out to create a new edge detector, various edge detection algorithms are used, so a brief overview of edge detection research is included.

2.1 Edge Relaxation and Binary Image Reconstruction

Improving the output of an edge detector can be considered a form of binary image reconstruction. Here two approaches to this problem that are most relevant to this project are considered: (1) Markov random field models, (2) probabilistic relaxation.

2.1.1 Markov Random Field Models

MRF models are of continuing interest as a basis for image analysis. Their main use has been in image reconstruction and segmentation, as well as texture modeling. They were originally applied to image domains by M. Hassner and J. Sklansky, G. Cross and A. Jain, D. and S. Geman, and J. Besag [Hassner and Sklansky, 1980; Cross and Jain, 1983; Besag, 1986; Geman and Geman, 1984]. A collection of MRF papers has recently been published by Chellappa and Jain [Chellappa and Jain, 1993].

Many researchers have followed up on this research, and have developed various techniques for solving images when the local neighborhood characteristics are specified as an MRF. To solve an image is to optimize it, according to some local optimization criterion, and some method of combining the local optimizations into a global optimum.

Various estimators have been used to consider a reconstructed image as a good fit to an MRF probability distribution. The measures of goodness include MAP [Besag, 1974; Besag, 1986; Geman and Geman, 1984; Goutsias, 1989; Jeng and Woods, 1990; Karssemeijer, 1990; Geman *et al.*, 1990; Goutsias, 1991; Johnson *et al.*, 1991; Tan *et al.*, 1992; Bedini and Tonazzini, 1992; Lugosi, 1992; Banerjee and Rosenfeld, 1992], and MPM [Marroquin, 1985; Marroquin *et al.*, 1985; Sher, 1991b; Geiger and Girosi, 1991]. The MAP (maximum *a posteriori*) estimate comes from Bayesian decision theory, and represents the approximation to the true image which maximizes the conditional probability P(true image | measured image). The MPM (most probable marginals) estimate improves MAP, in that it takes into account not only the overall maximal probability value, but also considers the number of mislabeled image pixels. The MMAP estimate used by Gimel'farb and Zalesny [Gimel'farb and Zalesny, 1991] is the same as the MPM estimate.

Various methods of generating these estimated images have been used. For example, simulated annealing (the Gibbs Sampler) [Geman and Geman, 1984; Bedini and Tonazzini, 1992; Blake, 1989; Metropolis *et al.*, 1953; Basu *et al.*, 1989; Lakshmanan and Derin, 1989], Iterated Conditional Modes (ICM) [Besag, 1974; Besag, 1986; Geman *et al.*, 1990; Karssemeijer, 1990], Highest Confidence First (HCF) [Chou, 1987; Swain *et al.*, 1990] and various refinements and modifications to parts of these methods, for example [Swain, 1990; Sher, 1991b; Johnson *et al.*, 1991].

However, not enough attention has been paid to the methods of deriving the probability density functions of the MRFs.

Simulated annealing is a technique that overcomes the requirement that the search space be convex—it is a method that will not converge to a local optimum, but rather will find the global optimum. It has been extensively studied in the domain of image reconstruction and edge detection [Jeng and Woods, 1990; Blake, 1989; Geman and Geman, 1984; Marroquin *et al.*, 1987]. Simulated annealing converges with high probability to the MAP probable labeling. But in most of the simulated annealing work, the authors assume the existence of a distribution, usually in the form of clique potentials, with which to anneal. This thesis eliminates that need, by generating the MRF distribution from a set of training images.

This thesis does not use clique potentials to define MRFs. However, it is interesting to see how these potentials, and neighborhood probabilities are related. Swain [Swain, 1990] studies a method of converting probability density functions in neighborhood probability form into clique potentials. His method is exact for images whose neighborhood structure forms a chordal graph, but is only approximate for other neighborhood structures. It is interesting to note that the standard 8-connected neighborhood graph is not chordal. But Swain, too, does not pay much attention to how the original neighborhood probabilities are calculated. Sher *et al.* [Sher *et al.*, 1993] demonstrates another way of converting to and from cliques, by solving an over specified set of linear equations using a pseudoinverse method.

In this thesis, in order to be able to represent the distribution of neighborhoods in various classes of objects, the MRF probability density functions are generated by sampling.

However, there is another way to generate them for various domains. Sher [Sher, 1991a; Sher, 1991c] approaches this problem of generating the density function by theoretically dropping objects of known shape into an image, and statistically calculating what the probabilities of the various neighborhoods would be.

Woods *et al.* [Woods *et al.*, 1987] and Manjunath and Chellappa [Manjunath and Chellappa, 1991] estimate a Gaussian random field using local image statistics. They use Markov models to control how the field parameters can vary over the image.

Veijanen [Veijanen, 1991] finds the Markov random field parameters by searching for parameters of a field that would generate the same cooccurrence matrix as provided by sample images. His work is similar in goals to that of this thesis. His approach, however, is more indirect—he iterates between sampling probabilities, and generating a field which has similar features to the original, and then sampling again. He does this to deal with inaccurate observations.

Gimel'farb and Zalesny [Gimel'farb and Zalesny, 1993] explore generating images by using MRFs with either only 2 pixel cliques, only 3×3 neighborhoods, and both together. They generate some interesting images using these techniques. However, they do not try to model the parameters from real images—they decide beforehand what the form of the neighborhood model will be.

Elfadel and Picard [Elfadel and Picard, 1994] develop a set theoretical extension to standard MRF techniques, and use this extension to demonstrate an analogy between the resulting pixel labelings, and the mixing and separating of fluids in a multicomponent solution. However, they require that the type of model be specified (such as isotropic or anisotropic) and this implies predefined values for the clique potentials. They are not trying to find clique potentials in terms of their extensions; rather they are expressing the Gibbs energy function in terms of both.[1]

[1] Their linear form of the Gibbs energy function is $\sum_{g,g' \in \Lambda} V(g,g') m(S_g, S_{g'})$ which still contains free

Gray, Kay and Titterington [Gray *et al.*, 1992] follow up on work by Derin and Elliott [Derin and Elliott, 1987] on the estimation of MRF parameters using a method called the logit method, which is based on a histogram count. MRF parameters are sampled from the image being solved, and are used in an ICM reconstruction. The sampled values are converted into clique potentials, using one of the standard simple clique models. The sampling ideas are similar to those used in this thesis. What this thesis does differently is to solve the problem of sparse distributions by using a technique by Hancock and Kittler (described elsewhere in this chapter, and also in chapter 3) to smooth them, rather than just naively forcing the zero entries to be non-zero. Also, their method trains only from the single image which is trying to be optimized, rather than from a set of training images. This can be seen as both an advantage and a disadvantage: their method does not require a (possibly difficult to obtain) training set; however if such a set exists, it does not draw from the larger set of histogram sampling data. Also, they do not do double neighborhood MRFs, which limits the possible noise types which they can model.

The idea of sampling images to generate operators is not new. In 1968, Saraga and Woollons [Saraga and Woollons, 1968] generated simple pattern recognition operators, such as a thinning operator, by sampling images of digitized characters.

Lakshmanan and Derin [Lakshmanan and Derin, 1989] and Manjunath and Chellappa [Manjunath and Chellappa, 1991] also estimate the MRF parameters from a single image, but they use either a simulated annealing or nearest neighbor approach to do so. They require the independent noise assumption for their method to work—something that the DNMRF method does not require.

Kim and Yang [Kim and Yang, 1991] learn MRF parameters from a set of training images. They use the error backpropagation technique from neural networks to find the set of clique potentials that most closely matches the regions found in the training images. They

references to V, the clique potential.

then use simulated annealing to generate an image segmentation. Their technique yields reasonable results.

Hebert and Malagre [Hebert and Malagre, 1994] apply MRF methods to build an edge detector. In fact, they use the Sobel edge detector as an initial estimate, and then use a simulated annealer to generate a MAP estimate, based on a set of predetermined 3×3 neighborhood configurations. This is very similar to chapter 4 of this thesis, a previous version of which was published in 1992 [Milun and Sher, 1992b]. The method in this thesis differs from their method in that in this thesis, the 3×3 neighborhood configurations to be used are not chosen by hand, but are sampled from a set of training images, and different neighborhoods have probability values based of their frequency, rather than just the 0 or 1 values that their method assigns. The method in this thesis is therefore more general, and applicable to domains other than edge detection, whereas their method is hand tuned for edge detection only. Their paper is interesting in that it describes the 3×3 neighborhoods as the cliques of the fifth-order 5×5 neighborhood, thereby removing the issue of converting from the neighborhood form into the clique potential form.

2.1.2 Probabilistic Relaxation

Hummel, Zucker, and Rosenfeld [Rosenfeld *et al.*, 1975; Zucker *et al.*, 1975; Hummel and Zucker, 1980; Rutkowski *et al.*, 1981; Rosenfeld and Smith, 1981; Fekete *et al.*, 1981; Hummel, March 1983] were responsible for much of the earlier work on probabilistic models and networks for vision. Another early pioneer in this area is Faugeras [Faugeras and Berthod, 1981; Faugeras and Price, 1981]. Price [Price, 1985] and Kittler and Illingworth [Kittler and Illingworth, 1985] survey the early work on relaxation and compare their algorithms on a realistic image analysis problem.

More recently Hancock and Kittler have revived the probabilistic relaxation approach to edge relaxation [Kittler, 1986; Kittler and Hancock, 1987; Hancock and Kittler, 1990a;

Hancock and Kittler, 1990b; Hancock and Kittler, 1990c]. They use more sophisticated prior distributions than previous work and have substantially improved natural edge images using their approach. Their method for handling neighborhoods that do not occur in their dictionary, their so called non-physical labelings, is used in this thesis to solve the problem of sparse MRF distributions. Even though storing all possible neighborhood labelings, as is done in this thesis, is not what their papers advocate, their techniques are still applicable here. See chapter 3 for details on how their technique is used.

2.2 Edge Detection

A general discussion of the evolution of edge detection algorithms is given in any of the various computer vision or image processing texts, such as [Pratt, 1978; Ballard and Brown, 1982; Haralick and Shapiro, 1992]. This section just serves as an very brief overview of some of the more relevant algorithms.

The oldest edge detection algorithms are gradient based—they detect edges by looking for places where the pixel intensity in the image changes. They are often implemented by convolving the image with some set of masks, and considering the point an edge if the value output of the convolution with the masks is large enough at that point. By using a selection of oriented masks, the method can get an estimate of the orientation of the edge as well. Some examples of this type of edge detector are [Roberts, 1965; Sobel, 1970; Prewitt, 1970; Kirsch, 1971]. The main problems with this type of edge detector is that they are sensitive to noise, and that they generate thick edges. The thick edges can sometimes be corrected for, by some sort of thinning algorithm.

Another class of edge detection algorithms are those based on zero crossings. These usually use the Laplacian operator, and the most well known algorithm is the Laplacian of a Gaussian, developed by Marr and Hildreth [Marr and Hildreth, 1980].

Canny's edge detector [Canny, 1986] is based on a filter which is optimum for finding step edges in the presence of Gaussian noise. This filter is calculated by convolving the image with a 2-D Gaussian. Maxima in the resulting image correspond to zero crossings in the Laplacian, and represent the location of the step edges. Many current edge detectors draw from the ideas introduced by Canny, and operate in a manner similar to Canny's edge detector.

Chapter 3

Improving sampled probability density functions

3.1 Introduction

Sampling the distribution of labeled neighborhoods in a single or collection of test images is a natural way to estimate a Markov random field. However the probabilities of extremely improbable labelings cannot be accurately computed by this method, because the small probabilities dictate an unreasonably large number of test images. This chapter tests a suggestion of Hancock and Kittler [Hancock and Kittler, 1990c] for supplying the probabilities of such labelings by smoothing the estimated probability density. This chapter presents experiments determining the level of smoothing that optimizes the performance of binary image reconstruction with Markov random fields using both Besag's ICM algorithm and simulated annealing.

[1] A previous version of this chapter is published in Pattern Recognition Letters, October 1993 [Milun and Sher, 1993c]

3.2 Theoretical Implementation

This application uses a standard, single field, MRF. Neighborhoods are sampled from typical images, and the sample frequency of each neighborhood labeling is calculated. Conditional probabilities that determine whether to turn a pixel on or off are generated from these true probabilities by the following formula:

$$\forall \vec{\mathcal{N}} \qquad P(L_x | L_{N_x} = \vec{\mathcal{N}}) = \frac{P((L_x)\&(L_{N_x} = \vec{\mathcal{N}}))}{P((L_x)\&(L_{N_x} = \vec{\mathcal{N}})) + P((\neg L_x)\&(L_{N_x} = \vec{\mathcal{N}}))}$$

where x is the pixel under consideration, L_x is its labeling and N_x is the neighborhood of x which has labeling $\vec{\mathcal{N}}$. So $P((L_x)\&(L_{N_x} = \vec{\mathcal{N}}))$ is the probability of the neighborhood with the center pixel set to L_x, and $P((\neg L_x)\&(L_{N_x} = \vec{\mathcal{N}}))$ is the probability of the neighborhood with the center pixel not set to L_x.[2] This calculation is done separately for each pixel in the image.

The conditional probability formulae which are used to extract the conditional probabilities from the true probabilities can easily be extended to deal with the neighborhoods at the boundaries of the image. The neighborhood used, N_x^*, is only part of the complete 8-connected neighborhood, and thus $P((L_x)\&(N_x^* = \vec{\mathcal{N}}))$ is the sum of all the neighborhood probabilities which have labeling L_x, the pixels in N_x^* as given, and all other pixels in $N_x - N_x^*$ either on or off.

The MRF determined by the above method serves as a prior probability distribution. The posterior probability is calculated by taking the prior and multiplying it by the probability of a false positive or negative. This probability is calculated from the noise model. In this research, this noise level is specified by the probability that a white pixel was turned black,

[2] This assumes that both $P((L_x)\&(L_{N_x} = \vec{\mathcal{N}}))$ and $P((\neg L_x)\&(L_{N_x} = \vec{\mathcal{N}}))$ cannot be zero at the same time, or else it will produce a division by zero. To force this not to be the case, all the probability densities were modified to replace all zero entries with 10^{-15}, and then normalized to sum to one again. However, as discussed in section 3.3, this is no longer necessary when the Hancock-Kittler technique is applied.

and the (not necessarily equal) probability that a black pixel turned white.

The most obvious method to use when reconstructing an image from the posteriors, is a gradient descent technique, such as Besag's ICM [Besag, 1986]. This method selects a pixel of the image at random, and checks whether making that pixel white or black will increase the posterior probability, and sets the pixel accordingly. This algorithm iterates until the image no longer changes.

This method tends to find a local minimum, but not the global minimum. Simulated annealing is a standard technique that can overcome this problem.

The simulated annealing method described in [Geman and Geman, 1984] was employed. This is the same as the "heatbath" algorithm described in [Blake, 1989]. The simulated annealer modifies the calculated posterior probabilities at each step by including a quantity which corresponds to the physical notion of temperature in actual annealing. This temperature causes pixel values to sometimes be changed in a way that will increase the energy of the system. This allows the annealing process to overcome local minima, and to find the global minimum of an energy function.

During the process of annealing, this temperature is lowered according to an "annealing schedule". Geman and Geman [Geman and Geman, 1984] prove that if one uses the schedule that they suggest, the simulated annealing is guaranteed to attain the global minimum. The problem is that their schedule can take centuries to complete. A truncated schedule similar to that suggested in [Blake, 1989], is therefore used. The schedule used is as follows:

$$T_i = \frac{T_0}{log(2+i)}$$

where i is the iteration number, and T_0 was chosen to be 4. At each iteration, the algorithm chooses a random pixel in the image to possibly update.

This temperature, T, is shown in the literature [Geman and Geman, 1984; Blake, 1989] to be combined into a Gibbs distribution as follows:

$$P(x) = \frac{e^{-U(x)/T}}{Z}$$

where $U(x)$ is the energy (from clique potentials) of having x set, and Z is a normalizing constant.

Since this method is not using cliques, the following equivalence is used:

$$E = -\log(P)$$

where E is energy, and P is probability. The derivation of the probability annealing formulae is below (ignoring the normalizing constant):

$$P(x) = e^{-U(x)/T} = e^{\log(P)/T} = e^{\frac{1}{T}\log(P)} = e^{\log(P)^{\frac{1}{T}}} = P^{\frac{1}{T}}$$

Thus all that must be done to calculate the annealing probabilities is to raise them to the power $\frac{1}{T}$.

Proceed by calculating the prior probability, P_{prior1}, of the pixel in question being a one, the prior probability, P_{prior0}, of the pixel in question being a zero, and then using P_{noise01} and P_{noise10}, the probabilities that noise changes a zero into a one, or a one into a zero respectively, from the given noise model, to yield the following set of formulae:

$$P_{\text{coinflip}} = \frac{1}{1+P}$$

where, if the given noisy image has a one in the pixel location being considered, then:

$$P = \left(\frac{P_{\text{prior0}} \times P_{\text{noise01}}}{P_{\text{prior1}} \times (1 - P_{\text{noise10}})}\right)^{\frac{1}{T}}$$

and if the given noisy image has a zero in the pixel location being considered, then:

$$P = \left(\frac{P_{\text{prior0}} \times (1 - P_{\text{noise01}})}{P_{\text{prior1}} \times P_{\text{noise10}}} \right)^{\frac{1}{T}}$$

Once the posterior probability at a pixel has been calculated, P_{coinflip} above, a random number is chosen, and the pixel is turned on or off probabilistically.

This is repeated until the image does not change significantly over some number of iterations. "Not change significantly" is defined as less than 1% of the pixels in the image changing. The images are compared between every N^2 possible pixel changes. (In fact, because the image comparison takes time, the process runs for N^3 iterations before the images are compared at all.)

3.3 Hancock and Kittler's technique

In theory, the method discussed in the previous section should work. However, the results do not stand up to theoretical expectations. Because the given probability density function was very sparse, there are many cases in which the calculated priors are very close to the default value of 0.5, since many neighborhoods do not occur at all in the sampled image. In that case there is no information from which to estimate the conditional probabilities.

The probability density is sparse because it is being generated from an artificial image consisting of, for example, vertical and horizontal lines only. In such an image, a very small number of the possible 512 different 3×3 neighborhoods occur at all. So most of the probability density function has value zero, and most of the weight in the distribution is on a few 3×3 neighborhoods, particularly the all-zero neighborhood.

Hancock and Kittler [Hancock and Kittler, 1990c] suggested a solution to a similar problem. They want to be able to generate probabilities for arbitrary neighborhood labelings, given only the probabilities of some small number of neighborhood labelings.

Their technique makes the assumption that the neighborhood labelings that are known, were in fact extracted from an image that was noisy, with known noise distribution. Their method essentially applies this known noise model to the probability density function. It assumes that pixels are independently flipped, with a probability p, and so mathematically compensates for this flipping. This, in effect, smoothes the density function. In their implementation, only the valid neighborhoods are stored (the "dictionary" in their terminology), and the probability of each non-dictionary neighborhood is calculated as it is needed, during image reconstruction.

Their scheme is applicable to the situation described in this thesis: the probability values are known for only some neighborhood labelings (because only those labelings existed in the training sample), and probability values for all other labelings are needed.

One modification was made to their scheme: the probability of each of the 512 8-connected neighborhood labeling is precalculated before running the optimization algorithm. Thus the algorithm is identical to what was used before, but it is just being fed a relaxed ("smoothed") probability density.

The calculation of the smoothed probability density function is as follows: For each neighborhood labeling i, let the smoothed probability of i be the sum over all neighborhood labelings j:

$$P'(i) = \sum_j P(j) \times p^{d(i,j)} \times (1-p)^{u(i,j)}$$

where $d(i,j)$ is the number of bits that differ between i and j, and $u(i,j)$ is the number of bits that are unchanged between i and j. This mathematically expresses the idea that any neighborhood is possibly a noisy version of any other neighborhood. The value p is what is referred to as the Hancock-Kittler (or HK) smoothing level.

Note that even the probability values of the neighborhood labelings that occurred in the

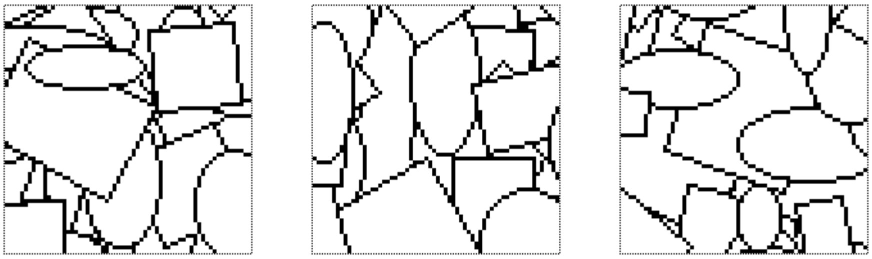


Figure 3.1: Some instances of generated random images.

sampled distribution are changed.

Using this scheme with appropriate smoothing levels (values for p), the accuracy of the reconstructions have been improved. Also, the relationship between the optimum value of p and the signal to noise level in the image is empirically calculated.

3.4 Results

A program which generates 64×64 images with 50 randomly placed, overlapping ellipses and randomly rotated rectangles all of random size was used to generate test data. 100 of these images were generated, the occurrences of the 3×3 neighborhoods were sampled, and those sampled distributions were averaged to get a probability distribution. Some examples of these images are shown in figure 3.1.

Figure 3.2 shows this probability density smoothed by various amounts. The first graph, (a), shows the probability density sampled from a single image. The other three show the averaged probability density generated from the 100 random images, smoothed by various amounts. (b) shows no smoothing, (c) shows a smoothing level of 0.025, and (d) shows a smoothing level of 0.20. Note how the large peaks are reduced, and the weight of the probability is spread out, as the amount of smoothing is increased.

Figure 3.3 shows noisy versions of an image from figure 3.1, which were used as some of the input images to the optimization algorithm. For the noise in these images, the probability

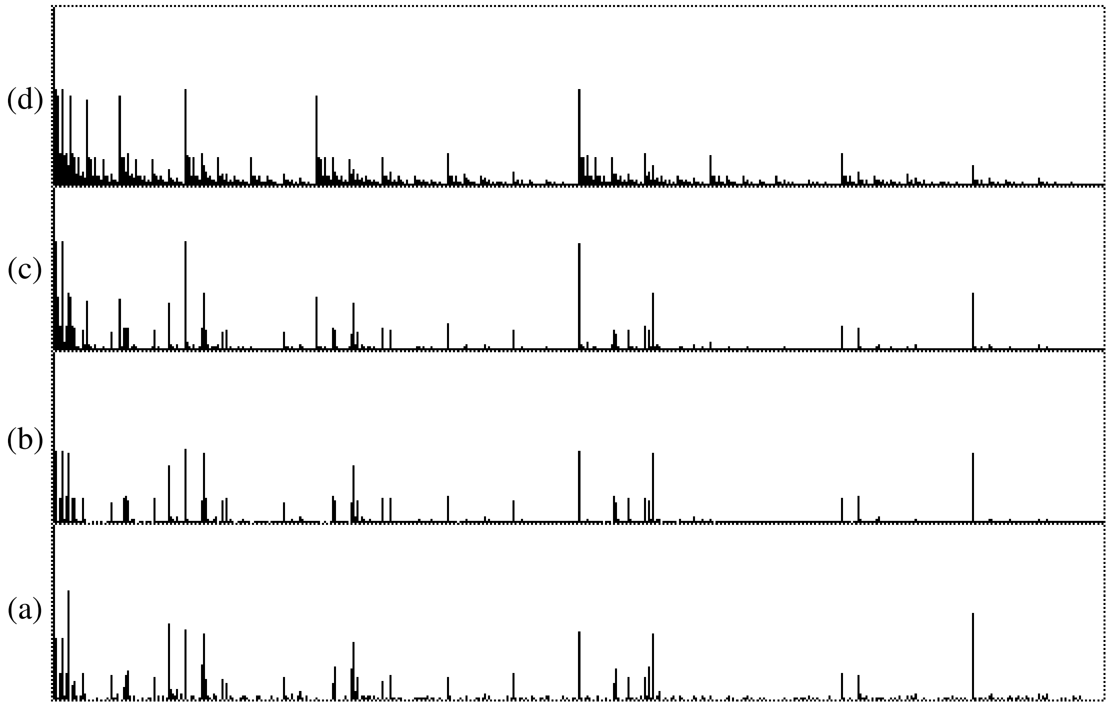


Figure 3.2: Plot of some probability density functions. The x-axis ranges over the neighborhoods. (The neighborhoods are numbered in a standard lexical order, treating the 3 rows of pixels in a neighborhood as the high, middle and low octal digits in a 9 bit number. Thus the all-zero neighborhood along either axis is number 0, and all-ones is number 511, and the neighborhood with only a single one in the center pixel, is number 16 (being 020 in octal).) The y-axis represents the probability. (a) is probability density function sampled from a single random image. The averaged probability density function generated from 100 random images (as described in the text) is shown with no smoothing in (b); with 2.5% smoothing in (c); and with 20% smoothing in (d).

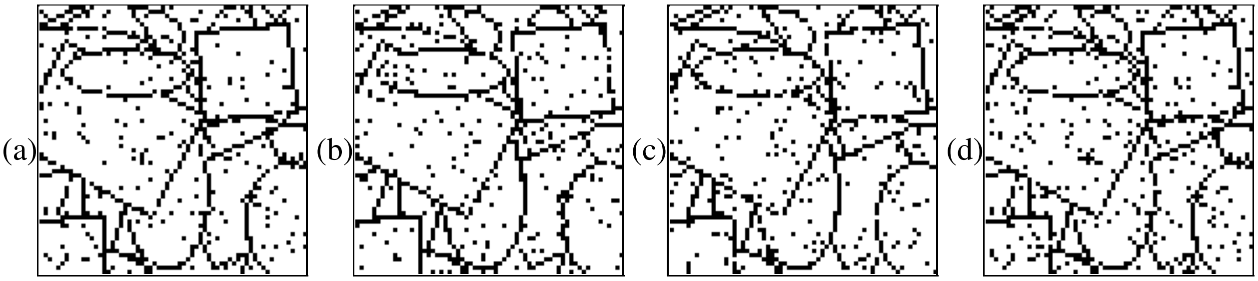


Figure 3.3: Images with noise added. (a)–(c) are the same image, each with 6% noise generated by flipping pixels. (d) has 8% of the pixels flipped.

of a white pixel being turned black is equal to the probability of a black pixel being turned white. These pixel flips are all independent. The first three images have a noise level of 6%, and the last image has a noise level of 8%. Tests were carried out with noise levels varying from 1% to 8% (see below).

The following set of runs were performed: at each noise level from 1% to 8%, in steps of 1%, 15 sets of runs with the implementation of the ICM algorithm, and 15 sets of runs with the simulated annealer. Each of these sets consisted of runs with Hancock-Kittler smoothing at levels from 0.000 to 0.275. The 15 runs at each noise level were all performed on the same random image, but used different random seeds for both the noise-adding process, and for the ICM or simulated annealing algorithm.

To show that the results hold for other random images of the same type, two more images were generated. Noise of 4% and 6% were added to each, and these images were run through the standard batch of 15 sets of runs.

As examples of the images reconstructed by the algorithms, see figure 3.4 for some generated output from the ICM algorithm, and figure 3.5 for some generated output from the simulated annealer. This output was all generated using the sampled probability density function and the image shown in figure 3.3(a).

Image (a) in each of figures 3.4 and 3.5 is what was generated with no smoothing of the probability density function. Subimages (b)–(e) show the reconstruction of the same

Figure 3.4: Sample runs of the ICM algorithm. The image being reconstructed is shown in figure 3.3(a). The images here are all reconstructions of the same image—the difference between them is the level of Hancock-Kittler smoothing applied to the sampled distributions. The HK levels used are: (a) No HK; (b) 0.022; (c) 0.038; (d) 0.100; (e) 0.250.

Figure 3.5: Sample runs of the simulated annealing algorithm. The image being reconstructed is shown in figure 3.3(a). The images here are all reconstructions of the same image—the difference between them is the level of Hancock-Kittler smoothing applied to the sampled distributions. The HK levels used are: (a) No HK; (b) 0.022; (c) 0.038; (d) 0.100; (e) 0.250.

image, but using distributions smoothed by various amounts. Notice that if the amount of smoothing is too great, both algorithms do no better (but also no worse) than the given noisy image.

Although the results generated when using distributions that have a small amount of HK smoothing are generally visually better than the images reconstructed using unsmoothed distributions, a quantitative measure was required. In this experiment, the obvious (and easiest) metric is simply to count the number of pixels that differ between the regenerated image, and the original noise-free image (which the annealer or ICM algorithm never sees). This can be expressed as a percentage of the total pixels in the image.

Using this metric, there were on average 558 pixels incorrect (out of 4096) (13.6%) when annealing on images with 6% noise, and using no Hancock-Kittler smoothing; but only about 135 pixels (3.3%) incorrect when the Hancock-Kittler smoothing level was 0.038. Similar results were achieved at other noise levels.

Figure 3.6 shows graphically the effect of Hancock-Kittler smoothing on the resulting reconstruction of the image with 6% noise. The graphs at other noise levels are very similar, but with the depth and x-position of the global minimum changing as noise level changes.

Figure 3.7 shows the same graph but using ICM rather than the simulated annealer. The graph exhibits similar behavior to the simulated annealer graph, also having a minimum at approximately the same noise level. This was constant at all tested noise levels.

The minimum on the ICM graph is at approximately the same HK level as the minimum on the annealer graph. However, the ICM graph achieves this minimum on a much larger range of HK levels than the annealer does. This means that when using the ICM algorithm, selecting an HK smoothing level is less critical than when using the simulated annealing algorithm.

The relationship between the noise level and the optimum level of Hancock-Kittler smoothing, was studied. This relationship is represented by the position of the minimum in the above graphs—the HK level at which the minimum occurs. Figure 3.8 shows plot of this optimum level, as calculated over all the runs of the simulated annealer that were performed. The thick line in the figure is the minimum of the mean results (over the 15 sets of runs) at each noise level. The upper and lower thin lines are the greatest and smallest Hancock-Kittler smoothing levels at which the minimum occurred, at each noise level. A simple linear regression estimates that the optimum Hancock-Kittler smoothing level to be 0.61 times the noise level.

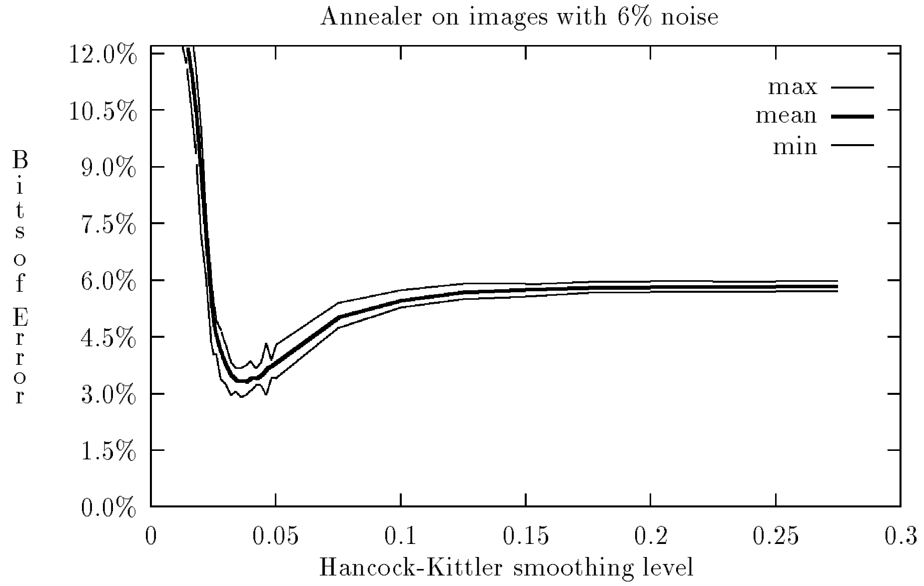


Figure 3.6: Number of pixels that the annealer got incorrect, as measured between the image it produced and the original image. This graph shows the mean number of errors measured at each noise level (over the 15 runs using different random seeds in both noise generation and in the annealer itself). It also shows the maximum and minimum number of errors measured at each noise level in any of the runs.

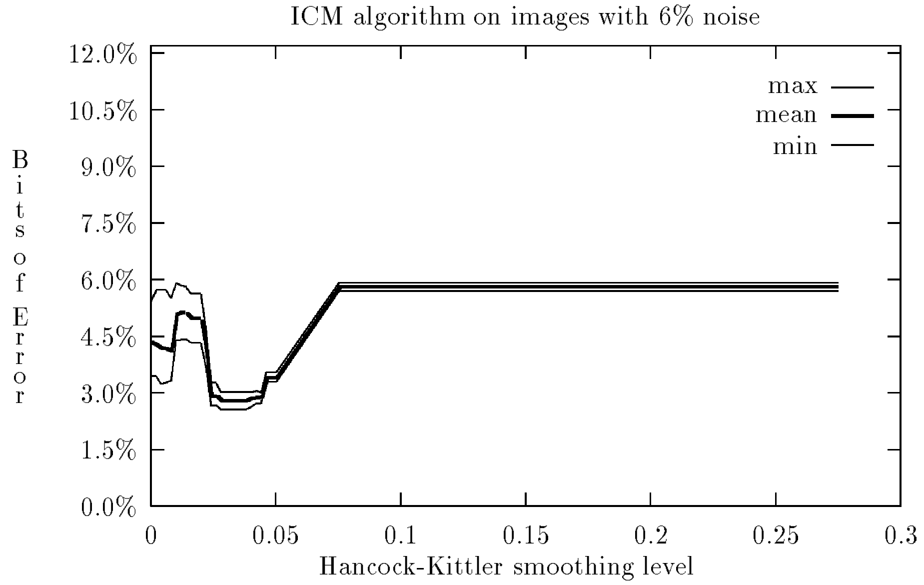


Figure 3.7: Number of pixels that ICM got incorrect, the image it produced and the original image. This graph shows the mean number of errors measured at each noise level (over the 15 runs using different random seeds in both noise generation and within the ICM algorithm itself). It also shows the maximum and minimum number of errors measured at each noise level in any of the runs.

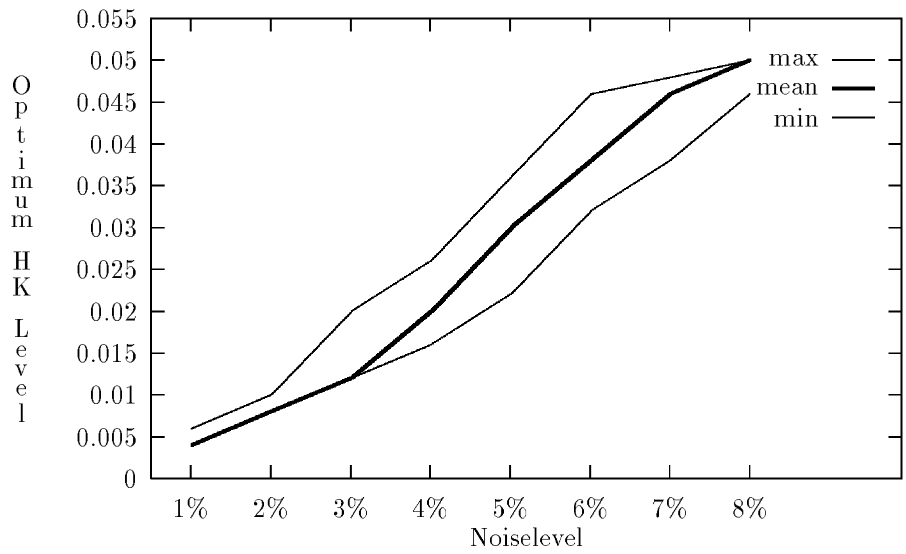


Figure 3.8: Optimum levels Hancock Kittler smoothing for various noise levels.

3.5 Conclusions

This chapter demonstrated that the smoothing technique of Hancock and Kittler [Hancock and Kittler, 1990c] is applicable to MRF distributions generated by sampling neighborhoods from a training set of images. Using distributions smoothed by their technique yields image reconstructions which are visually and statistically superior to images reconstructed using the unsmoothed distributions. It was found that, in the domain of binary edge images with random independent binary noise, the Hancock-Kittler smoothing level should be 0.61 times the true noise level.

Chapter 4

Binary Image Enhancement — DNMRFs

4.1 Introduction

Chapter 1 discussed the issue of obtaining a reasonable, non-trivial, noise model, by looking at original images together with noisy images, and so creating a probability density function over pairs of neighborhoods across both images. Any localized noise can be modeled. This arbitrary noise model is contained within the MRF probability density function. Doing this provides an easy way to generate MRF distributions for annealing or other relaxation methods. This technique is referred to as the double neighborhood MRF (DNMRF) technique, and is demonstrated in this chapter.

[1] A previous version of this chapter appeared in the ICPR 92 conference [Milun and Sher, 1992b]

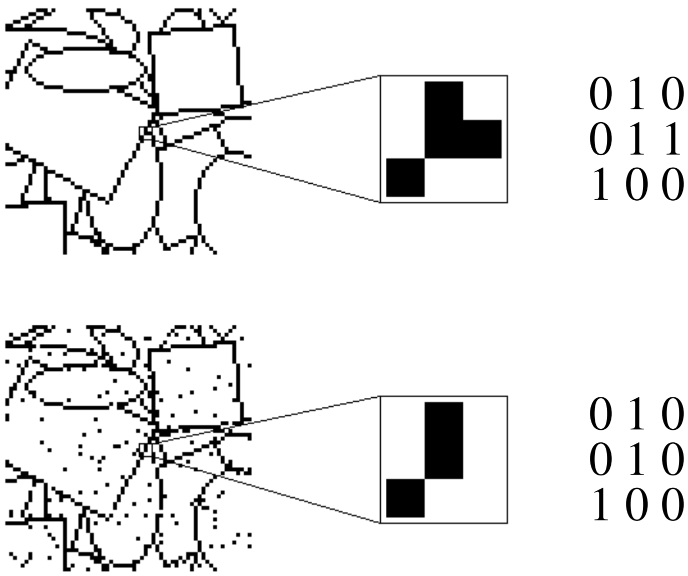


Figure 4.1: Example of a double neighborhood pair. The uncorrupted image is shown above, and the noisy, corrupted image is shown below. Enlarged is one neighborhood pair. The existence of this particular neighborhood pair would be recorded in the probability density function in position 010010100010011100_2 (reading the corrupted image as the high bits) which is decimal 75932.

4.2 Theoretical Implementation

One way to represent the DNMRF probability density over the possible pairs of binary 3×3 neighborhoods, is to store an array of 2^{18} probability values. Each probability value represents the probability of a particular pair of corrupted and uncorrupted neighborhoods existing. See figure 4.1.

This distribution can be generated the same way as the single field MRF distribution was generated—by sampling. But, this time, sample from both images at once—a 3×3 neighborhood from the given, noisy image, and the 3×3 neighborhood in the same position in the ground truth image.

The method used to reconstruct the corrupted image, is as follows: Store two images—the given corrupted image, Y, and a new image, into which the reconstruction, X, is going to be built. Throughout the program run, Y is never changed—only X is updated. The ICM algorithm performs the updates, deciding when it is an improvement in the image to flip the value of a pixel. (Contrast this with the method of chapter 3, where the given corrupted image is itself updated as the algorithm progresses.)

The algorithm checks whether $\text{Prob}(X_c, Y) > \text{Prob}(X, Y)$, where X_c is X with the pixel under consideration changed. If the probability of the image is greater (or equivalently, since $E = -\log(P)$, the total energy of the image is less) when the pixel under consideration is flipped, then the algorithm flips that pixel. Otherwise, it randomly selects another pixel. By the Markov property, these probabilities can be calculated by considering only the probabilities of the local neighborhoods, with and without the pixel at the center of the neighborhood flipped. This update algorithm is essentially the ICM algorithm of Besag [Besag, 1986].

This process of checking and flipping pixels is repeated until the image no longer changes at all.

This DNMRF can also be used for simulated annealing (such as with the Gibbs Sampler of Geman and Geman [Geman and Geman, 1984], or the heatbath algorithm of Blake [Blake, 1989]) by noting that the ratio

$$\left(\frac{\text{Prob}(X_c, Y)}{\text{Prob}(X, Y)}\right)^{\frac{1}{T}}$$

(where $\frac{1}{T}$ is the temperature term), is the energy ratio.

Section 3.3 discussed the use of Hancock and Kittler's [Hancock and Kittler, 1990c] technique to solve the problem of the sampled MRF distribution being sparse, primarily because the set of training image does not contain examples of all possible neighborhood labelings. In the case of DNMRF distributions, the sampled distribution is even more sparse—because the number of neighborhood labeling instances being sampled for is larger by a factor of 512 in the 3×3 case. So, the resulting distribution is almost all zero. Once again, the Hancock-Kittler smoothing technique is used to remedy the situation. However, two issues arise in the DNMRF case.

The first is computation speed. Since the smoothing is calculated by summing over all the probability values to calculate each new probability value, it is O(n^2) in the number of probability values. This was not a problem when there were 512 values, but with 512^2 values it becomes an issue. The solution turned out to be the Hadamard transform. The HK smoothing operation can be done in O(n) time in Hadamard space, and the fast Hadamard transform takes O($n\log(n)$) time. This technique worked so well that it was implemented for generating the single field HK smoothing as well.

The other issue is whether the whole distribution is considered as one large set of probabilities when calculating HK. This was found not to be the case. Theoretically and empirically, HK was shown to work better if just applied to each row to the DNMRF table individually. This method was used, and in all references to HK in DNMRF, that is what is referred to.

4.3 Results

The same sample image generation program was used. This time 1000 images were generated, and each had 4% independent random bit errors added. They were then sampled for the occurrences of the pairs of 3×3 neighborhoods, and the probability distribution for each averaged to get an overall probability distribution. Examples of these images are shown in figure 3.1.

Figure 4.2 shows this probability density function with and without Hancock Kittler smoothing applied. Note that when the HK smoothing is applied, the large peaks are reduced, and the weight of the probability is spread out. Whereas without HK most of the

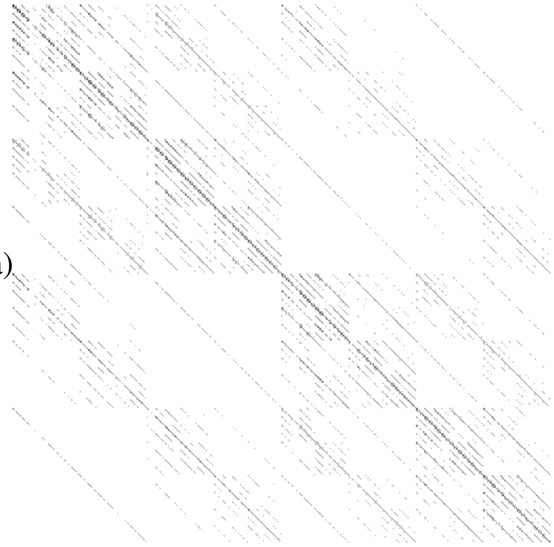
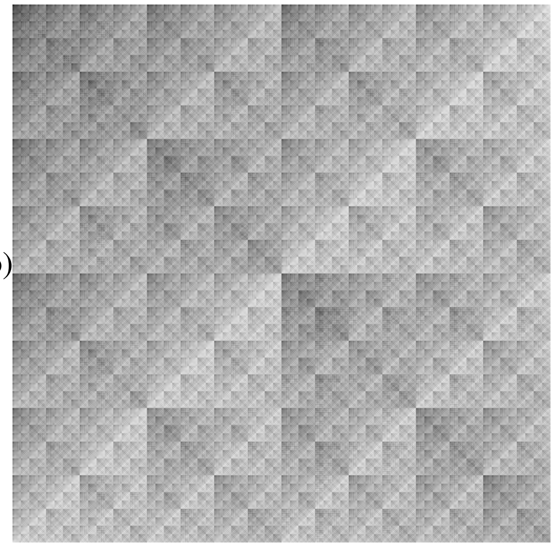
(a)  (b) 

Figure 4.2: Plot of a probability density function.
The x-axis ranges over the neighborhoods from the true image. The y-axis ranges over the neighborhoods from the noisy image[a]. The plots show a scaled logarithm of the probability. Subimage (a) has no Hancock-Kittler smoothing applied. Subimage (b) has Hancock-Kittler smoothing at the 0.10 level.

[a]The neighborhoods along each axis are numbered in a standard lexical order, treating the 3 rows of pixels in a neighborhood as the high, middle and low octal digits in a 9 bit number. Thus the all-zero neighborhood along either axis is number 0, and all-ones is number 511, and the neighborhood with only a single one in the center pixel, is number 16 (being 020 in octal).

distribution has value 0, once it has been smoothed there is a positive probability value in every cell for the distribution.

Figure 4.3 contains for some generated output from the ICM algorithm. The first row (subimages a–e) are results using an image which had 4% noise added. The original image is shown in (a), and the noisy image is shown in (b). Subimage (c) shows the reconstruction using the probability distribution which does not include Hancock-Kittler smoothing, and has 100 bits incorrect compared to image (a). This means that from 4%, there are now only 2.4% of the bits incorrect. The one using Hancock-Kittler smoothing, (d), has 85 bits incorrect (2.1%). Also shown, in subimage (e), is the reconstruction achieved by the single field methods of chapter 3 using the ICM algorithm. This image has only 74 bits (1.8%) incorrect, but the images generated with the DNMRF method are aesthetically superior.

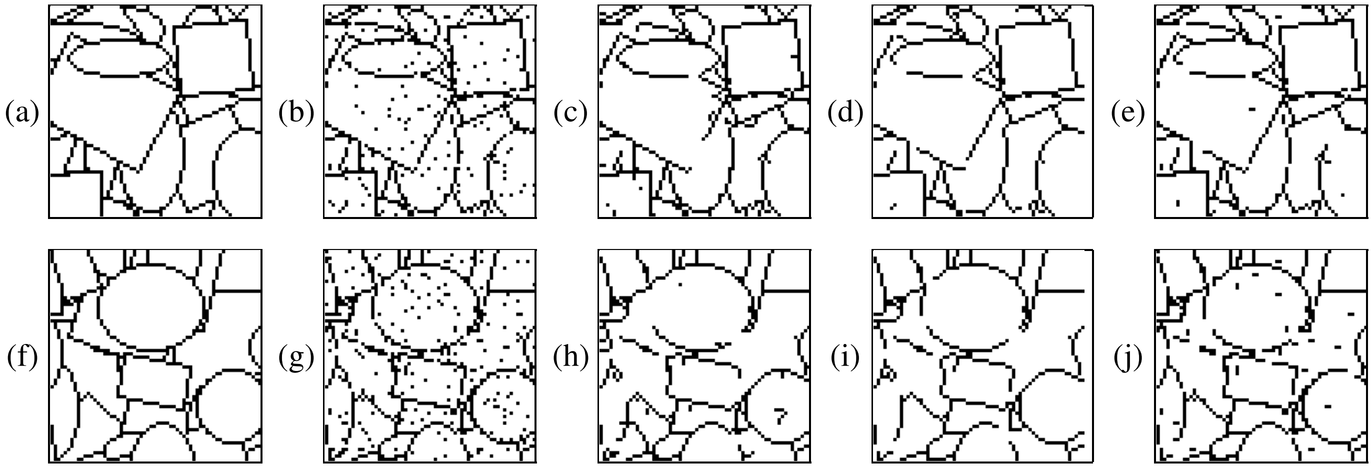


Figure 4.3: Sample run of ICM using the DNMRF. In (a)–(e) the original image has 4% noise added. In (f)–(j) the original image has 6% noise added. The images shown are: (a),(f) The original image; (b),(g) The corrupted image; (c),(h) The reconstruction without Hancock-Kittler smoothing; (d),(i) The reconstruction using a Hancock-Kittler smoothing level of 0.10; (e),(j) For comparison, the reconstruction using a single MRF with optimum Hancock-Kittler smoothing and the true noise model.

The second row (subimages f–j) has similar results, but for an image with 6% noise added. The number of bits of error were: 166 (4.1%) for double field with no HK (subimage h); 122 (3.0%) for double field with 0.100 HK (subimage i); 121 (3.0%) for single field with optimum HK (subimage j). Once again the double field reconstruction with HK is visually the best.

Figure 4.4 demonstrates how important an accurate noise model is to a single field MRF reconstruction. When the ICM algorithm is given noise models other than the true 6%, the reconstruction is degraded. This shows that even though the single field method can achieve somewhat better results than the DNMRF method *when the ICM algorithm is given the correct noise model*, it does not do well when that model is not known (or is not representable by a simple process). This is the big advantage of the DNMRF method—the noise model does not need to be known explicitly, as it is incorporated implicitly into the DNMRF during training.

Figure 4.4: Reconstructions of an image with 6% noise, using single neighborhood MRF models. In each image, the ICM algorithm is run using the same distribution, but is given different values for the noise level. The noise level that the ICM algorithm is given is: (a) 1%, (b) 2%, (c) 6%, (d) 10% and (e) 20%. Note that the reconstruction is good only when this matches the true noise level of 6%.

4.4 Conclusions

This chapter demonstrates that the DNMRF technique works. In the example shown, the actual noise model is in fact simple random bit noise. This allows us to compare the DNMRF method with the single field method discussed in chapter 3. The DNMRF method performs almost as well as that method.

However, the big advantage of the DNMRF method is that the noise model does not need to be explicitly known, as it is incorporated into the distribution at training time. In contrast, the single field method degrades considerably when the noise model is incorrectly specified.

Further applications of the DNMRF method are demonstrated in following chapters. For example, chapter 6 shows the method used to clean up edges generated by an edge operator, and chapter 8 shows how it is used to actually perform edge detection.

Chapter 5

How much training data is required

5.1 Introduction

This chapter addresses the issue of how large the set of training images needs to be. This chapter investigates how many pairs of training images are required to generate a useful and consistent set of local neighborhood probabilities, for both single field and double field MRFs.

5.2 Single field MRFs

As in the previous chapters, the test data generator for the single field study creates 64×64 binary images with 50 randomly placed, overlapping ellipses and randomly rotated rectangles all of random size. The overall training set contains 2048 such images; the test set contains 128 such images, but with 6% of their bits randomly flipped.

[1] Previous versions of this chapter appeared in the 1993 Orlando SPIE symposium [Milun and Sher, 1993b] and in the Vision Interface 93 conference. [Milun and Sher, 1993a]

Training	no HK		HK=.010		HK=.036		HK=.050	
set size	Mean	Stddev	Mean	Stddev	Mean	Stddev	Mean	Stddev
2	8.5532	1.973	6.6563	1.430	3.7036	0.867	3.6609	0.366
4	6.9001	1.610	5.9468	1.289	3.2842	0.651	3.6331	0.362
8	5.5754	1.220	5.8892	1.367	3.2568	0.639	3.6360	0.362
16	5.2292	1.119	5.8184	1.274	3.0271	0.429	3.6335	0.363
32	4.9031	1.033	5.9285	1.262	3.0281	0.429	3.6353	0.362
64	4.8047	1.004	5.8894	1.298	3.0300	0.430	3.6340	0.361
128	4.7180	1.032	5.9502	1.297	3.0259	0.429	3.6326	0.362
256	4.7581	0.994	5.8755	1.292	3.0291	0.429	3.6338	0.361

Table 5.1: Numerical results of the single field study using the bits of error as the metric. The numbers shown are the percentage of bits that differ, out of the 4096 bits in the 64×64 image. Each row contains the results generated with a set training pairs of one size, but the results are the average over 4 different sets of training pairs of that size. All reconstructions were performed on the same set of 128 test images.

In order to gauge the performance of the method, the metric of the number of bits that differ between the reconstructed image and the true image, was used.

Table 5.1 shows the numerical results, using training images of size 64×64. By the way the neighborhoods are collected during training, an image of this size yields 3472 neighborhood instances. From the data in the table, the amount of training data required can be summarized as follows: For no HK, there is no statistically significant improvement after 32 images. For the case of HK = 0.010, there is no statistically significant improvement after 4 images. For the case of HK = 0.050 it does not matter how many training images were used.

For the most interesting case, when HK is optimal at 0.036 (being 0.6 times the true noise level, as discussed in chapter 3), there is no statistically significant improvement after 16 training images were used. Figure 5.1 shows some sample runs of this case, and figure 5.2 shows the plot of the results with 99% confidence interval error bars.

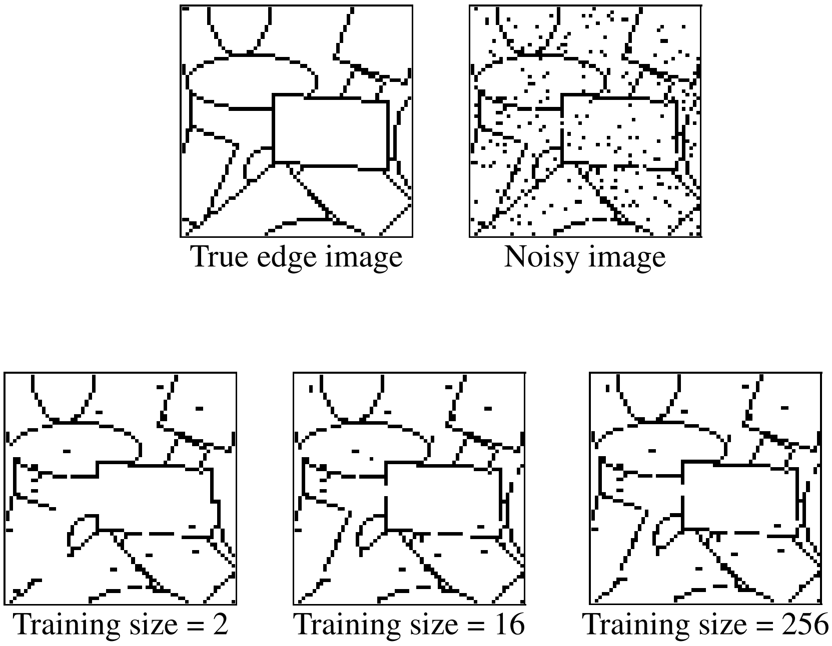


Figure 5.1: Sample runs of single field case with Optimal HK.

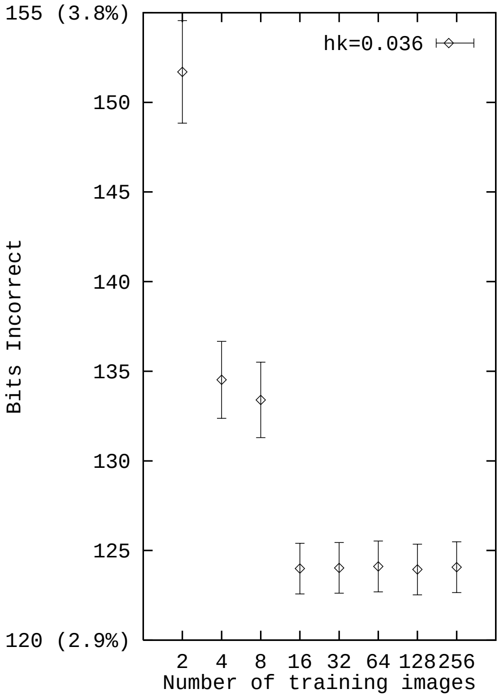


Figure 5.2: Single field results using optimum Hancock-Kittler smoothing level. The 99% confidence intervals of the means are plotted.

Figure 5.3: Some instances of generated gray scale random images.

5.3 Double neighborhood MRFs

The test data generator for the double field study creates pairs of gray scale images together with the corresponding ground truth edge images. Each pair consists of randomly placed, randomly sized, circles of random solid color, and an edge map representing those pixels through which an edge passes. Some examples of these images are shown in figure 5.3. The training set contains 2048 such pairs of gray scale images. The images used during testing were random pairs of images, generated as they were needed during the test runs. Each image was corrupted with additive Gaussian noise of standard deviation 16 and then corrupted with JPEG compression down to 1.5 bits per pixel (bpp). A thresholded, thinned Sobel [Sobel, 1970; Duda and Hart, 1973] edge detector was run on the set of doubly corrupted images. The DNMRF distribution was trained on the pairs of true edge images and edge images of the corrupted images. An example image can be found in figure 5.4.

An ICM [Besag, 1986] algorithm was applied to the test set at a variety of Hancock-Kittler smoothing levels. Figure 5.5 shows the results with HK=0.010 using distributions sampled from training sets of 3 different sizes.

In order to gauge the performance the metric of the number of bits that differ between the reconstructed image and the true image was used.

Table 5.2 shows the numerical results. For double fields, the amount of training data

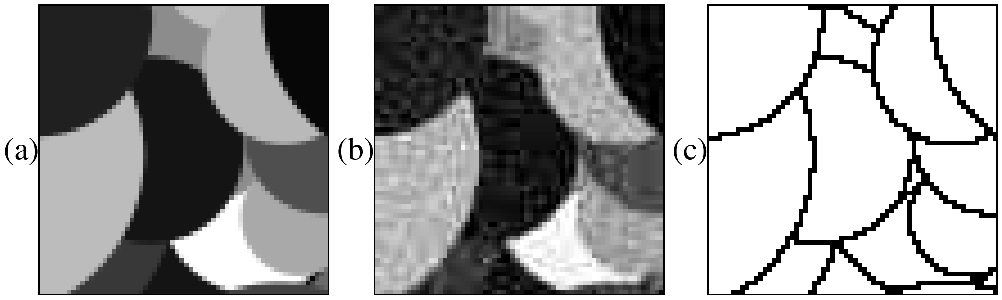


Figure 5.4: Noisy images. (a) A noiseless image; (b) An image with standard deviation 16 Gaussian noise added and then compressed using JPEG to 1.5 bpp; (c) The true edge in this image

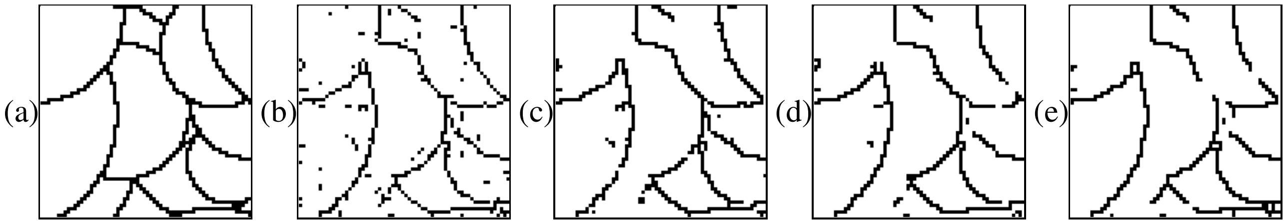


Figure 5.5: Sample run of the ICM algorithm using the DNMRF distribution. (a) The true edge image; (b) The thinned Sobel; (c) The reconstruction based on training the distribution on 2 images; (d) on 4 images and (e) on 256 images.

required can be summarized as follows: For no HK and HK = 0.06, there is no statistically significant difference regardless of the number of training images used. For the cases where HK = 0.010 – 0.050 there are no pairwise significant differences, however there is a statistically significant difference between the very low and very high number of sampled images. This seems to imply that even 256 images are not a sufficient number, and that even better results could be achieved by using a larger training set. The HK = 0.050 results are displayed in figure 5.6.

5.4 Summary

The amount of training data required to form a useful distribution for the reconstruction of images using the single field MRF, and the DNMRF method was studied.

Training	no HK		HK=.006		HK=.010		HK=.025		HK=.050	
set size	Mean	Stddev	Mean	Stddev	Mean	Stddev	Mean	Stddev	Mean	Stddev
2	276.16	66.83	252.91	67.74	253.88	66.02	252.58	67.69	253.04	68.64
4	272.28	74.84	247.99	64.16	244.48	66.55	243.90	67.35	241.17	64.12
8	265.48	66.45	241.57	66.85	244.07	65.13	240.96	65.87	241.19	67.50
16	271.57	67.43	242.44	66.63	236.16	67.58	241.50	66.98	243.46	68.53
32	266.98	68.50	240.20	66.56	242.04	65.29	240.09	66.47	235.72	67.00
64	274.31	70.17	243.46	63.31	240.24	66.81	237.85	65.47	236.20	63.26
128	272.10	68.76	244.65	69.12	240.54	69.24	238.78	64.28	235.06	65.75
256	275.34	71.50	242.08	70.41	240.01	65.51	235.60	63.61	232.89	64.30

Table 5.2: Numerical results of the double field study using the bits of error as the metric. Each row contains the results generated with a set training pairs of one size, but the results are the average over 4 different sets of training pairs of that size. All reconstructions were performed on a random set of 256 test images.

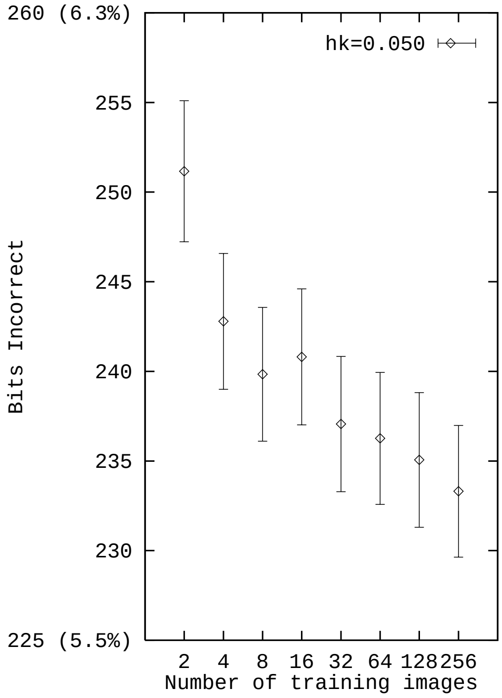


Figure 5.6: Double field results using Hancock-Kittler level of 0.050. The 99% confidence intervals of the means are plotted.

It was found that in the single field MRF case, only a few images are needed to obtain results as good as those which used as many as 256 training images. This is not unexpected, as there are only a relatively small number of classes being trained for (512, in fact), and in 4 images of size 64×64, there are 13888 neighborhood instances, which yields an average of about 27 neighborhood instances per class.

However, for DNMRFs, it was found that much larger training sets are needed. In fact, within the range of the experiment, no optimum number was found—the more training images there were, the better the results. This is easily explained by noticing that there are now many more classes being trained for. In fact, in the 3×3 neighborhood case, there are now 262144 classes. And in 256 images, there are 888832 neighborhood instances. The division now yields a value of about 3.5 neighborhood instances per class. From this one can infer that a lot more training images are required to reach the same level of accuracy in the DNMRF distribution, as is in the single field distribution—possibly even a training set 10 times as large.

Chapter 6

Edge detection enhancement — DNMRFs

6.1 Introduction

Edge detectors and other binary feature detectors generate an imperfect binary representation of the boundaries in an image. Knowledge about the structures and shapes contained in an image can be used to improve these binary images, *e.g.* the fact that object boundaries are continuous can be used to repair boundaries with holes in them.

The method described in this chapter does not specifically use that knowledge, but uses similar information—information which it derives by comparing the output of the edge detector, with images of the true edges in the objects.

One can consider these differences or errors as noise, and consider the process of improving the output of the edge detection as noise removal.

[1] A previous version of this chapter appeared in the July 1992 SPIE conference on Neural and Stochastic Methods in Image and Signal Processing [Milun and Sher, 1992a]

The correlation of this "noise" is incorporated into the DNMRF model by simultaneously sampling pairs of noisy and noiseless neighborhoods from the training images. For an example of a 3×3 pair of such neighborhoods, see figure 4.1. This chapter applies this technique to enhancing edge detection in artificial and natural images.

6.2 Measures of goodness

In order to gauge the performance of the method, two different measures were used. One is the metric of the number of bits that differ between the reconstructed image and the true image. The results from this do not, however, always match the visual sense of what is best. The second measure used was Pratt's Figure of Merit [Pratt, 1978]. This is a number in range 0 to 1, which represents how good, with 1 being perfect, an edge detector is. It takes both edge position and accuracy into account.

It is computed as follows:

$$FOM = \frac{1}{I_N} \sum_{i=1}^{I_A} \frac{1}{1 + \alpha d^2}$$

where $I_N = \max(I_I, I_A)$ and I_I is the number of pixels in the ideal edge map, and I_A is the number of pixels in the actual edge map; and α is a scaling constant (which was set to 1 in this thesis); and d is the distance from the actual edge pixel to the (nearest) ideal edge pixel.

6.3 Results on images with Gaussian noise

Test data for edge detection enhancement was generated using a program which creates pairs of gray scale images. Each pair consists of randomly placed, randomly sized, circles or tilted rectangles of random solid color, and an edge map representing those pixels through

Figure 6.1: Some instances of generated gray scale random images.

which an edge passes. Examples of these images are shown in figure 6.1. Test sets were generated for each object type (circles and tilted rectangles) consisting of 128 gray scale images. Gaussian noise with standard deviations of 4, 8 and 16 was added to each of the images. A thresholded, thinned Sobel [Duda and Hart, 1973] edge detector was run on each gray scale image yielding an edge map. The DNMRF was trained on the pairs of true edge images and Sobel edge maps, at each level of noise.

The ICM algorithm was run on these 1024 images (2 objects $\times$ 128 images $\times$ 3 noise levels and no noise), with a variety of Hancock-Kittler smoothing levels. Figures 6.2 and 6.3 show some results at the highest noise level tested—at lower noise levels the results were even better.

Subimages (f) and (g) of figure 6.2 demonstrate some of the interesting features of the reconstruction. The method does more than just remove all the disconnected pixels—it actually cleans up the existing edges. Regions A and D demonstrate how the method corrects for incorrect pixels that are near to the true edge. Region B shows that the method does what it is trained to do: in this case, the true images that it was trained on contained thick edges, so it changes the thin edge into a thick edge. Region C demonstrates the method closing a gap in an edge.

Region E shows a gap in an edge that cannot be closed by this method. The gap was probably generated in the output of the edge detector because the difference in color between

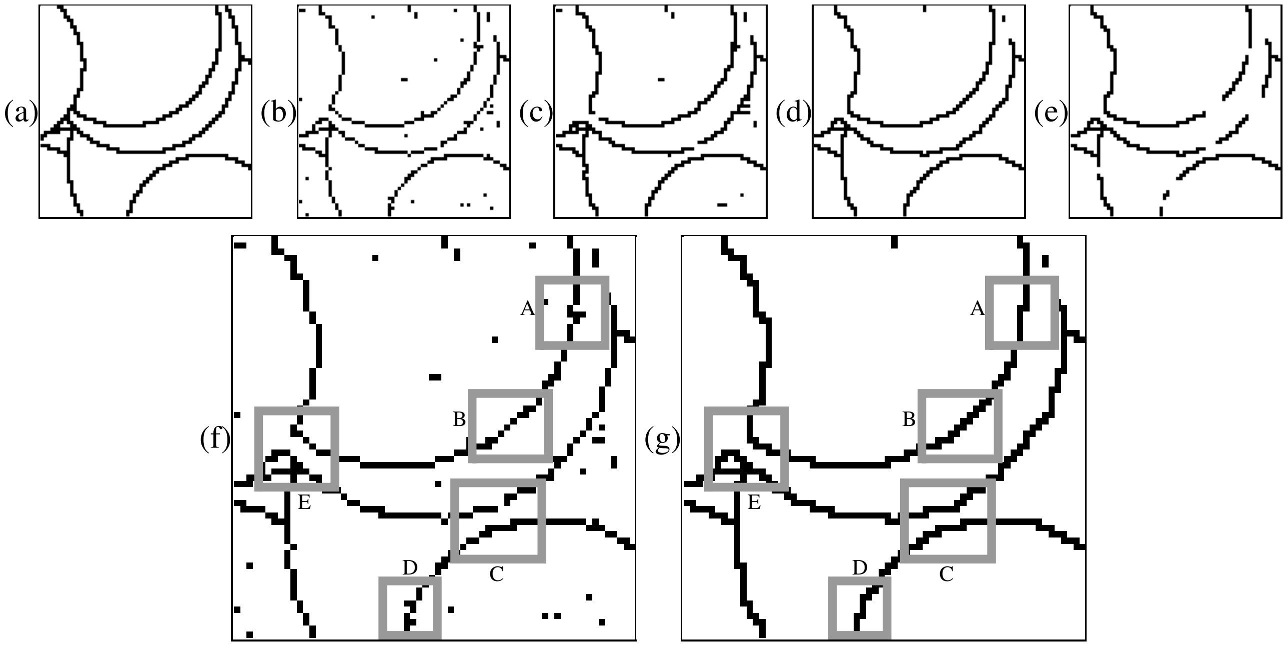


Figure 6.2: Sample run of double neighborhood MRF used for Edge Enhancement on an artificial image of circles with standard deviation 16 noise added. The 5 images shown are: (a) The true edge map; (b) The output of the thresholded, thinned Sobel; (c) A reconstruction with no HK smoothing; (d) A reconstruction with an HK smoothing level of 0.00125; (e) A reconstruction with an HK level of 0.050. (f) and (g) are enlargements of (b) and (d) respectively, with some interesting regions of the image highlighted.

the two circles was less than the gaussian noise level. And the DNMRF method cannot close it because it is too large for a 3×3 neighborhood to be able to detect as an edge. It is possible that if 5×5 neighborhoods were used, the method would be able to close that gap.

Using these two measures, this DNMRF reconstruction technique, without Hancock-Kittler smoothing, shows an improvement over the thinned Sobel. However, at the levels of Hancock-Kittler smoothing that were tested, no improvement (but also no deterioration) was gained in a numerical sense, but the reconstructions using the smoothed density functions are aesthetically improved. From the trends in tables 6.1 and 6.2 it seems that further experimentation at levels of smoothing even lower than 0.00125 is needed.

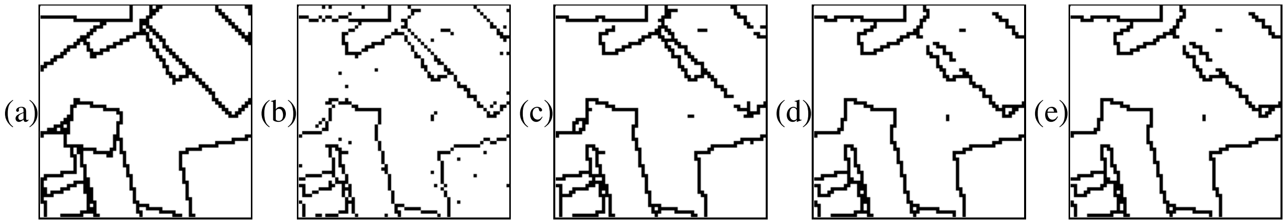


Figure 6.3: Sample run of double neighborhood MRF used for Edge Enhancement on an artificial image of tilts with standard deviation 16 noise added. The 5 images shown are: (a) The true edge map; (b) The output of the thresholded, thinned Sobel; (c) A reconstruction with no HK smoothing; (d) A reconstruction with an HK smoothing level of 0.00125; (e) A reconstruction with an HK level of 0.050.

Noise Level		Hancock-Kittler Smoothing Level						
	Thinsobel	0	.00125	.0025	.005	.025	.050	.100
Pratt's Figure of Merit								
No noise	0.764	0.881	0.860	0.856	0.859	0.828	0.838	0.686
Standard deviation 4	0.728	0.807	0.766	0.769	0.773	0.758	0.741	0.658
Standard deviation 8	0.675	0.755	0.715	0.711	0.708	0.693	0.673	0.619
Standard deviation 16	0.568	0.635	0.594	0.591	0.589	0.567	0.552	0.503
Percentage Bits of Error (of 4096 bits)								
No noise	4.113	3.576	3.415	3.389	3.348	3.362	3.303	4.229
Standard deviation 4	4.792	4.137	3.919	3.896	3.854	3.820	3.926	4.395
Standard deviation 8	5.067	4.572	4.407	4.408	4.404	4.393	4.488	4.742
Standard deviation 16	5.832	5.432	5.327	5.336	5.317	5.357	5.438	5.680

Table 6.1: Numerical results for the test set of images of circles. The approximate standard deviations of these results are given below:

	Pratt	%Bits
No noise	0.03	0.88
Standard deviation 4	0.06	1.10
Standard deviation 8	0.09	1.25
Standard deviation 16	0.11	1.46

Noise Level		Hancock-Kittler Smoothing Level						
	Thinsobel	0	.00125	.0025	.005	.025	.050	.100
Pratt's Figure of Merit								
No noise	0.786	0.858	0.792	0.794	0.789	0.785	0.771	0.701
Standard deviation 4	0.713	0.766	0.717	0.722	0.726	0.723	0.706	0.657
Standard deviation 8	0.647	0.696	0.653	0.657	0.660	0.660	0.644	0.597
Standard deviation 16	0.520	0.567	0.527	0.526	0.525	0.516	0.499	0.467
Percentage Bits of Error (of 4096 bits)								
No noise	6.742	5.941	5.708	5.686	5.623	5.581	5.744	6.399
Standard deviation 4	7.029	6.385	6.363	6.305	6.257	6.157	6.283	6.669
Standard deviation 8	7.524	7.194	7.161	7.099	7.076	6.993	7.064	7.420
Standard deviation 16	8.838	8.633	8.678	8.676	8.644	8.656	8.758	8.941

Table 6.2: Numerical results for the test set of images of tilted rectangles. The approximate standard deviations of these results are given below:

	Pratt	%Bits
No noise	0.05	1.22
Standard deviation 4	0.07	1.34
Standard deviation 8	0.09	1.54
Standard deviation 16	0.10	1.64

6.4 Compression noise

Since this chapter deals with edge detector enhancement of images with correlated, difficult to characterize noise, using images corrupted by lossy compression was a natural choice of domain, because the noise that it adds is a prototypical example of correlated noise. Segmenting such imagery is also important because JPEG is a technique that is becoming more popular, and some way is needed to remove any artifacts that the compression introduces into the segmentation.

The JPEG technique for lossy compression was chosen because it is a standardized image compression mechanism [Wallace, 1991]. JPEG is an acronym for Joint Photographic Experts Group, the name of the committee that developed the standard. The degree of image compression is usually quite large, but one pays for this in that the compressed image is not identical to the original one. However it is usually close enough that a human can't tell the difference, because it was developed to remove those features of the image that are not important to the human visual system.

Many image processing operators and techniques are sensitive to the noise introduced by JPEG. The DNMRF method can remove many of the errors in detection resulting from the image noise and the compression.

6.5 Results with compression noise

The same image generation program as in section 6.3 was used. The training set consisted of 128 pairs of gray scale images of circles, and the test set consisted of 128 different pairs of images. Each image was corrupted with Gaussian noise of standard deviation 16—call this set of images G. The clean and noisy images were corrupted with JPEG compression down to 1.5 bits per pixel (bpp)—generating sets J and GJ respectively. A thresholded, thinned Sobel [Sobel, 1970; Duda and Hart, 1973] edge detector was run on the 3 sets of corrupted

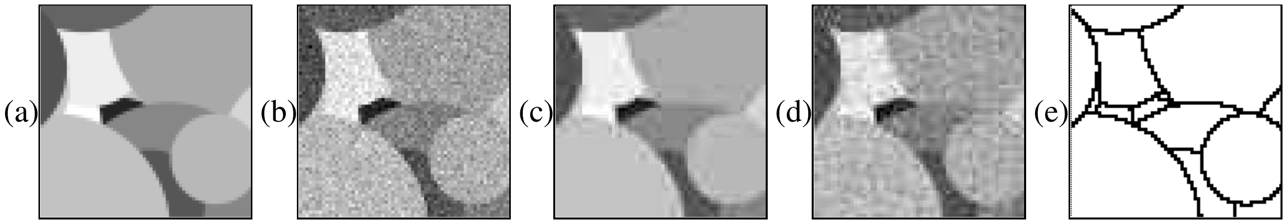


Figure 6.4: Noisy images. (a) a noiseless image; (b) an image with standard deviation 16 Gaussian noise added (from set *G*); (c) an image compressed by JPEG to 1.5 bpp (from set *J*); (d) both JPEG and Gaussian noise (from set *GJ*); (e) the true edges of this image

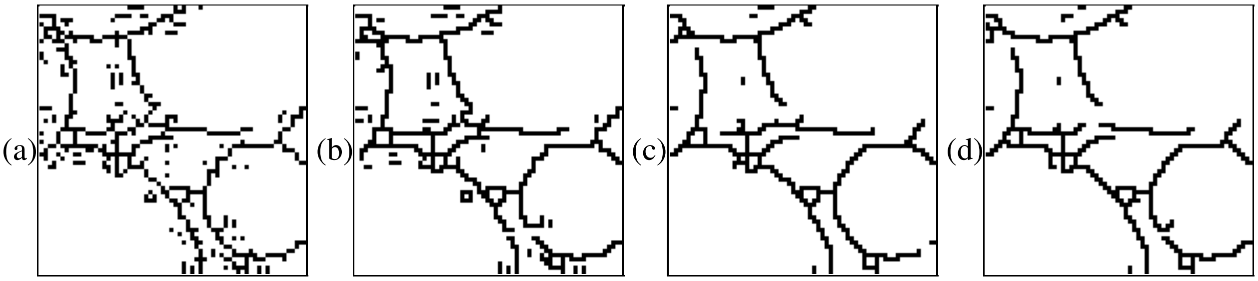


Figure 6.5: Sample run of double neighborhood MRF used for Edge Enhancement on an artificial image of circles which was compressed by JPEG to 1.5bpp (from Set *J*). The 4 images shown are: (a) the output of the thresholded, thinned Sobel; (b) a reconstruction with no HK smoothing; (c) a reconstruction with 0.010 HK smoothing; (d) a reconstruction with 0.100 HK smoothing.

images. DNMRF distributions was trained on the pairs of true edge images and elements of *G*, *J* and *GJ*, yielding three different distributions. An example of an image from each of the 3 sets can be found in figure 6.4.

An ICM algorithm [Besag, 1986] was applied to the *G*, *J* and *GJ* versions of the test set, at a variety of Hancock-Kittler smoothing levels. Figure 6.5 shows the results on an image from *J*. Figure 6.6 shows the results on two images from *GJ*. In both cases, the DNMRF method removes a large number of the spurious edges.

Using the two measures described above, this technique of reconstruction shows an improvement over the thinned Sobel. Using the Hancock-Kittler smoothing technique gained a significant improvement, both in a numerical sense, and by visual inspection. In the case of the *GJ* set (compressed noisy images), the improvement is especially notable: the

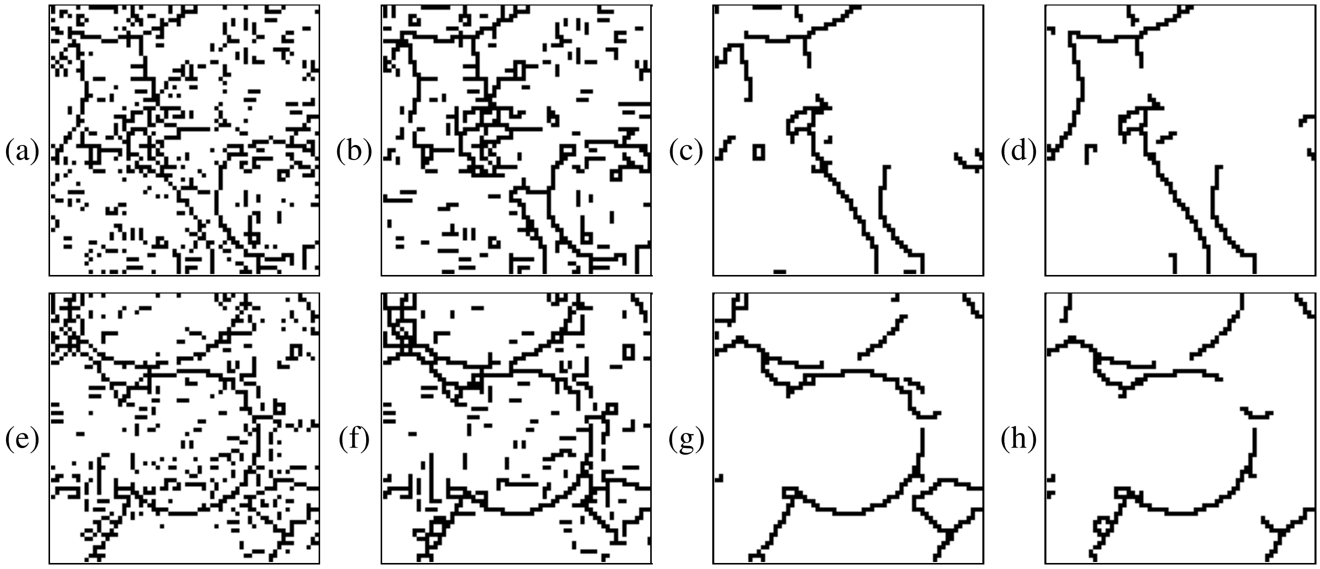


Figure 6.6: Sample runs of double neighborhood MRF used for Edge Enhancement on an artificial image of circles with standard deviation 16 noise added which was compressed by JPEG to 1.5bpp (from Set *GJ*). The 4 images shown in each strip are: (a),(e) the output of the thresholded, thinned Sobel; (b),(f) a reconstruction with no HK smoothing; (c),(g) a reconstruction with 0.010 HK smoothing; (d),(h) a reconstruction with 0.100 HK smoothing.

thinned Sobel has an mean of about 650 bits incorrect (15.9%) with a standard deviation of about 100 (2.4%); the method, with Hancock-Kittler smoothing, has a mean of just below 300 bits incorrect (7.3%) with a standard deviation of about 75 (1.8%). See table 6.3 for detailed numerical results.

6.6 Results on real images

This method has also been tested on real images. Figure 6.7 shows such an example. The original scene is a 64×64 region cut from a digitized photo. In order to have some quantitative measure for the accuracy of the edge reconstructions, an image of a feasible set of "true" edges in the image was drawn. It was generated by running an edge operator on the image, and then cleaning up the resulting edges by hand. It is shown in subimage (b).

Noise Level		Hancock-Kittler Smoothing Level					
	Thinsobel	0	.006	.010	.025	.050	.100
Pratt's Figure of Merit							
No noise	0.764	0.881	0.887	0.887	0.888	0.888	0.887
Noise with $\sigma = 16$ (Set G)	0.568	0.635	0.630	0.633	0.636	0.634	0.639
JPEG at 1.5bpp (Set J)	0.656	0.688	0.706	0.714	0.723	0.737	0.752
Noise and JPEG (Set GJ)	0.430	0.467	0.532	0.534	0.523	0.530	0.524
Percentage Bits of Error (of 4096 bits)							
No noise	4.113	3.576	3.371	3.380	3.268	3.240	3.198
Noise with $\sigma = 16$ (Set G)	5.832	5.432	5.427	5.357	5.314	5.264	5.260
JPEG at 1.5bpp (Set J)	8.612	7.878	5.878	5.803	5.726	5.672	5.535
Noise and JPEG (Set GJ)	15.840	14.410	7.350	7.258	7.086	7.002	6.792

Table 6.3: Numerical results for the test set of images of circles corrupted by JPEG and Gaussian noise. The approximate standard deviations of these results are given below:

	Thinsobel and no HK		HK>0	
	Pratt	%Bits	Pratt	%Bits
No noise	0.025	0.92	0.025	0.83
Noise with $\sigma = 16$ (Set G)	0.100	1.37	0.110	1.42
JPEG at 1.5bpp (Set J)	0.050	2.05	0.080	1.51
Noise and JPEG (Set GJ)	0.075	2.44	0.100	1.78

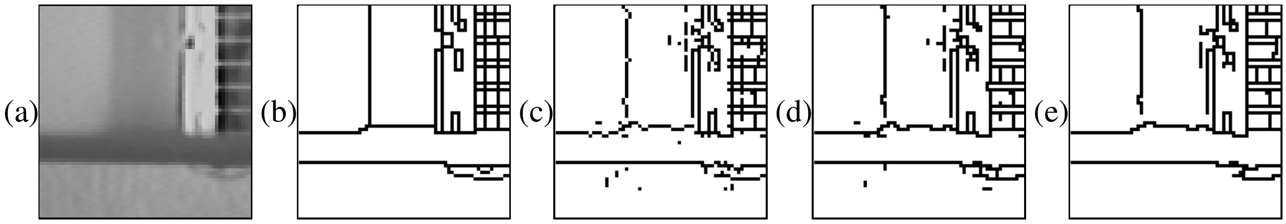
(a) 

Figure 6.7: Results of double neighborhood MRF on a natural image. The images are: (a) The original image; (b) A hand drawn bitmap of the "true" edges (based on the output on an edge operator); (c) The output of the thinned Sobel; (d) A reconstruction with no HK smoothing; (e) A reconstruction with an HK smoothing level of 0.00125.

Measure	Thinsobel	No HK	0.010 HK	0.00125 HK
% Bits different	4.565	4.468	4.346	3.760
Figure of Merit	0.873256	0.882557	0.879612	0.897282

Table 6.4: Numerical results for the real image of figure 6.7.

Subimage (c) shows the results of the thresholded, thinned Sobel on the image. (d) and (e) were generated using the DNMRF density function that was sampled from the images of tilted rectangles with standard deviation 4 Gaussian noise added. (d) had no Hancock-Kittler smoothing applied to the MRF density, and (e) had 0.00125, which was optimal of the various levels of smoothing that were tried on this image. The numerical results are given in table 6.4.

Chapter 7

Taking Advantage of Rotational Invariance

7.1 Introduction

Some image processing problems are inherently rotationally invariant. This section describes some domains that are rotationally invariant and verifies that generating a DNMRF distribution by sampling training data from within such a domain yields a rotationally invariant distribution. The chapter then goes on to show how to use this fact to both improve the quality of existing distributions, and to reduce the size of distributions for such domains.

7.2 Rotational Invariance

An image processing problem is rotationally invariant, or rotationally symmetric, if the feature being detected is equally likely to occur at any orientation. This implies that rotating the image by any amount, and rotating the operator by the same amount, will yield the same result as the unrotated case. In the continuous domain of real images, this rotation can indeed be by any amount, but once the image is digitized, there are only certain discrete rotations that are possible. In the case where the portion of the image being considered is only a 3 pixel $\times$ 3 pixel region, there are only 8 rotations—the successive 45° rotations.

Some features are inherently rotationally invariant. Edges are one such example: a real object can be scanned at any orientation, and the edge of the object will be at the same orientation as the image of the object.

In the context of this thesis, rotational invariance is useful because it provides a way to reduce the size of the distribution. When the DNMRF method is being used to implement an operator for a rotationally invariant domain, then only one instance from each rotationally invariant class of objects needs to be accounted for.

In a single field MRF, it would mean that

$$\forall \vec{\mathcal{N}} \qquad P(L_x | L_{N_x} = \vec{\mathcal{N}}) = P(L_x | N_x = R_\theta(\vec{\mathcal{N}}))$$

where $\vec{\mathcal{N}}$ is a neighborhood labeling, L_x is the labeling of point x, N_x is the neighborhood of x, and $R_\theta(\cdot)$ represents a rotation by angle θ. What this means is that the probability of a pixel being set to one is not changed by rotating the labeling of its neighborhood.

In the DNMRF case, it means that

$$\forall \vec{\mathcal{N}}\, \forall \vec{\mathcal{T}} \qquad P(L_x | L_{N_x^n} = \vec{\mathcal{N}}, L_{N_x^t} = \vec{\mathcal{T}}) = P(L_x | L_{N_x^n} = R_\theta(\vec{\mathcal{N}}), L_{N_x^t} = R_\theta(\vec{\mathcal{T}}))$$

where N_x^n is the neighborhood in the noisy image, and N_x^t is the neighborhood in the true image.

This is usually referred to in terms of the marginal probabilities: it is said that

$$P(\vec{\mathcal{N}}, \vec{\mathcal{T}}) = P(R_\theta(\vec{\mathcal{N}}, \vec{\mathcal{T}}))$$

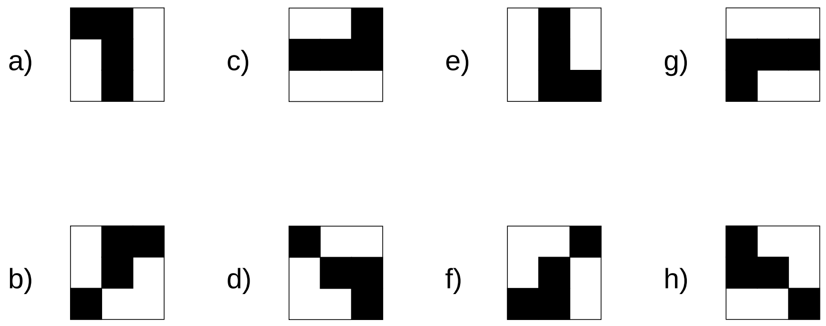


Figure 7.1: The 8 45° rotations of a particular neighborhood labeling.

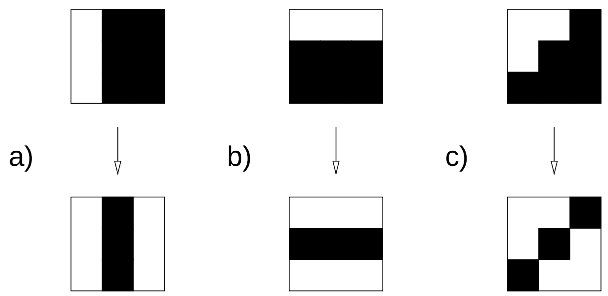


Figure 7.2: Some neighborhood rotations in the DNMRF case. Each lower image represents the edge in the corresponding upper image.

or even just that the neighborhood labeling $\vec{\mathcal{N}}, \vec{\mathcal{T}}$ is rotationally invariant.

As described above, the level of discretization, in particular the size and shape of the neighborhood, effects which values of θ are meaningful. In the case of the 8-connected 3×3 neighborhood, one can consider the 90° rotations, making θ come from the set $\{0, 90, 180, 270\}$, or the 45° rotations, making θ be from the set $\{0, 45, 90, 135, 180, 225, 270, 315\}$.

For example, in figure 7.1, (a), (c), (e) and (g) are the 90° rotations, and (a)–(g) are the 45° rotations. Figure 7.2 shows some pairs of rotated neighborhoods in the DNMRF case. (a) and (b) are examples of 90° rotations, and (c) shows a 45° rotation.

7.3 Rotationally Invariant Domains

There are some domains, such as face images or road signs, in which the images are not rotationally invariant—because some edge orientations are much more likely to occur in the image, or have special meaning. However, in cases where the orientation of the camera is (close to) uniformly distributed, such as aerial photography, or images of object dropped at random on a conveyer belt, the edges become rotationally invariant.

So, in cases when edge orientation can be ignored, edge detection is a rotationally invariant domain. For general images there should be no reason that edges of any one orientation are more or less likely than edges of any other orientation, nor that a pattern of pixels in one orientation is any more or less likely to represent an edge than the same pattern in another orientation.

To test this hypothesis, a DNMRF distribution was sampled from a collection of 256 pairs of images of solid circles and their true edges. Rotations were performed on the resulting distribution, and the number of rotationally invariant neighborhood labelings were counted.

In this case, since the true edge neighborhood labeling, $\vec{\mathcal{T}}$, is the row index into the table of distributions: $\mathcal{R}^{\vec{\mathcal{T}}} = \mathcal{D}_{\vec{\mathcal{T}},\cdot}$. Consider the maximally probable neighborhood labeling within that row:

$$rowmax(\mathcal{R}^{\vec{\mathcal{T}}}) = \max_{0 \leq i \leq 511}(P(\mathcal{R}^{\vec{\mathcal{T}}}{}_i))$$

What is being tested here is whether or not

$$R_\theta(rowmax(\mathcal{R}^{\vec{\mathcal{T}}})) = rowmax(\mathcal{R}^{R_\theta(\vec{\mathcal{T}})})$$

for each applicable value of θ, for each row $\vec{\mathcal{T}}$.

Count the number of values of $\vec{\mathcal{T}} = \{0 \ldots 511\}$ and $\theta = \{0, 45, 90, 135, 180, 225, 270,$

315} for which the above equation holds. This gives a total number of $512 \times 8 = 4096$ instances[1].

Using this method, and the DNMRF distribution generated from circles as described above, the following results were obtained: 76% of the neighborhood labelings were to be rotationally invariant. When considering the 45° and 90° rotations as separate classes, 95% of neighborhood labelings were rotationally invariant with respect to the 90° rotations, but of the 4 orientations that occur only in the 45° case, only 56% were rotationally invariant.

Using the same set of circle images, but with 5% random noise added to the image of the circles, the numbers came out as follows: 82% were rotationally invariant, breaking down into 93% for the 90° and 71% for the 45° rotations.

It is interesting to note that none of these numbers change much, if at all, when the HK smoothing is applied to the distributions. This reconfirms the fact that applying the HK smoothing operation to a distribution does not change the nature of the distribution.

This shows that with respect to 90° rotations, the data is definitely rotationally invariant. The small percentage of instances that empirically are not rotationally invariant can be explained by assuming that the sample set is too small, and that if the distribution were sampled from a larger set of images, then those numbers would be 100%.

With respect to the 45° rotations, this seems to indicate that the data might not be rotationally invariant. This is most likely caused by the fact that discretization leads to imperfect digital images. It causes the 45° rotations to sometimes look quite different from the 90° rotations, as is evident from looking at figure 7.1. In particular, subimages (a) and (b) seem to have a different topological structure: the "elbow" in (a) seems to no longer exist in (b).

[1] Of course, some of these are rotational duplicates of others, but this should simply introduce a factor 8 into these numbers, which should not effect the overall results.

However, as will be shown below, forcing the DNMRF distribution to be 45° rotationally invariant tends to produce better results for edge detection.

7.4 Improving Distributions through Rotational Invariance

Assuming that a distribution is indeed rotationally invariant (because it was sampled from a rotationally invariant domain), this fact can be used when sampling the training images. Whenever a pair of neighborhoods is sampled, rather than increasing one "cell" in the distribution by 1, increase the 8 corresponding rotationally invariant "cells" by $\frac{1}{8}$. This effectively multiplies the amount of available training data by a factor of eight, because $\frac{1}{8}$ the number of categories are being trained for, using the same data.

Note that this can also be done from an already sampled distribution, by creating a new distribution, where each "cell" is calculated by averaging the 8 corresponding rotationally invariant "cells." This technique is used in chapter 8.

7.5 Reducing the Size of Distributions

One big advantage of rotationally invariant domains, is that they reduce the size of the distribution. Rather than having to store all $512 \times 512 = 262144$ neighborhoods, just one instance from each rotationally invariant set of neighborhoods needs to be stored. This reduces the size of the distribution by roughly a factor of eight. (Actually, because some mappings have fewer than 8 rotationally invariant counter parts, the size of the resulting table would be 32920 rather than 32768.)

This technique could be used as a way to keep the size of the distribution under control when using neighborhoods larger than 3×3. Also, with neighborhoods larger than 3×3, because the level of discretization is different, it is possible to also consider rotations of less than 45° , and thus get even greater savings.

The issue here is that the size of the distribution is 2^{2n^2} for an $n \times n$ neighborhood. So, even for a 4×4 neighborhood, there are then $2^{32} \approx 4$ billion possible neighborhood labelings. Reducing this huge number by a factor of 8, or a factor of 12 (which can be achieved with the level of discretization inherent in a 4×4 block), represents a significant savings in memory.

Chapter 8

Edge detection

8.1 Introduction

This chapter shows that the DNMRF method works in its own right as an edge detector. This is demonstrated on some artificially generated binary images, and also on some binarized real machine-vision type images. The results are comparable with other edge detectors images of the same type. This demonstrates the applicability of the DNMRF method to the wide set of domains in which binary images are used.

8.2 Training

Two separate classes of DNMRFs were generated, both for edge detection. One was trained using the same binary images of solid black circles and their true edges, as described in chapter 7, and the other was trained on images of narrow wedges, and their true edges. Each training set consisted of 256 pairs of images, all of size 64×64. Examples of the training images are shown in figure 8.1.

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

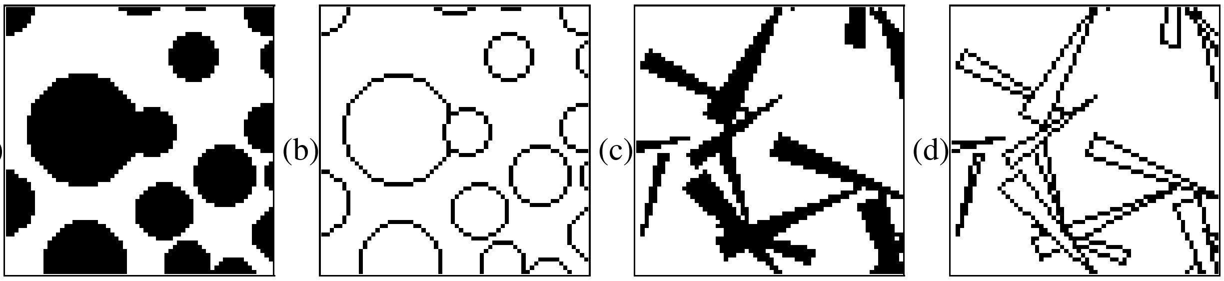


Figure 8.1: Sample training images: (a) A circle image; (b) True edges of the circle image; (c) A wedge image; (d) True edges of the wedge image

DNMRFs were also trained on corrupted versions of this data—with 5% independent random binary noise added to the object files, and with 10% independent random binary noise added.

Another set of DNMRF distributions were created using the rotationally invariant method described in chapter 7.

Finally, the Hancock-Kittler smoothing technique (as described in chapters 3 and 4) was applied to each of these distributions, yielding the smoothed, usable distributions. The HK smoothing level used was 5% for the circle distributions and 10% for the wedge distributions.

8.3 Measures of correctness

As a measure of goodness of the results, the two metrics discussed previously were used, (bits of error, and Pratt's figure of merit) as well as a pair of new measures: the number of Type I errors (missed edges) and Type II errors (false positives) each expressed as a percentage of the number of edge-pixels in the true image. Note that these new measures correctly disregard the boundary pixels which are outside the range of the edge detector, whereas the two older measures, as implemented, both count the boundary. Thus the bits of error and Pratt's figure of merit both tend to report results worse than is actually the case—

they compare bits outside the region of the image on which the edge detectors operate.

8.4 Results

The generated test images were of the same form as the training images. A set of 64 circle images and 64 wedge images were generated. These were used as input to the gradient descent algorithm, using the distributions described above. The corresponding true edges were also generated, and used to verify the results.

As well as the DNMRF ICM method, with and without the rotational invariance modification, three other edge detectors were tested on these binary images: the thresholded 4-operator difference of boxes, the thresholded Sobel operator [Sobel, 1970; Duda and Hart, 1973], and the Canny edge operator[1] [Canny, 1986]. (We also tried the Roberts edge detector [Roberts, 1965; Rosenfeld, 1981], but it proved effectively useless in the presence of noise.)

One parameter has to be chosen for the Canny operator: the value of σ, the variance of the Gaussian operators used. Various values were tried, starting with the default value (in this implementation of Canny) of 2.0. For the circle images, a value of 2.5 was found to generate the best results. For the wedge images, $\sigma = 2.5$ generates images in which the edges were clearly too widely spaced. However, when lower values are used, the noise causes too many spurious edges. For example, for one particular image with 5% noise added (the image shown in figure 8.4), the Type I error was 90.11% at $\sigma = 2.5$, and went down to 54.94% at $\sigma = 0.9$. However, at $\sigma = 2.5$ the Type II error is 85.75%, but is 189.90% when using $\sigma = 0.9$. So, in the table below, the value of $\sigma = 2.5$ was chosen for the circle and the wedge images.

[1] The Canny edge operator code used was written by Paul Kloschinsky of the University of British Columbia.

The complete table of results for the two sets of 64 images with 5% and with 10% random binary noise added, are shown in table 8.1.

Some sample images and their resulting edges are shown in figures 8.2–8.5.

Overall, the DNMRF method consistently works better than any of the other methods tested, using any of the measures of correctness.

Note that the true edges, against which all these metrics are measuring, have "thin" edges, even on diagonal line segments, and that this is what the DNMRF was trained with. However, some of the other edge detectors are designed to generate slightly thicker edges—one pixel thick on the horizontal and vertical portions, but two pixels thick on the diagonal sections of edge. This might explain why the DNMRF method is doing better than all the other edge detectors—it is trained to generate exactly the type of edges that are contained in the true edge image.

The difference of boxes and Sobel operators tend to be effected by noise more than the DNMRF operator. Canny's operator, because it has a fairly large set of support, tends to connect objects that are close to each other. See, for example, the lower-right portion of the image in figure 8.4 (f). Canny's operator also has a very large region in which it cannot find edges—an 8 pixel wide border at the top and bottom on the image (when $\sigma = 2.5$). While this region can be discounted when calculating error metrics, it is still a disadvantage of the method.

8.5 Real Images

The method was also tested on real images generated by taking photographs of various sized nuts, on a light background, as might be found in a typical machine vision environment. These photographs were then scanned, at 96 dots per inch, and the resulting images were

Circles with 5% noise				
Method	% Type I Errors	% Type II Errors	Bits different	Pratt's FOM
DNMRF	13.29	11.46	145	0.86
Rot. Inv. DNMRF	12.42	10.63	127	0.88
Difference of boxes	51.31	91.78	560	0.62
Sobel	16.01	107.50	588	0.70
Canny	31.60	62.33	382	0.65
Circles with 10% noise				
Method	% Type I Errors	% Type II Errors	Bits different	Pratt's FOM
DNMRF	31.89	16.45	211	0.73
Rot. Inv. DNMRF	36.79	11.08	210	0.67
Difference of boxes	49.50	161.60	811	0.49
Sobel	26.61	125.40	593	0.62
Canny	35.03	71.88	419	0.63
Wedges with 5% noise				
Method	% Type I Errors	% Type II Errors	Bits different	Pratt's FOM
DNMRF	20.00	16.74	195	0.87
Rot. Inv. DNMRF	19.86	22.35	221	0.86
Difference of boxes	52.43	71.31	611	0.64
Sobel	31.75	90.30	603	0.69
Canny	84.68	109.20	824	0.34
Wedges with 10% noise				
Method	% Type I Errors	% Type II Errors	Bits different	Pratt's FOM
DNMRF	28.19	22.51	265	0.83
Rot. Inv. DNMRF	29.31	22.03	270	0.82
Difference of boxes	49.65	127.00	845	0.51
Sobel	40.86	103.90	704	0.61
Canny	83.72	114.90	837	0.34

Table 8.1: Numerical results. Results shown are the mean values over 64 images. (Note that the Bits different, and Pratt's FOM entries are particularly meaningless for Canny, because Canny has an 8 pixel wide border on the top and bottom of the images.) The test images were all of size 64×64, making 4096 total pixels per image.

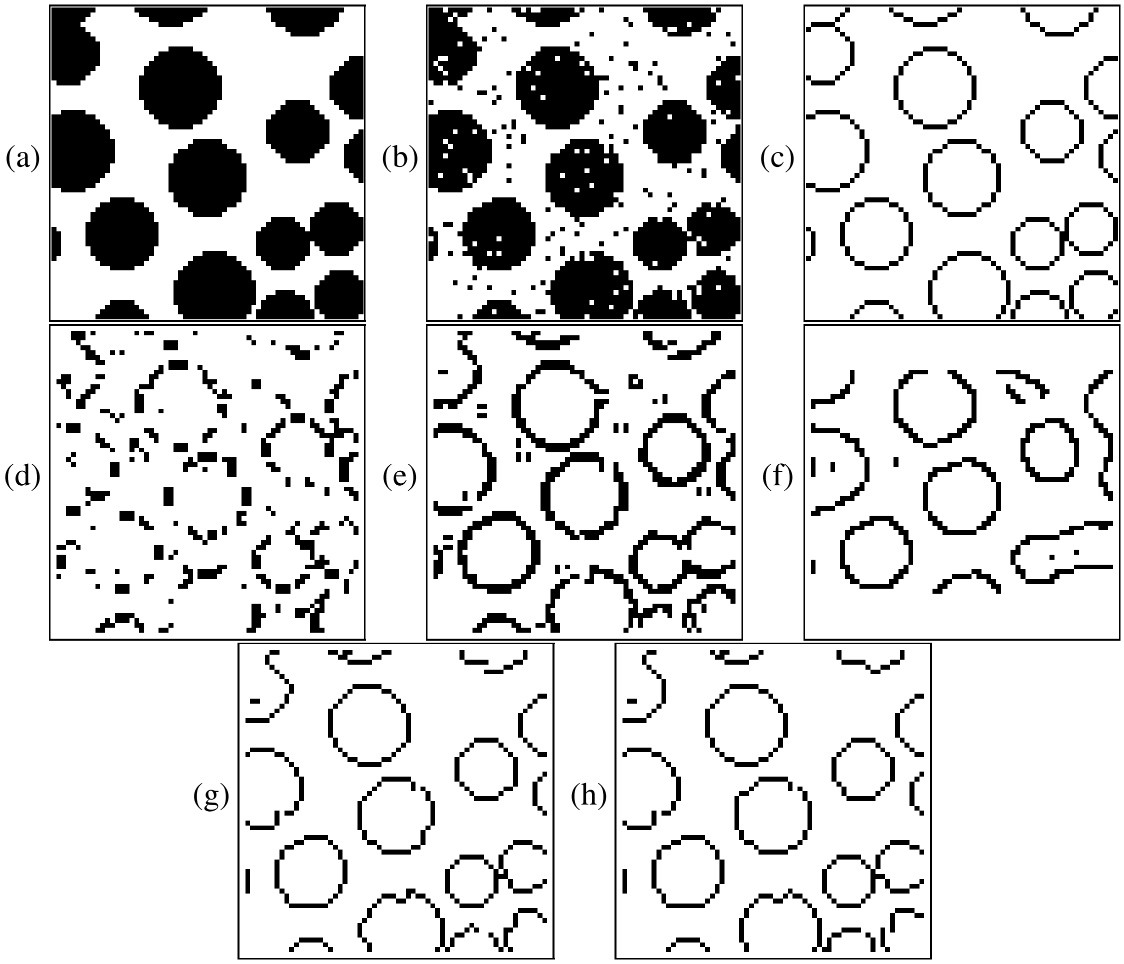


Figure 8.2: Sample runs on a circle image. (a) Original Image; (b) Image with 5% noise added; (c) True edges; (d) Output of thresholded difference of boxes operator; (e) Output of thresholded Sobel operator; (f) Output of Canny operator; (g) Output of ICM algorithm using the DNMRF; (h) Output of ICM algorithm using the rotationally invariant DNMRF.

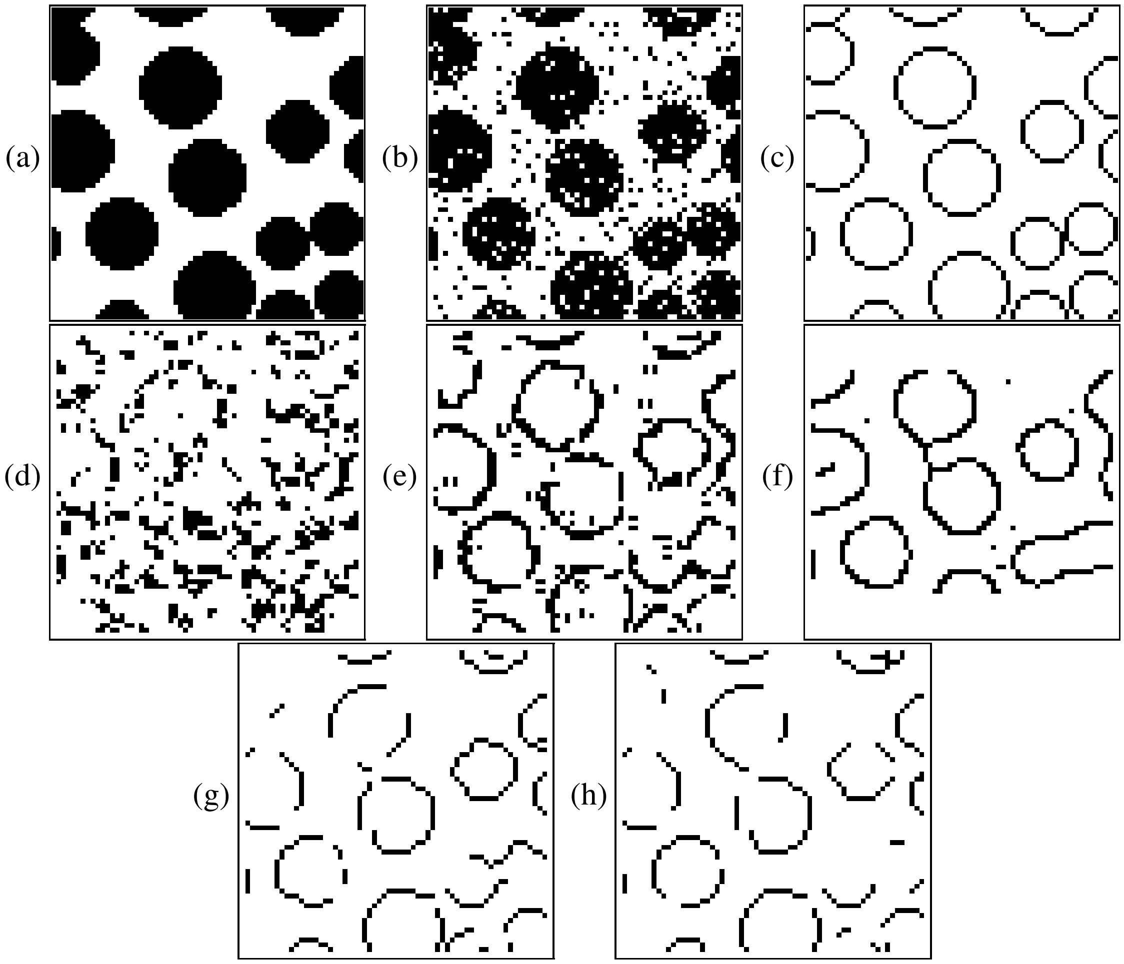


Figure 8.3: Sample runs on a circle image. (a) Original Image; (b) Image with 10% noise added; (c) True edges; (d) Output of thresholded difference of boxes operator; (e) Output of thresholded Sobel operator; (f) Output of Canny operator; (g) Output of ICM algorithm using the DNMRF; (h) Output of ICM algorithm using the rotationally invariant DNMRF.

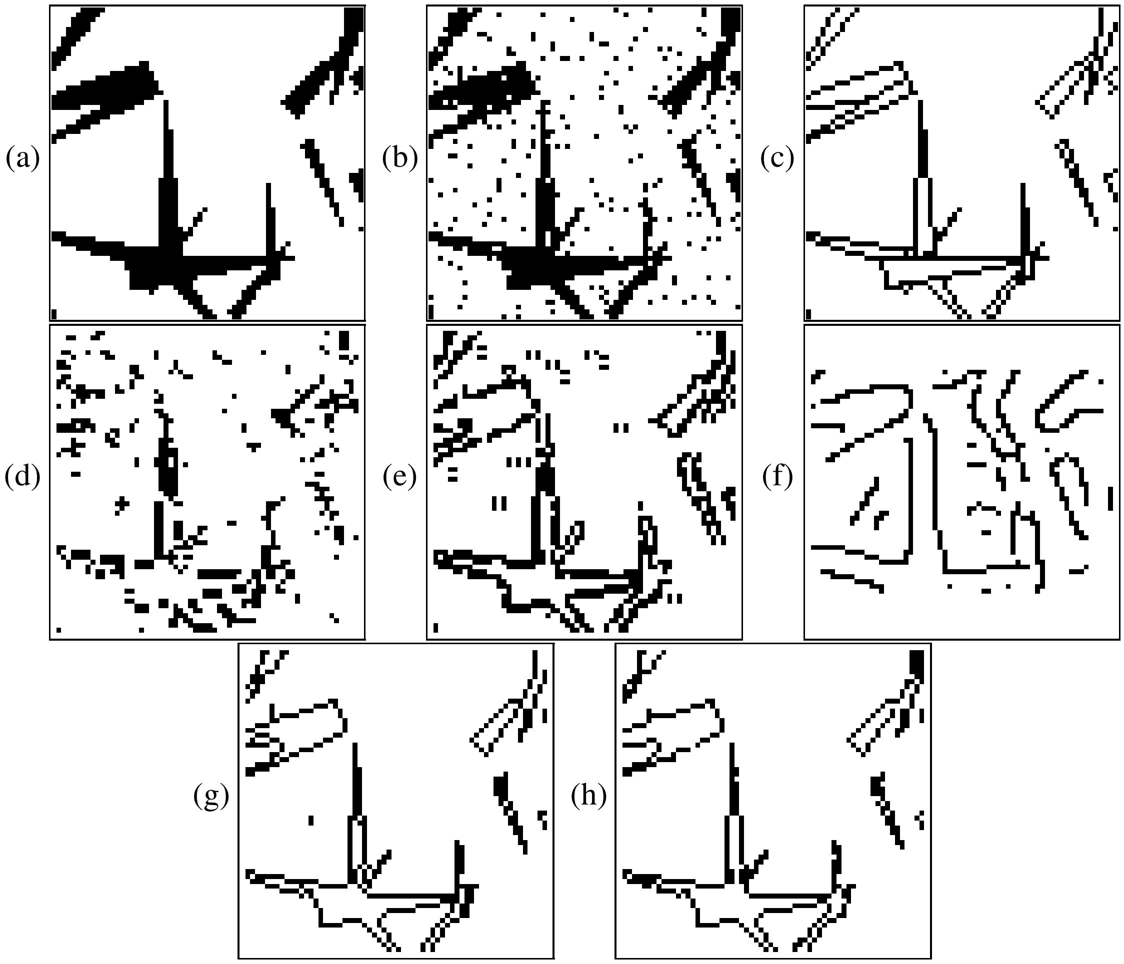


Figure 8.4: Sample runs on a wedge image. (a) Original Image; (b) Image with 5% noise added; (c) True edges; (d) Output of thresholded difference of boxes operator; (e) Output of thresholded Sobel operator; (f) Output of Canny operator; (g) Output of ICM algorithm using the DNMRF; (h) Output of ICM algorithm using the rotationally invariant DNMRF.

(a) (b) (c) (d) (e) (f) (g) (h)

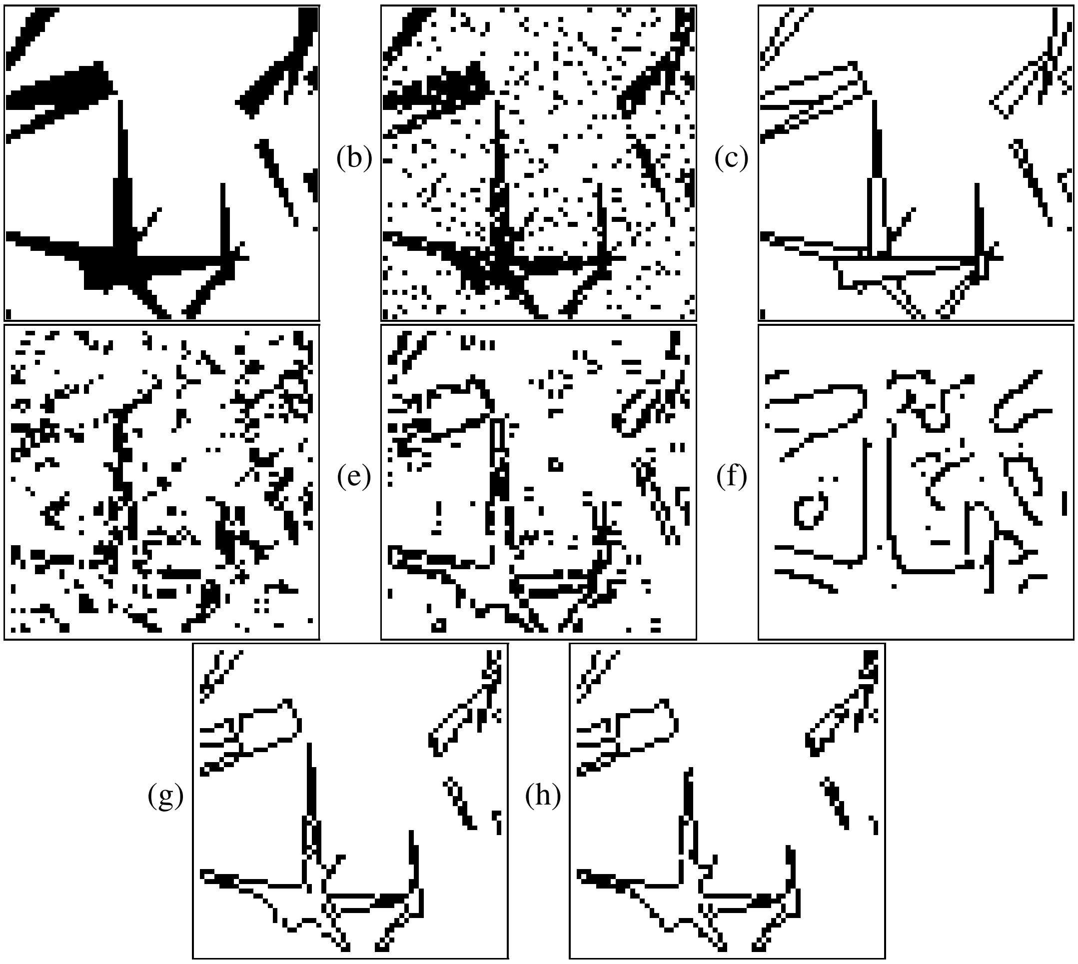


Figure 8.5: Sample runs on a wedge image. (a) Original Image; (b) Image with 10% noise added; (c) True edges; (d) Output of thresholded difference of boxes operator; (e) Output of thresholded Sobel operator; (f) Output of Canny operator; (g) Output of ICM algorithm using the DNMRF; (h) Output of ICM algorithm using the rotationally invariant DNMRF.

thresholded to yield binary images. This process tended to generate clean binary images, with little random noise.

The various edge detectors discussed in section 8.4 were then run on these images. Some were run on the scanned gray scale images, and some on the binary images, and some were run on both.

The question with the DNMRF algorithm is what training images should be used. Since the nuts are almost round, it seemed reasonable to use the DNMRF distribution that was generated from sampling the circle images as described in section 8.2.

The results of one particular image are shown in figure 8.6. Since these are real images, there is true image against which to run measures of goodness, and thus generate numbers for comparison. However, visual inspection of these images shows that the DNMRF ICM algorithm yields results clearly comparable with the other edge detectors.

One of the strengths of this method, compared to many other edge detectors, can be seen by looking carefully at the two adjacent nuts in the top left of figure 8.6. You'll notice that most of the edge detectors were not able to correctly resolve the edges into two distinct objects, for example subimages (e), (f) and (g). In (d), which is the output of Canny, the two edges are separate, but strange artifacts are introduced. Contrast to those subimages (h) and (i), generated by the DNMRF method, which do not show such problems.

The reason for this is that many of the edge detection operators do some sort of averaging within their set of support. They tend to try to find one edge (or maybe one corner) within that region. So when two edges appear close together, it causes problems. Whereas in the DNMRF method, no averaging is done; the specific pattern of pixels is what is relevant. So, unlike the Canny operator, the DNMRF method should not generate more of this sort of error as the neighborhood size increases.

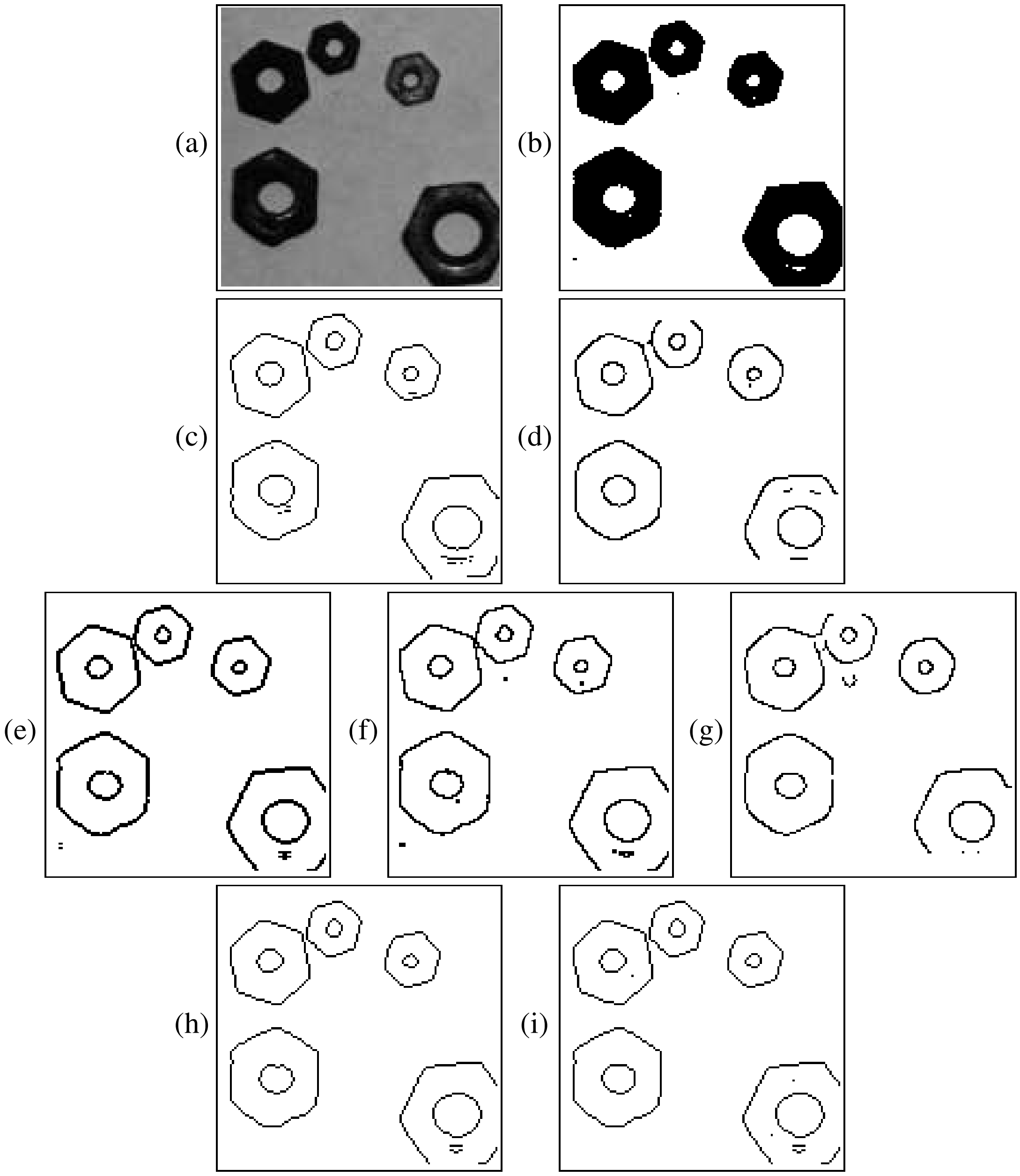


Figure 8.6: Sample runs on a real image. (a) Original scanned gray scale image; (b) Binary image, generated by thresholding (a); (c) Output of thresholded thinned Sobel operator applied to image (a); (d) Output of Canny's operator ($\sigma = 2.5$) applied to image (a); (e) Output of thresholded Sobel operator applied to image (b); (f) Output of thresholded Roberts operator applied to image (b); (g) Output of Canny's operator ($\sigma = 2.5$) applied to image (b); (h) Output of ICM algorithm using the DNMRF trained on the circles images with 5% noise added, applied to image (b); (i) Output of ICM algorithm using the rotationally invariant DNMRF trained on the circles images with 5% noise added, applied to image (b).

8.6 Conclusions

The DNMRF method, when trained on an appropriate set of images, can hold its own as an edge detector. On artificial images, it generated numerically better results than a collection of other edge detectors, including Canny's edge operator. On real machine-vision type images, it also performed at least as well as the other edge detectors, especially if a one-pixel wide edge is needed.

However, one should bear in mind that the DNMRF method is more than just an edge detector—it is a more general method, that can solve a variety of problems. Edge detection is just one of the possible problem domains to which is can be successfully applied.

Chapter 9

Sampling generates different distributions

9.1 Introduction

This chapter demonstrates that the choice of training set indeed influences the operation that the method will perform. An appropriate training set can cause the method to do a variety of tasks, from edge detection to image reconstruction.

9.2 Distribution varies between image classes

One of the primary motivations behind this thesis is that an image processing system can be generated for a particular domain, simply by sampling a sufficiently large number of images (and their "solution") from that domain. This section demonstrates that the DNMRF distributions sampled from different domains are indeed different.

To test this, two sets of images were generated—the same image sets as described in the training section of chapter 8. One set consisted of 256 pairs of images of solid black circles and their true edges, and the other set consisted the same number of pairs of images of narrow wedges, and their true edges. Each set contained 256 pairs of images, all of size 64×64. Examples of these images are shown in figure 8.1.

In order to demonstrate that two distributions are effectively different, some measure of similarity is needed. One obvious metric is the following: consider each of the 2^{18} values in the DNMRF distribution to be one of the components of a vector in 2^{18}-space, and measure the Euclidean distance between the two vectors. This metric does not turn out to be very useful in the context of the algorithms for which these distributions are being used. For example, the ICM algorithm only ever compares two values in the distribution table: within the row of the table determined by the given, noisy neighborhood, it compares the probability values of the two neighborhoods which differ by only the center pixel. This means that only the relative magnitudes of these two entries matter. Therefore, two distributions that have completely different values at every point, and therefore are very far apart by the metric described above, will both operate identically in the ICM algorithm, if the relative magnitudes of all pairs of neighborhoods in each row are preserved.

Therefore, another measure is needed, and one immediately suggests itself from the previous discussion: count the number of such switches in the magnitude of pairs of pixels, where a pair is defined as two neighborhoods which differ only in the labeling of the center pixel of the true/clean image. This measure was used.

One problem with this new measure is that it does not take into account how often any particular neighborhood occurs. The measure would be more meaningful if it weighted more strongly those neighborhoods that occur more frequently. So a slight modification was made, which weights these swaps by the probability of the neighborhood entries in question.

In order to provide a reasonable test, subsets of the image sets were generated. Two sets of size 64 were selected from the complete set of 256 pairs images, for each of the circle and wedge images. This therefore yields four sets of images, which are labeled circle1, circle2,

	circle1		circle2		wedge1		wedge2	
	#	weight	#	weight	#	weight	#	weight
circle1	–	–	35	.00343	466	.00125	476	.00148
circle2	35	.00354	–	–	467	.00115	478	.00143
wedge1	466	.0908	467	.0900	–	–	149	.00343
wedge2	476	.0914	478	.0910	149	.00354	–	–

Table 9.1:

wedge1 and wedge2. A DNMRF was trained on each of these six sets.

Table 9.1 contains the complete results of comparing all the possible combinations of the four smaller sets. Note that while the table is symmetric with respect to the number of switches, it is not symmetric with respect to the weight of the switched probabilities. This is because the weights are calculated from the probability values in the distribution listed on the y-axis.

These results confirm the hypothesis. The inter-class differences are much larger than the intra-class differences. This leads one to believe that training on similar sets of images yields similar distributions, and that training on different sets of images yields distributions that are different.

9.3 Conclusions

The choice of training set determines what problem the DNMRF method will solve. And even within a particular domain, such as edge detection, different training sets generate different DNMRF distributions, which causes the method to be fine tuned for the domain from which the training set was drawn.

Chapter 10

Directions for future research

Even though this thesis demonstrated some new techniques, and generated some useful results, it is far from a finished work. There are many more tests that can be performed, and many more studies that can be done. For example, experiments can be done using larger training sets, as indicated in chapter 5.

More testing with edge detection on real images can also be done. In particular, in order to get more objective results it would be useful to obtain a set of images and their true edges, in order to have something against which to compare the generated edge maps. These true edge could be generated by hand, or could be generated by modeling the exact locations of the objects in the scene, given their shapes and positions.

One large and obvious improvement to the DNMRF method would be to apply it directly to gray scale images. It seems that this should be a simple extension to the method, but it turns out to be more complicated.

The issue is that it is unrealistic to build an MRF which takes into account all possible gray levels, because of the huge number of possible neighborhood labelings—the 2^{18} in the binary case becomes $256^{18} \approx 10^{44}$ when there are 8 bits per pixel. However, this problem should be reduced by the fact that the relationships in the local structure are not dependent on the exact gray scale values of the pixels involved, but rather on the relative intensities of the pixels.

Some work was done on solving this problem, but no completely satisfactory solution was discovered. The objective is to represent the important features of the gray scale 3×3 neighborhood, without having to store all 9 intensity values. Various binarization methods were experimented with or considered, such as an idea of Johnson [Johnson, 1991] to use the brightness relative to the center pixel of each neighborhood as a binarization, or using the output of a set of feature detectors on a region of the image. None of these worked.

The most successful method that was tried used clustering. The concept was to find a clustering algorithm that could be trained on the gray neighborhoods from any given domain, and would classify those gray neighborhoods into clusters. This would yield a cluster number for each neighborhood, which would then be used as the “binary neighborhood”.

A clustering algorithm with the following properties was thus required:

1. It should be unsupervised, in that it should not require any user input to determine which input vector corresponds to which cluster.
2. After training, it should be able to take a previously unseen input vector, and classify it.

There are many clustering algorithms for which (1) holds. However, once most of them have been run on a given set of data, and have classified that data into clusters, they do not provide a means of classifying input vectors not already seen.

One method that does not have this problem is Kohonen’s Self Organizing Feature Maps [Kohonen, 1982; Kohonen, 1984]. It is a neural net based method, that is trained on a set of input vectors. Once trained, it will classify any input vector, regardless of whether it had previously been seen by the clustering algorithm.

Some experimentation was done using Kohonen's method to perform the clustering, and using the clusters which were output as the input to the DNMRF method—both during training, and during the ICM algorithm.

More work on this is needed before any firm results will be available.

Chapter 11

Conclusion

This thesis started by improving MRF distributions which were sampled from a set of training images. They were improved by smoothing using the techniques of Hancock and Kittler. While the results were promising, classic single field MRFs proved to be inadequate for many tasks, because they could not capture complex noise models, which are the norm in many domains. So double neighborhood MRFs were introduced. They proved to be more flexible, and powerful enough to represent complex noise models.

But in both these methods, training sets are needed. It was shown that for the single field case, large training sets are not required, but when DNMRFs are used, a large set of training images is needed. This can be somewhat reduced by taking advantage of rotational invariance in those domains in which it is applicable. It is important to train on the correct type of images, so that the DNMRF distribution will accurately reflect the type of images being operated on.

One situation in which DNMRFs are useful is when dealing with edge detectors. The errors in the output of an edge detector are correlated, and the DNMRF can incorporate such noise models, and thus correct for that type of error. The DNMRF method was demonstrated improving the output of the Sobel edge detector, in several different domains.

The DNMRF contains enough local information about the images to be able to act as an edge detector itself. It was shown that the DNMRF method, trained as an edge detector, can perform as well as some state of the art edge detectors, in certain domains.

In conclusion, the power of the DNMRF method has been demonstrated. It has theoretical roots in the MRF literature, and is practical enough to solve a variety of problems. Some directions in which further study is required have been suggested.

This thesis has shown that it is possible to build a low-level vision system that is reasonably generic, and can be trained to perform one of a variety of tasks. The concept of a double neighborhood MRF was introduced and shown to be usable and useful. The DNMRF method was employed with success for image reconstruction, for edge detector enhancement, and for edge detection.

Bibliography

[Ballard and Brown, 1982] Dana H. Ballard and Christopher M. Brown. *Computer Vision*. Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[Banerjee and Rosenfeld, 1992] Saibal Banerjee and Azriel Rosenfeld. MAP estimation of context free grammars. *Pattern Recognition Letters*, 13:95–101, Feb 1992.

[Basu *et al.*, 1989] Sujit Basu, M. Mohan, and Vijay K. Agarwal. Estimation of cloud microstructural parameters from satellite imagery: A Markov random field approach. *Mausam*, 40(3):243–246, July 1989.

[Bedini and Tonazzini, 1992] Luigi Bedini and Anna Tonazzini. Image restoration preserving discontinuities: the Bayesian approach and neural networks. *Image and Vision Computing*, 10(2):108–118, March 1992.

[Besag, 1974] J.E. Besag. Spatial interaction and the statistical analysis of lattice systems (with discussion). *Journal Royal Statistical Society, Series B*, 36:192–236, 1974.

[Besag, 1986] J.E. Besag. On the statistical analysis of dirty pictures. *Journal Royal Statistical Society, Series B*, 48:259–302, 1986.

[Blake, 1989] Andrew Blake. Comparison of the efficiency of deterministic and stochastic algorithms for visual reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(1):2–12, January 1989.

[Canny, 1986] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, November 1986.

[Chellappa and Jain, 1993] R. Chellappa and A. Jain, editors. *Markov Random Fields—Theory and Application*. Academic Press, 1993.

[Chou, 1987] Paul Chou. Multi-modal segmentation using Markov random fields. In *International Joint Conference on Artificial Intelligence*, July 1987.

[Cross and Jain, 1983] G.R. Cross and A.K. Jain. Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(1):25–39, 1983.

[Derin and Elliott, 1987] Haluk Derin and Howard Elliott. Modeling and segmentation of noisy and textured images using Gibbs random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):39–55, January 1987.

[Dobruschin, 1968] P. L. Dobruschin. The description of a random field by means of conditional probabilities and conditions of its regularity. *Theory of Probability and its Applications*, 13:197–224, 1968.

[Duda and Hart, 1973] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, Inc., New York, London, Sydney, Toronto, 1973.

[Elfadel and Picard, 1994] Ibrahim M. Elfadel and Rosalind W. Picard. Gibbs random fields, cooccurrences, and texture modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):24–37, January 1994.

[Faugeras and Berthod, 1981] Olivier D. Faugeras and Marc Berthod. Improving constancy and reducing ambiguity in stochastic labeling: An optimization approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-3(4):412–424, 1981.

[Faugeras and Price, 1981] Olivier D. Faugeras and Keith Price. Semantic description of aerial images using stochastic labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-3(6):633–642, 1981.

[Fekete *et al.*, 1981] G. Fekete, J.-O. Eklundh, and A. Rosenfeld. Relaxation: Evaluation and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-3(4):459–469, 1981.

[Geiger and Girosi, 1991] D. Geiger and F. Girosi. Parallel and deterministic algorithms from MRF's: Surface reconstructions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):401–413, May 1991.

[Geman and Geman, 1984] Stuart Geman and Donald Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, November 1984.

[Geman *et al.*, 1990] D. Geman, S. Geman, C. Graffigne, and P. Dong. Boundary detection by constrained optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):609–628, July 1990.

[Gimel'farb and Zalesny, 1991] G.L. Gimel'farb and A.V. Zalesny. Low-level Bayesian segmentation of piecewise-homogeneous noisy and textured images. *International Journal of Imaging Systems and Technology*, 3:227–243, 1991.

[Gimel'farb and Zalesny, 1993] G.L. Gimel'farb and A.V. Zalesny. Probabilistic models of digital region maps based on Markov random fields with short- and long-range interaction. *Pattern Recognition Letters*, 14(10):789–797, 1993.

[Goutsias, 1989] J. K. Goutsias. Mutually compatible Gibbs random fields. *IEEE Transactions on Information Theory*, 35(6):1233–1249, November 1989.

[Goutsias, 1991] J. K. Goutsias. A theoretical analysis of Monte Carlo algorithms for the simulation of Gibbs random field images. *IEEE Transactions on Information Theory*, IT-37(6):1618–1628, November 1991.

[Gray *et al.*, 1992] A.J Gray, J.W. Kay, and D.M. Titterington. On the estimation of noisy binary Markov random fields. *Pattern Recognition*, 25(7):749–768, 1992.

[Hancock and Kittler, 1990a] E.R. Hancock and J. Kittler. Discrete relaxation. *Pattern Recognition*, 23(7):711–733, 1990.

[Hancock and Kittler, 1990b] E.R. Hancock and J. Kittler. Edge-labeling using dictionary-based relaxation. *Pattern Analysis and Machine Intelligence*, 12(2):165–181, February 1990.

[Hancock and Kittler, 1990c] E.R. Hancock and J. Kittler. A label error process for discrete relaxation. In *Proceedings of the 10th international conference on pattern recognition*, pages 523–527. IEEE, June 1990.

[Haralick and Shapiro, 1992] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision*, volume I. Addison-Wesley, Reading, MA, 1992.

[Hassner and Sklansky, 1980] M. Hassner and J. Sklansky. The use of Markov random fields as models of texture. *Computer Graphics and Image Processing*, 12:357–370, 1980.

[Hebert and Malagre, 1994] Thomas J. Hebert and Denis Malagre. Optimization approach to edge detection. *Journal of the Optical Society of America, Series A*, 11(1):80–88, January 1994.

[Hummel and Zucker, 1980] Robert A. Hummel and Steven W. Zucker. On the foundations of relaxation labeling. Technical Report TR-80-7, Computer Vision and Graphics Lab, Department of Electrical Engineering, McGill University, July 1980.

[Hummel, March 1983] Robert A. Hummel. A design method for relaxation labeling applications. Technical report, New York University Courant Institute of Mathematical Sciences, March, 1983.

[Jeng and Woods, 1990] F.C. Jeng and J.W. Woods. Simulated annealing in compound Gaussian random fields. *IEEE Transactions on Information Theory*, 36(1):94–107, January 1990.

[Johnson *et al.*, 1991] V. E. Johnson, W. H. Wong, X. Hu, and C. T. Chen. Image restoration using Gibbs priors: Boundary modeling, treatment of blurring and selection of hyperparameter. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):413–425, May 1991.

[Johnson, 1991] J. Johnson. Personal Communication, July 1991.

[Karssemeijer, 1990] N. Karssemeijer. A relaxation method for image segmentation using a spatially dependent stochastic model. *Pattern Recognition Letters*, 11:13–23, 1990.

[Kim and Yang, 1991] Il Young Kim and Hyun Seung Yang. Markov random field based image labeling with parameter estimation by error backpropagation. *IEICE Transactions on Information and Systems*, E74(10):3513–3521, October 1991.

[Kirsch, 1971] R.A. Kirsch. Computer determination of the constituent structure of biological images. *Computers and Biomedical Research*, 4:315–328, 1971.

[Kittler and Hancock, 1987] J. Kittler and E. R. Hancock. Contextual decision rule for region analysis. *Image and Vision Computing*, 5(2):145–153, May 1987.

[Kittler and Illingworth, 1985] J. Kittler and J. Illingworth. Relaxation labelling algorithms—a review. *Image and Vision Computing*, 3:206–216, 1985.

[Kittler, 1986] J. Kittler. Compatibility and support functions in probabilistic relaxation. In *Proceedings of the International Conference on Pattern Recognition*, pages 186–189, 1986.

[Kohonen, 1982] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[Kohonen, 1984] Teuvo Kohonen. *Self-Organisation and Associative Memory*. Springer Verlag, 1984.

[Lakshmanan and Derin, 1989] S. Lakshmanan and H. Derin. Simultaneous parameter estimation and segmentation of Gibbs random fields using simulated annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(8):799–813, August 1989.

[Lugosi, 1992] Gabor Lugosi. Learning with an unreliable teacher. *Pattern Recognition*, 25(1):79–85, 1992.

[Manjunath and Chellappa, 1991] B. S. Manjunath and R. Chellappa. Unsupervised texture segmentation using Markov random field models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):478–482, May 1991.

[Marr and Hildreth, 1980] D. Marr and E. Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London*, B-207:187–217, 1980.

[Marroquin *et al.*, 1985] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. In Lee S. Bauman, editor, *Proceedings: Image Understanding Workshop*, pages 293–309. Science Applications International Corporation, December 1985.

[Marroquin *et al.*, 1987] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. *Journal American Statistical Association*, 82:76–89, 1987.

[Marroquin, 1985] Jose Luis Marroquin. Probabilistic solution of inverse problems. Technical Report TR 860, MIT Artificial Intelligence Laboratory, September 1985.

[Metropolis *et al.*, 1953] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, June 1953.

[Milun and Sher, 1991] D. Milun and D. B. Sher. Improving sampled probability distributions for Markov random fields. Technical Report 91-08, S.U.N.Y. at Buffalo, Computer Science Department, July 1991.

[Milun and Sher, 1992a] D. Milun and D.B. Sher. Learning structural and corruption information from samples for Markov random field edge detection enhancement. In *The SPIE 1992 International Symposium on Optical Applied Science and Engineering: Neural and Stochastic Methods in Image and Signal Processing*, San Diego, 1992.

[Milun and Sher, 1992b] Davin Milun and David Sher. Learning structural and corruption information from samples for Markov random field binary image reconstruction. In *11th ICPR: Conference on Image, Speech and Signal Analysis*, volume 3, pages 513–516, The Hague, The Netherlands, September 1992.

[Milun and Sher, 1993a] D. Milun and D. B. Sher. Training double neighborhood Markov random fields by sampling—How much data is required? In *Vision Interface 93*, Toronto, Canada, May 1993.

[Milun and Sher, 1993b] D. Milun and D.B. Sher. Training Markov random fields by sampling—How much data is required? In *The SPIE 1993 International Symposium* *on Optical Engineering and Photonics in Aerospace and Remote Sensing: Adaptive and Learning Systems II*, Orlando, April 1993.

[Milun and Sher, 1993c] Davin Milun and David Sher. Improving sampled probability distributions for Markov random fields. *Pattern Recognition Letters*, 14(10):781–788, October 1993.

[Pratt, 1978] W. K. Pratt. *Digital Image Processing*. Wiley-Interscience, New York, 1978.

[Prewitt, 1970] J. Prewitt. Object enhancement and extraction. In *Picture Processing and Psychopictorics*, pages 75–149. Academic Press, New York, 1970.

[Price, 1985] K.E. Price. Relaxation matching techniques—a comparison. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7:617–623, 1985.

[Roberts, 1965] L.G. Roberts. Machine perception of three-dimensional solids. In J.T.T Tippett et al, editor, *Optical and electro-optical information processing*, pages 159–197. MIT Press, 1965.

[Rosenfeld and Smith, 1981] Azriel Rosenfeld and Russel C. Smith. Thresholding using relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-3(5):598–606, 1981.

[Rosenfeld *et al.*, 1975] Azriel Rosenfeld, Robert A. Hummel, and Steven W. Zucker. Scene labelling by relaxation operations. Technical Report TR-379, University of Maryland, Computer Science Department, May 1975.

[Rosenfeld, 1981] Azriel Rosenfeld. The max Roberts operator is a Hueckel-type edge detector. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-3(1):101–103, 1981.

[Rutkowski *et al.*, 1981] W. S. Rutkowski, S. Peleg, and A. Rosenfeld. Shape segmentation using relaxation. *IEEE Transaction of Pattern Analysis and Machine Intelligence*, 3(4):368–375, July 1981.

[Saraga and Woollons, 1968] P. Saraga and D. J. Woollons. The design of operators for pattern processing. In Teddington, editor, *Proceedings of the IEEE NPL Conference on Pattern Recognition*, pages 106–116, 1968.

[Sher *et al.*, 1993] D. B. Sher, C. E. Wafford, and D. Milun. Relating Gibbs distributions to empirically derived marginal distributions for image analysis. Technical Report 93-39, S.U.N.Y. at Buffalo, Computer Science Department, December 1993.

[Sher, 1991a] David Sher. Deriving the distribution of edge neighborhoods from a library of occluding objects. In *Bar-Ilan Symposium on the Foundations of Artificial Intelligence*, June 1991.

[Sher, 1991b] David Sher. Minimizing the cost of errors with a Markov random field. *Pattern Recognition Letters*, 12(2):85–89, February 1991.

[Sher, 1991c] D.B. Sher. A technique for deriving the distribution of edge neighborhoods from a library of occluding objects. In V. Cantoni, M. Ferretti, S. Levialdi, R. Negrini, and R. Stefanelli, editors, *6th International Conference on Image Analysis and Processing : Progress in Image Analysis and Processing II*, pages 422–428, September 1991.

[Sobel, 1970] I. E. Sobel. *Camera Models and Machine Perception*. PhD thesis, Electrical Engineering Department, Stanford University, 1970.

[Swain *et al.*, 1990] Michael J. Swain, Lambert E. Wixson, and Paul B. Chou. Efficient parallel estimation for Markov random fields. *Uncertainty in Artificial Intelligence*, 5:407–419, 1990.

[Swain, 1990] Michael J. Swain. Parameter learning for Markov random fields with highest confidence first estimation. Technical Report TR350, Department of Computer Science, University of Rochester, August 1990.

[Szeliski, 1989] R. Szeliski. *Bayesian Modeling of Uncertainty in Low-level Vision*. Kluwer Academic Publishers, 1989.

[Tan *et al.*, 1992] H. L. Tan, S. B. Gelfand, and E. J. Delp. A cost minimization approach to edge detection using simulated annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(1):3–19, January 1992.

[Veijanen, 1991] A. Veijanen. A simulation-based estimator for hidden Markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):825–829, August 1991.

[Wallace, 1991] Gregory K. Wallace. The JPEG still picture compression standard. *Communications of the ACM*, 34(4):30–44, April 1991.

[Woods *et al.*, 1987] J.W. Woods, S. Dravida, and R. Mediavilla. Image estimation using doubly stochastic Gaussian random field models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:245–253, 1987.

[Zucker *et al.*, 1975] Steven W. Zucker, Robert A. Hummel, and Azriel Rosenfeld. Applications of relaxation labelling, 1: Line and curve enhancement. Technical Report TR 419, University of Maryland, October 1975.