

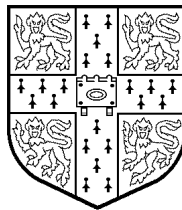
---

# IMAGE SEQUENCE RESTORATION USING GIBBS DISTRIBUTIONS

---

by  
**Robin David Morris**

A thesis submitted to the  
University of Cambridge for the degree  
of Doctor of Philosophy



**To my Parents  
and the memory of  
my Grandparents**

## **Declaration**

---

The research described in this thesis was carried out by the author between October 1991 and May 1995. Except as indicated in the text, the contents are entirely original and are not the result of work done in collaboration. No part of this thesis has been submitted to any other University. This thesis contains not more than 65,000 words.

Robin D. Morris

## **Acknowledgements**

---

I would like to thank my supervisor, Dr. Bill Fitzgerald for his support and advice during the past three and a half years. He has been a constant source of ideas and encouragement, and this thesis owes much to him.

I would like to thank all my friends in the lab who have made it such a pleasant place to work. Dr. Anil Kokaram and Dr. Joan Lasenby deserve special mentions. Dr. Simon Godsill and Dr. Jebu Rajan for proof reading sections of this thesis. Also Ray Auchterlounie, Pete Wilson, Ian Calderbank, Adam Tibbalds, and Alex Stark for maintaining the lab computer network.

When I started my undergraduate degree, my Dad offered to fund me as long as I was still studying. Thanks to grants from the Science and Engineering Research Council and latterly Trinity College I have not needed to take up his offer. That I always knew that my parents' love was there means a great deal to me. Thanks.

## **Keywords**

---

image processing; image sequence processing; Markov random fields; Markov chain Monte Carlo; motion pictures; motion estimation; dirt and sparkle; line scratch removal;

## Abstract

---

This thesis addresses a number of issues concerned with the restoration of one type of image sequence, namely archived black and white motion pictures. These are often a valuable historical record, but because of the physical nature of the film they can suffer from a variety of degradations which reduce their usefulness. The main visual defects are ‘dirt and sparkle’ due to dust and dirt becoming attached to the film, or abrasion removing the emulsion, and ‘line scratches’ due to the film running against foreign bodies in the camera or projector.

For an image restoration algorithm to be successful, it must be based on a mathematical model of the image. A number of models have been proposed, and here we explore the use of a general class of model known as *Markov Random Fields* (MRFs), based on Gibbs distributions, by analogy with models from statistical physics. The earliest such model was the Ising model, and subsequently these models have been developed for use in other fields. The Gibbs distribution is a probability distribution over all possible images, specified by local interactions between picture elements. The distribution is constructed such that of all possible images, those with the desired characteristics are assigned higher probability.

Because of the complexity inherent in these models, a number of numerical techniques have been developed to use them for practical processing problems. The Metropolis-Hastings algorithm and the Gibbs sampler provide the basic tools, constructing Markov Chains with the desired Gibbs distribution as the limiting distribution of the chain. These algorithms can also be used for other signal processing and inference problems. A number of deterministic approximations based on Mean Field theory are also used.

The problem of detecting ‘dirt and sparkle’ is found to be equivalent to finding temporal discontinuities in the sequence, corresponding to spurious motion estimates. A segmentation based approach working on motion estimates from a separate motion estimation algorithm is developed. To complete the restoration, a different Markov Random Field model is used to interpolate the detected area, the aim being to produce as seamless a restoration as possible.

This approach treats the motion estimation and ‘dirt and sparkle’ detection as two separate processes. The flexibility of the Gibbs distributions which may be constructed allows a field to be specified which includes both the motion estimates and the segmentation in one distribution. The use of an MRF model on the motion estimates helps to produce a motion estimation algorithm which gives close approximations to the true motion and occlusion. This integrates the two processes into one estimation stage. The workings of this estimation algorithm are explored and applied to the problem of scratch detection.

A slightly different approach is taken to the removal of ‘line scratches’. Their definite structure allows a strong model to be constructed. Because the number of line scratches present in any frame is unknown the standard Metropolis-Hastings and Gibbs sampler algorithms cannot be used. The Reversible Jump Markov chain Monte Carlo sampler is used to construct an algorithm for determining both the number and locations of the line scratches.

---

# CONTENTS

---

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Markov Chain Monte Carlo (MCMC) Methods</b>	<b>5</b>
2.1	Introduction . . . . .	5
2.2	Markov Chain Theory . . . . .	6
2.2.1	Invariant Distributions . . . . .	7
2.3	The Gibbs Sampler . . . . .	8
2.3.1	Example: Bayesian Marginal Inference . . . . .	10
2.4	Metropolis-Hastings Algorithm . . . . .	12
2.4.1	Example: Self-Structuring of Artificial Neural Networks . . . . .	16
2.5	Discussion . . . . .	24
2.6	Summary . . . . .	25
<b>3</b>	<b>Markov Random Fields and Scratch Removal from Motion Pictures</b>	<b>27</b>
3.1	Introduction . . . . .	27
3.2	Markov Random Field Image Models . . . . .	28
3.2.1	Neighbourhoods . . . . .	29
3.2.2	Cliques . . . . .	29
3.2.3	The Hammersley-Clifford Theorem . . . . .	30
3.2.4	Gibbs Distributions and Potential Functions . . . . .	31
3.3	Scratch Removal from Motion Pictures . . . . .	33
3.3.1	Introduction and Review . . . . .	33
3.3.2	An MRF Based Scratch Detection Algorithm . . . . .	40

---

3.3.3	Detection Results . . . . .	46
3.3.4	Missing Region Interpolation Using MRFs . . . . .	51
3.3.5	An MRF Based Interpolation Algorithm . . . . .	51
3.3.6	Interpolation Results . . . . .	55
3.3.7	Interpolating the Areas Detected by the Scratch Detection Algorithm	56
3.4	Summary . . . . .	57
<b>4</b>	<b>Combining the Motion Estimation and Scratch Detection</b>	<b>59</b>
4.1	Introduction . . . . .	59
4.2	Conventional Approaches to Motion Estimation . . . . .	61
4.2.1	Block Matching Motion Estimation . . . . .	61
4.2.2	Gradient Based Motion Estimation . . . . .	63
4.3	Hierarchical Markov Random Fields . . . . .	64
4.4	Estimating Motion, Discontinuities and Occlusion . . . . .	66
4.4.1	Estimating Motion and Discontinuities . . . . .	67
4.4.2	MRF Based Approaches to Motion Estimation . . . . .	68
4.4.3	Modelling the Occlusion . . . . .	70
4.4.4	Parameter Estimation . . . . .	72
4.5	Experimental Results . . . . .	75
4.5.1	‘Square’ Sequence . . . . .	75
4.5.2	‘Western’ Sequence . . . . .	75
4.6	Summary . . . . .	77
<b>5</b>	<b>Reversible Jump MCMC and Line Scratch Removal</b>	<b>81</b>
5.1	Introduction . . . . .	81
5.2	Reversible Jump MCMC . . . . .	82
5.3	Line Scratch Removal . . . . .	84
5.3.1	Introduction . . . . .	84

---

5.3.2	A Model for Line Scratches . . . . .	84
5.3.3	Constructing the Reversible Jump Sampler . . . . .	85
5.4	Results . . . . .	89
5.5	Summary . . . . .	92
<b>6</b>	<b>Conclusions and Suggestions for Further Research</b>	<b>94</b>
<b>7</b>	<b>Bibliography</b>	<b>96</b>
<b>A</b>	<b>Results used in proving the convergence of regular Markov chains</b>	<b>105</b>
<b>B</b>	<b>Training and updating the Volterra Connectionist Model</b>	<b>106</b>
B.1	Training the Volterra Connectionist Model (VCM) . . . . .	106
B.2	Updating the VCM for a change in the included expansion terms . . . . .	107
B.2.1	Adding a term . . . . .	107
B.2.2	Removing a term . . . . .	108
<b>C</b>	<b>Matrix inverse updating results</b>	<b>109</b>
C.1	Increasing the matrix size . . . . .	109
C.2	Decreasing the matrix size . . . . .	109
<b>D</b>	<b>Markov Chain state occupancy statistics</b>	<b>111</b>
D.1	Mean state occupancy . . . . .	111
D.2	Second moments of state occupancy . . . . .	111
D.3	Occupancies under random starting . . . . .	112
D.4	General 2-state process . . . . .	112
<b>E</b>	<b>Derivation of the acceptance probabilities for the Reversible Jump MCMC sampler</b>	<b>114</b>

---

## INTRODUCTION

---

The motivation behind this thesis is the restoration of archived black and white motion picture material, although the techniques developed have wider applicability. A large quantity of black and white film footage, particularly from newsreel archives, is of historical interest, and other films are of interest to the entertainment industry. The physical nature of the film material means that it is very susceptible to degradations. Not only does the physical medium decay over time, but the projection process, involving physically moving the film through the projector, can also cause the film to be damaged and degraded.

The type of degradation which is most visually annoying is ‘dirt and sparkle’, also referred to as ‘scratches’ or ‘blotches’. Vertical ‘line scratches’ which persist from frame to frame are also a problem, as, to a much lesser extent, is global noise – many black and white films have surprisingly high levels of detail and contrast in areas not affected by scratches or lines. Figure 1.1 shows a frame exhibiting all these defects. In this thesis we focus on the problems of ‘dirt and sparkle’ and ‘line scratches’. ‘Dirt and sparkle’ is the bright flashes and dark areas which appear randomly, due to dirt becoming attached to the film, or the emulsion being abraded during projection or rough handling. Fingerprints and other foreign objects can also cause this type of defect. ‘Line scratches’ are caused by the film material running against foreign bodies in either the camera or projector. This results in a narrow, vertical, bright or dark line being visible on the frames.

For a processing algorithm to be successful it must be based on a mathematical model of the image or image sequence. The model which we investigate in this thesis is known as the *Markov Random Field* (MRF) model. These models are specified by local interactions between the pixels. These interactions produce a probability distribution which is known as a Gibbs distribution, by analogy with models from statistical physics, such as the Ising and Potts models. The probability distribution is constructed such that of all possible images, those with the desired characteristics are assigned higher probabilities.

Because of the complexity of the probability distributions used in these models, specialised numerical techniques have been developed. The Metropolis algorithm was originally developed for exploring statistical physics models, and was generalised by Hastings in the statistics literature. The Gibbs sampler was developed for use specifically with MRF models, and has subsequently found application elsewhere. All these algorithms work by



Figure 1.1: Frame from degraded sequence showing dirt and sparkle, line scratches and noise

constructing Markov chains with the desired Gibbs distribution as the chain's invariant distribution. The statistical physics techniques of mean field theory provide a number of deterministic approximations which are also used. We demonstrate the use of these algorithms in applications in Bayesian marginal inference and in developing a new, fast self-structuring algorithm for generalised single layer networks.

Most conventional image sequence processing algorithms perform some form of global filtering of the frames. The degradation that is 'dirt and sparkle' is, however, local in nature and this motivates the development of scratch detection algorithms. The problem of detecting 'dirt and sparkle' is found to be equivalent to finding temporal discontinuities in the sequence. We use an MRF model to encode the spatial continuity of the blotches, and develop a detection algorithm which is more effective at correctly identifying scratch regions than all previous detectors. We compare the working of various exact and approximate optimisation algorithms on this detection formulation, and also on the MRF which is used to interpolate the detected areas.

This approach to detecting scratches is a post-processing state after motion estimation has been performed. As such the effectiveness of the algorithm is to some extent a function of the motion estimation algorithm used. MRF models can be used to model the image motion, and also to integrate the scratch detection and motion estimation into one process. We develop a consistent MRF integrating the motion estimation and scratch detection and perform experiments to assess its performance in detecting scratches.

The approach taken to the removal of 'line scratches' is slightly different. In addressing this problem we focus less on the image model, but build a very strong model of the effect

of the line scratches' presence on the image. This parametric model can then be fitted to the image data. The number of line scratches present on any frame is unknown, however, and we explore the use of the *Reversible Jump MCMC* framework to construct a sampling algorithm which enables us to sample the number of line scratches present together with the scratches' parameters. This algorithm is found to be effective on real degraded image frames.

The organisation of this thesis is as follows:

## **Chapter 2: Markov Chain Monte Carlo (MCMC) Methods**

We begin by reviewing the Markov chain Monte Carlo (MCMC) algorithms that will be used in later chapters. The necessary theory of Markov chains is presented, followed by descriptions of the Gibbs sampler and the Metropolis-Hastings algorithm. An example from Bayesian inference is used to illustrate the Gibbs sampler, and the Metropolis-Hastings algorithm is used to develop a new, fast algorithm for the self-structuring of generalised single layer networks. Discussion of more technical aspects of the use of Markov chain Monte Carlo methods is given.

## **Chapter 3: Markov Random Fields and Scratch Removal from Motion Pictures**

An introduction to the theory of Markov Random Field image models is presented. We then use this theory to develop a new scratch detection algorithm. We show how mean field theory can be used with this algorithm to give deterministic solutions. Experiments demonstrate the improvements this algorithm gives over current detectors. An MRF interpolation algorithm is also developed.

## **Chapter 4: Combining the Motion Estimation and Scratch Detection**

In this chapter we use MRFs to combine the motion estimation and scratch detection. Justification for our particular formulation is given. The use of this combined algorithm is demonstrated and found to be extremely effective, especially at suppressing false detections.

## **Chapter 5: Reversible Jump MCMC and Line Scratch Removal**

This chapter contains material on the Reversible Jump MCMC sampler, an extension of the Metropolis-Hastings algorithm. It allows us to sample from both the model structure and the models' parameters. Using this sampler we construct an algorithm to determine the number of line scratches present in a motion picture frame, and also to locate the line scratches.

## **Chapter 6: Conclusions and Suggestions for Future Research**

This chapter draws conclusions from the work preceding it, highlighting the progression of the algorithms. It also discusses proposals for future research, improving the current algorithms and developing new ones for other problems in the field.

---

## MARKOV CHAIN MONTE CARLO (MCMC) METHODS

---

In this chapter two Markov chain Monte Carlo algorithms are detailed, namely the Gibbs Sampler and the Metropolis-Hastings algorithm. These algorithms rely heavily on the theory of Markov chains, and the relevant theory is presented in section 2.2. The Gibbs Sampler is discussed in section 2.3, and an example of its use in marginal Bayesian inference given. The more general Metropolis-Hastings algorithm is presented in section 2.4 and its use in the self-structuring of Generalised Single Layer Networks is contained in section 2.4.1. A discussion of some further considerations of the use of these algorithms is given in section 2.5.

### 2.1 Introduction

In many areas of inference we deal with complex probability distributions [7]. The complexity may be due to the large number of variables involved, as in many of the examples in later chapters, or due to the form of the density [86]. In both cases little analytical progress can be made, and we must turn instead to simulation methods, whereby we try to build up a picture of the characteristics of a distribution, or a function of that distribution, from a number of samples drawn from that distribution [93].

Let  $p(\mathbf{x})$  be the joint distribution of the variables  $\{x_i : i = 1 \dots n\}$ , and for simplicity let the  $x_i$  take values from a discrete state-space  $\Omega$ . We wish to generate samples  $\hat{\mathbf{x}}$  distributed as  $p(\mathbf{x})$ . For some forms of  $p(\mathbf{x})$  it may be possible to generate samples directly, by some transformation of uniformly distributed variates [79]. The multivariate Gaussian distribution is a good example of this. For distributions where this is not possible, rejection sampling has been used to generate samples. This involves sampling from some other distribution,  $q(\mathbf{x})$ , and then accepting that sample as a sample from  $p(\mathbf{x})$  with probability  $p(\hat{\mathbf{x}})/(cq(\hat{\mathbf{x}}))$  where  $c$  is such that  $cq(\mathbf{x}) > p(\mathbf{x})$ . The efficiency of this method depends critically on the choice of  $q(\mathbf{x})$  – it must both approximate  $p(\mathbf{x})$  well and be easy to sample from. Often, especially in high-dimensional problems, these two desiderata will be in conflict. This often leads to inefficient sample generation, as many proposals are rejected [92].

At the cost of producing a sequence of correlated samples, rather than a set of indepen-

dent ones, the theory of Markov chains provides a more efficient method of sampling from a distribution [43]. In outline, we wish to construct a Markov chain which has as its invariant distribution the desired distribution,  $p(\mathbf{x})$ . An invariant distribution of a Markov chain is a distribution over the states,  $\pi(\mathbf{x})$ , which is maintained as the chain undergoes transitions. Simulating the chain will cause the chain to wander through the states, visiting each with probability  $p(\mathbf{x})$ . For this to be a practical procedure we must construct the chain such that the following conditions are met.

1.  $p(\mathbf{x})$  is indeed the invariant distribution,  $\pi(\mathbf{x})$ .
2. This invariant distribution is unique.
3. The chain converges to this invariant distribution, irrespective of the initial distribution.

Clearly condition 3 is important, as if we could start the chain in a state drawn from  $p(\mathbf{x})$ , the whole process would be unnecessary.

The next section presents the relevant Markov chain theory.

## 2.2 Markov Chain Theory

Markov chains are the first generalisation of independent events – the probability for the event at time  $n + 1$  is a function only of the outcome of the event at time  $n$ . Formally, for a set of random variables  $X^{(0)}, X^{(1)} \dots$  we have

$$p(X^{(n+1)} = \mathbf{x}^{(n+1)} | X^{(k)} = \mathbf{x}^{(k)}, k = 0 \dots n) = p(X^{(n+1)} = \mathbf{x}^{(n+1)} | X^{(n)} = \mathbf{x}^{(n)})$$

and the chain is fully specified by giving an *initial distribution*,  $p_0(\mathbf{x})$ , and the *transition probabilities*,  $T_n(\mathbf{x}^{(n)}, \mathbf{x}^{(n+1)})$ . A chain is called *homogeneous* if the transition probabilities are not a function of  $n$ . For most of this section we assume for simplicity that the  $x_i$  take values from a common, discrete, state-space,  $\Omega$ . The distribution after  $n + 1$  steps,  $p_{n+1}(\mathbf{x})$  is related to  $p_n(\mathbf{x})$  by

$$p_{n+1}(\mathbf{x}) = \sum_{\mathbf{x}'} p_n(\mathbf{x}') T_n(\mathbf{x}', \mathbf{x}), \mathbf{x}' \in \Omega^K$$

where  $K$  is the dimensionality of  $\mathbf{x}'$ , and this can be iterated back to give  $p_{n+1}(\mathbf{x})$  in terms of  $p_0(\mathbf{x})$ . The transition probabilities may be gathered into a transition matrix  $\mathbf{T} = \{T(\mathbf{x}', \mathbf{x})\}$ . In subsequent sections we will treat only homogeneous chains.

### 2.2.1 Invariant Distributions

As explained above we must show that a Markov chain can converge to an invariant distribution, irrespective of the initial distribution. The following is taken from [54, chapter 4].

Assume  $\mathbf{T}$  is a regular transition matrix, that is  $\mathbf{T}^N$  has no zero entries for some  $N$ . This ensures that it is possible to move from any state to any other state in at most  $N$  steps along the chain. Then

- (i)  $\mathbf{T}^n \rightarrow \mathbf{A}$ , a probability matrix.
- (ii) Each row of  $\mathbf{A}$  is the same probability vector  $\alpha$ .
- (iii) The components of  $\alpha$  are positive.
  - (a) For any  $\pi$ ,  $\pi\mathbf{T}^n \rightarrow \alpha$ ,  $n \rightarrow \infty$ .
  - (b)  $\alpha$  is the unique distribution satisfying  $\alpha\mathbf{T} = \alpha$ .

Where  $\pi$  is a probability distribution over the states of the chain.

This will cover conditions 2 and 3 above. Condition one is met by careful choice of the transition probabilities  $\mathbf{T}$ , and will be discussed later (sections 2.3, 2.4).

To derive the above results we make use of the results contained in Appendix A.

Assume for simplicity that the minimum entry of  $\mathbf{T}$  is  $\eta > 0$ . Let  $\rho_j$  be a column vector with a 1 in the  $j^{\text{th}}$  entry and zeros elsewhere. Let  $M_n, m_n$  be the maximum and minimum entries of  $\mathbf{T}^n \rho_j$  respectively.

Now

$$\mathbf{T}^n \rho_j = \mathbf{T} \mathbf{T}^{n-1} \rho_j$$

and the results in appendix A give

$$M_1 \geq M_2 \geq M_3 \dots$$

$$m_1 \leq m_2 \leq m_3 \dots$$

$$M_n - m_n \leq (1 - 2\eta)(M_{n-1} - m_{n-1}), n \geq 1$$

Let  $d_n = M_n - m_n$ , giving

$$d_n \leq (1 - 2\eta)^n d_0 = (1 - 2\eta)^n$$

Hence as  $n \rightarrow \infty$ ,  $d_n \rightarrow 0$ ,  $M_n \rightarrow m_n$  and  $\mathbf{T}^n \rho_j$  tends to a vector with all components equal,  $a_j$ . Also  $m_n \leq a_j \leq M_n$ . Since  $0 < m_1$  and  $M_1 < 1$ ,  $0 < a_j < 1$ .

Now  $\mathbf{T}^n \rho_j$  is the  $j^{\text{th}}$  column of  $\mathbf{T}^n$ , hence the columns of  $\mathbf{T}^n$  tend to constant values. Hence  $\mathbf{T}^n \rightarrow \mathbf{A}$ , a matrix with all rows the same, and row sums equal to one, as the row sums of  $\mathbf{T}^n$  equal one.

For  $\mathbf{T}$  having some zeros entries, but  $\mathbf{T}^N$  having only non-zero, use  $\eta'$  as the smallest entry of  $\mathbf{T}^N$  and the above derivation holds.

We are now in a position to show a) convergence and b) uniqueness.

(a) convergence

$\pi \mathbf{A} = \pi \zeta \alpha = \alpha$ , where  $\zeta$  is a column vector of ones. Now  $\mathbf{T}^n \rightarrow \mathbf{A}$ , hence  $\pi \mathbf{T}^n \rightarrow \alpha$  for arbitrary  $\pi$ .

(b) uniqueness

Let  $\beta$  be a probability vector such that  $\beta \mathbf{T} = \beta$ . By (a)  $\beta \mathbf{T}^n \rightarrow \alpha$  and  $\beta \mathbf{T}^n = \beta$ , hence  $\beta = \alpha$  and there is only one stationary distribution.

The chain is also *aperiodic* if there is finite probability of remaining in the same state at each transition. Nothing in the above derivations has given any indication as to the *rate* of convergence to the stationary distribution, indeed this is an active research area for the types of transition matrix we will be considering later [73, 81].

The stationary distribution may also be demonstrated directly by showing either that

$$\sum_{\mathbf{x}'} \pi(\mathbf{x}') T(\mathbf{x}', \mathbf{x}) = \pi(\mathbf{x})$$

or that the *detailed balance* condition

$$\pi(\mathbf{x}) T(\mathbf{x}, \mathbf{x}') = \pi(\mathbf{x}') T(\mathbf{x}', \mathbf{x}) \quad (2.1)$$

holds for an *homogeneous* Markov chain [71].

## 2.3 The Gibbs Sampler

In the previous section we showed that certain Markov chains converge to an unique invariant distribution, but made no mention of how to construct the transitions such that the desired distribution  $p(\mathbf{x})$  was indeed the chain's invariant distribution  $\pi(\mathbf{x})$ . In this section the *Gibbs sampler*, due to Geman [33] will be outlined, and shown to produce a chain

which converges to the desired invariant distribution. Issues concerning the application and generality of the method will be discussed later in section 2.5.

Consider again the joint distribution  $\pi(x_1 \dots x_n)$ . Begin with  $\mathbf{x}^0$  and proceed as

$$\begin{aligned} x_1^1 &\leftarrow \pi(x_1|x_2^0, x_3^0 \dots x_n^0) \\ x_2^1 &\leftarrow \pi(x_2|x_1^1, x_3^0, x_4^0 \dots x_n^0) \\ x_3^1 &\leftarrow \pi(x_3|x_1^1, x_2^1, x_4^0 \dots x_n^0) \\ &\vdots \\ x_n^1 &\leftarrow \pi(x_n|x_1^1 \dots x_{n-1}^1) \end{aligned}$$

where  $\leftarrow$  indicates 'is drawn from the distribution of'. This completes one cycle of the Gibbs sampler and is equivalent to a transition probability of [93]

$$T(\mathbf{x}', \mathbf{x}) = \prod_{l=1}^n \pi(x_l|x'_j, j > l, x_j, j < l)$$

We now show that this set of transition probabilities has  $\pi(\mathbf{x})$  as its unique invariant distribution. We do this directly.

$$\begin{aligned} \sum_{\mathbf{x}'} \pi(\mathbf{x}') T(\mathbf{x}', \mathbf{x}) &= \sum_{\mathbf{x}'} \pi(\mathbf{x}') \prod_{l=1}^n \pi(x_l|x'_j, j > l, x_j, j < l) \\ &= \pi(x_n|x_1 \dots x_{n-1}) \sum_{\mathbf{x}'} \pi(\mathbf{x}') \pi(x_1|x'_2 \dots x'_n) \pi(x_2|x_1 x'_3 \dots x'_n) \dots \pi(x_{n-1}|x_1 x_2 \dots x_{n-2} x'_n) \\ &= \pi(x_n|x_1 \dots x_{n-1}) \sum_{x'_2 \dots x'_n} \pi(x'_2 \dots x'_n) \pi(x_1|x'_2 \dots x'_n) \pi(x_2|x_1 x'_3 \dots x'_n) \dots \pi(x_{n-1}|x_1 \dots x_{n-2} x'_n) \\ &= \pi(x_n|x_1 \dots x_{n-1}) \pi(x_1) \pi(x_2|x_1) \pi(x_3|x_1 x_2) \dots \pi(x_{n-1}|x_1 \dots x_{n-2}) \\ &= \pi(\mathbf{x}) \end{aligned}$$

Hence application of the Gibbs sampler maintains the correct invariant distribution.

In the earlier derivation of convergence we required that all the elements of  $\mathbf{T}^N$  be greater than zero for some  $N$ . This is equivalent to saying that we may move from any state to any other state in  $N$  moves. This will definitely be possible if all of the full conditional distributions used are positive over the entire single variable state space. If this is not true then the chain may still converge. The chain will be ergodic as at each state there is a finite probability of remaining in the same state.

The efficient use of the Gibbs sampler depends on the ease with which we may draw samples from the full conditional distributions. For variables defined over small state spaces it is possible to evaluate the probability distribution explicitly, and to use rejection sampling to generate the variates. This is the technique used in later chapters when the Gibbs sampler is employed. For continuous conditional distributions, for efficient variate generation the distribution must be one for which we can generate samples easily, either via direct transformation from uniform variates, or via an efficient rejection sampling scheme. Often these distributions will be of a standard, parameterised form, as in the example below. The use of *conjugate priors* [5] helps ensure that the distributions are simple to sample from.

### 2.3.1 Example: Bayesian Marginal Inference

Bayes theorem provides a consistent method for probabilistic inference [83, 26]. In employing this form of inference, we are often trying to estimate parameters of a model from noisy data, or, as in this section, estimate probability distributions for the parameters from the data - there are many ways of making point estimates from knowledge of the distribution [98].

In the Bayesian paradigm the values the model parameters took when the data was generated are considered as having had fixed, but unknown values. Our state of knowledge of these values is, however, uncertain, and it makes sense to describe our state of knowledge using probability distributions. Bayes theorem then provides the method for updating our state of knowledge, from the *a-priori* distribution we had before the data was measured, to the *a-posteriori* distribution which results from combining our prior beliefs with the information contained in the measurements.

Consider for concreteness the simple straight line model  $y = mx + c$ . This has two parameters, the slope,  $m$ , and the intercept,  $c$ . Clearly a-priori all values of slope and intercept are equally likely - our prior distributions are uniform over all possible values of  $m$  and  $c$ .

Assume now that we make  $N$  measurements,  $d_i$ ,  $i = 1 \dots N$  where  $d_i = y_i + n_i$ , and  $n_i$  is Gaussian noise with unknown variance  $\sigma^2$ . What are the probability distributions we now assign to the parameters?

Bayes theorem gives

$$p(m, c, \sigma | \mathbf{d}) = p(\mathbf{d} | m, c, \sigma) p(m, c, \sigma) / p(\mathbf{d})$$

The three terms on the right hand side are the *likelihood*, the *prior* and the *evidence*

(or marginalised likelihood) respectively. The last being a term useful in model selection [75, 16], but which will be treated as a constant normalising factor in this section as we are not interested here in discriminating between models.

The term  $p(\mathbf{d}|m, c, \sigma)$ , the likelihood, expresses our belief in how probable the observed data is, given particular values of the model parameters. In this case it is equivalent to the distribution of the modelling error or noise, because the Jacobian of the transformation from  $n$  to  $d$  is unity, and since this noise is Gaussian

$$p(\mathbf{d}|m, c, \sigma) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left(\frac{-1}{2\sigma^2} \sum_{i=1}^N (d_i - y_i)^2\right)$$

The term  $p(m, c, \sigma)$  represents our prior knowledge. There is no prior reason to assume that the model parameters are correlated, and so we use independent priors,  $p(m)$ ,  $p(c)$ ,  $p(\sigma)$ . As discussed earlier, uniform priors are used for  $m$  and  $c$ . The noise variance,  $\sigma$ , is a *scale* parameter, and so a Jeffrey's prior is used [53],  $p(\sigma) \propto 1/\sigma$ .

Combining the likelihood and the prior, and neglecting constants, gives the posterior

$$p(m, c, \sigma|\mathbf{d}) \propto \sigma^{-(N+1)} \exp\left(\frac{-1}{2\sigma^2} [D_2 - 2cD_1 - 2mX_D + 2mcX_1 + m^2X_2 + Nc^2]\right)$$

where  $D_1 = \sum_{i=1}^N d_i$ ,  $D_2 = \sum_{i=1}^N d_i^2$ ,  $X_D = \sum_{i=1}^N x_i d_i$ ,  $X_1 = \sum_{i=1}^N x_i$ ,  $X_2 = \sum_{i=1}^N x_i^2$ .

We are interested here in the *marginal* distributions,  $p(m|\mathbf{d})$ ,  $p(c|\mathbf{d})$ ,  $p(\sigma|\mathbf{d})$ , which summarise our posterior beliefs about each parameter separately. For this example the process of marginalisation, or integrating over the unwanted parameters, can be performed analytically, with the following results.

$$\begin{aligned} p(m|\mathbf{d}) &\propto \left[ D_2 - 2mX_D + m^2X_2 - \frac{(mX_1 - D_1)^2}{N} \right]^{-\frac{N-1}{2}} \\ p(c|\mathbf{d}) &\propto \left[ D_2 - 2cD_1 + Nc^2 - \frac{(cX_1 - X_D)^2}{X_2} \right]^{-\frac{N-1}{2}} \\ p(\sigma|\mathbf{d}) &\propto \sigma^{-N+1} \exp\left(\frac{-1}{2\sigma^2} \left[ D_2 - \frac{D_1^2}{N} - \frac{1}{N} \frac{(X_1D_1 - NX_D)^2}{NX_2 - X_1^2} \right]\right) \end{aligned}$$

The purpose of this section is to now show how the Gibbs sampler can be used to form numerical estimates of these distributions, without the need for any analytic marginalisation.

We showed above how the Gibbs sampler can be used to draw samples from the joint distribution  $p(m, c, \sigma|\mathbf{d})$ , from knowledge of the conditional distributions  $p(m|c, \sigma, \mathbf{d})$ ,

$p(c|m, \sigma, \mathbf{d})$ ,  $p(\sigma|m, c, \mathbf{d})$ . These conditionals are very easily derived to within a normalising factor from the joint distribution by considering all the parameters which we are not interested in to be fixed at their current values. This results in  $p(m|c, \sigma, \mathbf{d})$  and  $p(c|m, \sigma, \mathbf{d})$  both having Gaussian distributions, with means of  $(X_D - cX_1)/X_2$ ,  $(D_1 - mX_1)/N$  and variances of  $\sigma/\sqrt{X_2}$ ,  $\sigma/\sqrt{N}$  respectively. Gaussian variates are easy to generate, and so sampling from these two conditional densities is straightforward [76]. The third conditional density  $p(\sigma|m, c, \mathbf{d})$  has the form

$$p(\sigma|m, c, \mathbf{d}) \propto \sigma^{-(N+1)} \exp\left(\frac{-1}{2\sigma^2} \sum_{i=1}^N (d_i - mx_i - c)^2\right)$$

which is an *inverse chi* distribution with  $N$  degrees of freedom [64], and variates from this distribution may be generated by using a transformation applied to variates drawn from a Gamma distribution with  $N/2$  degrees of freedom [86]. Thus all three conditionals may be sampled easily to generate samples from the joint distribution  $p(m, c, \sigma|\mathbf{d})$ , using the Gibbs sampler.

From these samples there are a number of methods that can be used to obtain estimates of the marginal distributions [29]. Clearly a histogram estimate may be formed simply by plotting the histogram of each variable separately. A pointwise estimate of the marginal distribution may be formed by averaging the distributions found for each of the samples. Here we present results using the first of these two methods.

Figure 2.1 shows the data used in the experiment - it consists of two hundred points. The analytic and histogram estimates for the marginal distributions are plotted in figures 2.2, 2.3 and 2.4. The histograms were plotted from 10,000 iterations of the Gibbs sampler. Clearly the histogram estimates are very close to the analytic results, even for the case of the variance, which is a non-Gaussian, skew distribution. This demonstrates the ease with which the Gibbs sampler may be used for Bayesian inference - in later chapters we will discuss cases where analytic results are not available, and simulation methods are the only practical tools available.

## 2.4 Metropolis-Hastings Algorithm

Since its introduction in 1953 [69], the Metropolis algorithm has been widely used in statistical physics, and its use has recently become more widespread in the more general statistics literature. It is the first example of a Markov chain Monte Carlo algorithm, and was generalised by Hastings [43] to give the Metropolis-Hastings algorithm.

The basic Metropolis algorithm constructs the Markov chain by a series of randomly

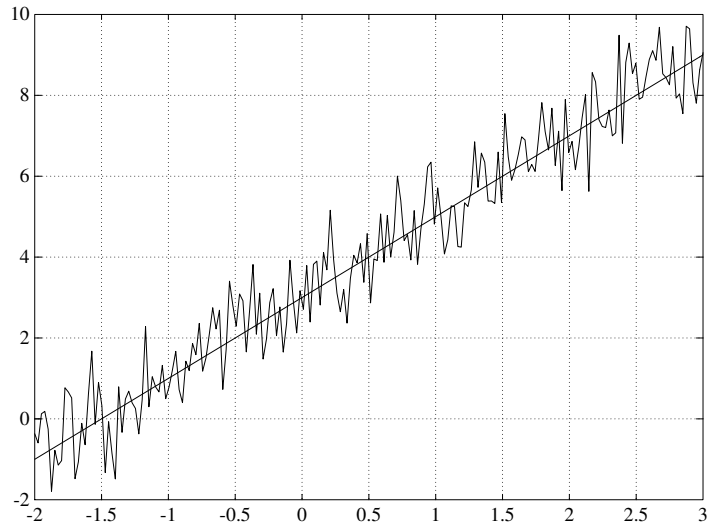


Figure 2.1: Data - straight line with additive gaussian noise

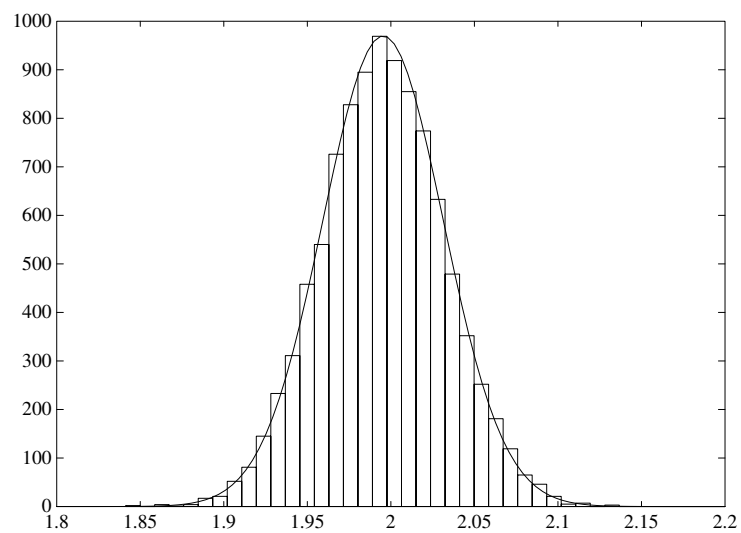


Figure 2.2: Slope - analytic marginal and histogram estimate

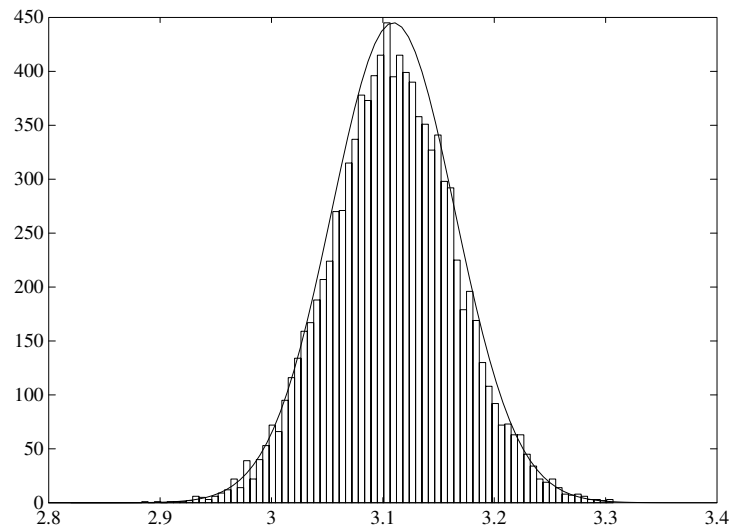


Figure 2.3: Intercept - analytic marginal and histogram estimate

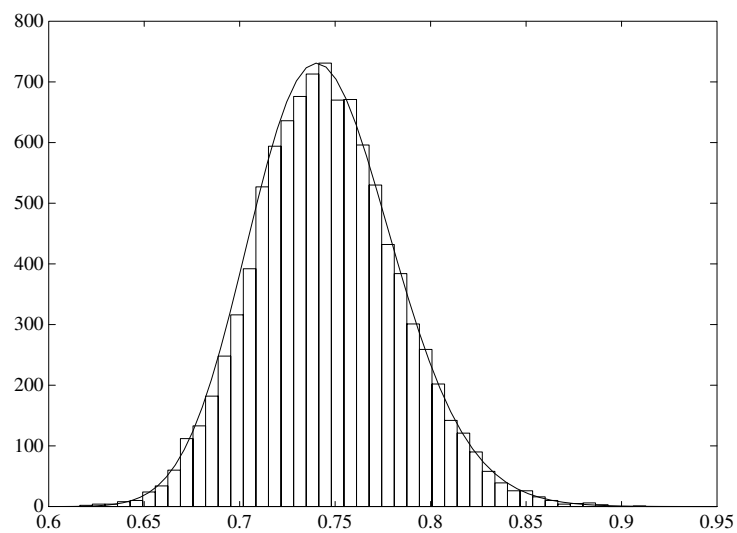


Figure 2.4: Noise variance - analytic marginal and histogram estimate

generated changes. These proposed changes to the current state are either accepted, forming the new state, or rejected, when the old state is retained as the new state. The choice of proposed changes, and the decision of when to accept the change, is made in a manner that ensures the chain converges to the correct invariant distribution. The algorithm functions as follows

1. Select a candidate state from a *symmetric* proposal distribution  $q(\mathbf{x}, \mathbf{x}')$ .
2. With probability  $A(\mathbf{x}, \mathbf{x}')$  accept the proposed state as the new state, else retain the old state.

The acceptance function used by Metropolis *et al.* is

$$A(\mathbf{x}, \mathbf{x}') = \min(1, \pi(\mathbf{x}')/\pi(\mathbf{x}))$$

and this can be shown to produce the correct invariant distribution, most easily by using the detailed balance condition of equation 2.1. The chain so constructed will also be ergodic if it is possible to move from any state to any other state. This will be the case if  $\pi(\mathbf{x})$  is everywhere positive, and the proposal distribution is such that all states may be proposed in a finite number of moves.

Hastings [43] generalised the Metropolis algorithm to allow non-symmetric proposal distributions. This requires a modification to the acceptance probability function, which becomes

$$A'(\mathbf{x}, \mathbf{x}') = \min\left(1, \frac{\pi(\mathbf{x}')q(\mathbf{x}', \mathbf{x})}{\pi(\mathbf{x})q(\mathbf{x}, \mathbf{x}')}\right) \quad (2.2)$$

The proposal function may now also depend on which element of  $\mathbf{x}$  is being changed. Again the detailed balance condition of equation 2.1 can be used to demonstrate that the correct invariant distribution is achieved.

$$\begin{aligned} \pi(\mathbf{x})T(\mathbf{x}, \mathbf{x}') &= \pi(\mathbf{x})q(\mathbf{x}, \mathbf{x}')A(\mathbf{x}, \mathbf{x}') \\ &= \min(\pi(\mathbf{x})q(\mathbf{x}, \mathbf{x}'), \pi(\mathbf{x}')q(\mathbf{x}', \mathbf{x})) \\ &= \min(\pi(\mathbf{x}')q(\mathbf{x}', \mathbf{x}), \pi(\mathbf{x})q(\mathbf{x}, \mathbf{x}')) \\ &= \pi(\mathbf{x}')q(\mathbf{x}', \mathbf{x})A(\mathbf{x}', \mathbf{x}) \\ &= \pi(\mathbf{x}')T(\mathbf{x}', \mathbf{x}) \end{aligned}$$

Choice of the proposal distribution such that any state can be reached in a finite number of steps will again ensure ergodicity and hence convergence to the invariant distribution.

A number of proposal distributions are commonly used. For discrete variables a proposal distribution uniform over all the states, or uniform over all the states except the current state is a common choice, and we will use the latter in the example below. For continuous variables the proposal distribution often takes the form of a distribution from which it is easy to sample, for example a Gaussian, centered at the current value. A uniform distribution with some spread about the current value is another common choice.

In relation to the use of the Gibbs sampler, the restriction that we require the conditional distributions to be easy to sample from is lifted by the Metropolis-Hastings algorithm. Instead we must now find a proposal distribution which generates changes which are accepted in stage 2 of the algorithm - this can be very problem dependent. Again we are not required to be able to normalise the distribution - probabilities only enter in the form of ratios  $\pi(\mathbf{x}')/\pi(\mathbf{x})$ , and often, because of the form of the distribution, this allows proposals to be found which allow very rapid calculation of the acceptance probability, as we shall see later. If in equation 2.2 we consider only changing one variable, and use as the proposal distribution the conditional distribution of that variable given the current values of the other variables, then  $A'(\mathbf{x}, \mathbf{x}') = 1$ , all proposals are accepted, and we have the Gibbs sampler.

We now demonstrate the use of the Metropolis-Hastings algorithm in the self-structuring of artificial neural networks, where it enables a fast, effective algorithm to be developed. In this example, under a reasonable assumption, we may also calculate theoretical values for the convergence time of the Markov chain, and the length of simulation required for results of a specified accuracy.

### 2.4.1 Example: Self-Structuring of Artificial Neural Networks

This section considers the problem of self-structuring of the Generalized Single Layer Network [47]. A typical network is illustrated in figure 2.5. It comprises a single layer of  $N$  hidden nodes, each of which computes a different fixed non-linear function  $\phi_k(\mathbf{x})$  of the input vector. This maps the input vector into an extended vector. The output is a weighted sum of the nonlinear extensions, defining a linear function in the extended space. The output of the network for input  $\mathbf{x}^i$  is<sup>1</sup>

$$d_i = \sum_{k=1}^N w_k \phi_k(\mathbf{x}^i) = \mathbf{w}^T \Phi(\mathbf{x}^i)$$

---

<sup>1</sup>Note that the use of  $\mathbf{x}$  in this section differs from its use in previous sections.

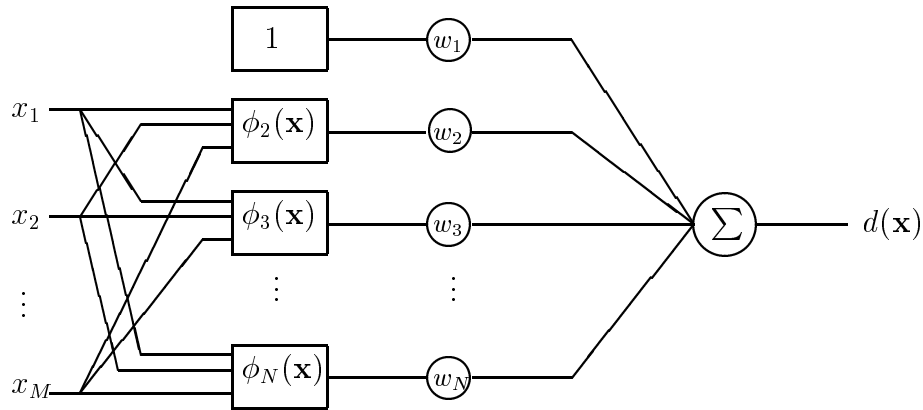


Figure 2.5: Generalised Single Layer Network

where  $\mathbf{w}$  is the weight vector and  $\Phi$  is the vector whose components are the  $\phi_k$ . The optimal weights (under the minimum SSE criterion) for this type of network can be simply calculated.

Clearly, the actual basis functions used will determine how easily the network will learn a particular function. A number of different sets of basis functions are commonly used, namely Radial Basis functions [66], Volterra Expansion functions [90, 67] and polynomials, all of which produce useful networks. Combinations of these bases may also be used. A truncated Volterra Expansion has been used to produce the nonlinear functions in the examples presented in this section.

The  $V(N, K)$  Volterra Expansion results in the network computing the following function

$$\begin{aligned}
 d(\mathbf{x}) &= \mathbf{w}^T \Phi(\mathbf{x}) \\
 &= w_0 \\
 &+ \sum_{i_1=1}^N w_{i_1} x_{i_1} \\
 &+ \sum_{i_1=1}^N \sum_{i_2=1}^N w_{i_1 i_2} x_{i_1} x_{i_2} \\
 &\vdots \\
 &+ \sum_{i_1=1}^N \sum_{i_2=1}^N \cdots \sum_{i_K=1}^N w_{i_1 i_2 \dots i_K} x_{i_1} x_{i_2} \cdots x_{i_K}
 \end{aligned}$$

that is, an expansion up to order  $K$  of an input vector of length  $N$ . The task of a self-structuring algorithm is to determine which of these terms is important to solve the problem at hand.

Most conventional self-structuring algorithms are based on growing and pruning [87]. A growing algorithm is a method for adding terms to a small network [25], whilst a pruning algorithm removes terms from a large one [63]. In many cases considerable computational effort is required to determine the change to be made to the network, and very often sub-optimal results are obtained. Many of these algorithms can be considered as a form of steepest descent in the discrete space of the network architecture [27], that is, at each iteration they make the change to the architecture which results in the largest decrease in cost function. As such they find a local minimum of some cost function. Unless the cost function is chosen carefully even the global minimum is often unsuitable, as this would give a network which significantly over-fitted the data. We now show how the Metropolis-Hastings algorithm may be used to overcome these difficulties

We use the Metropolis-Hastings algorithm to explore the space of models, the models being visited with a frequency which is a function of how well they fit the training data. Thus models which fit the data well will be visited relatively often. We hypothesise that these models will *all* contain the expansion terms which are *necessary* to model the training data, together with other, irrelevant terms, chosen essentially at random. By looking at which terms are always included we can decide which of the expansion terms are necessary. The method of exploration of the model space allows us to specify the number of models which must be visited in order to determine with a given probability the necessary expansion terms.

### Algorithm

The self-structuring algorithm is based on the following assumption

*Any model comprised of at least all the terms of the ‘true’ model will have a small sum squared error (SSE) when presented with the training data, and any model not including all the terms of the ‘true’ model will have a large SSE when presented with the training data.*

There are  $2^N$  possible models. Define the probability distribution over the space of possible models

$$p(\text{model } k) = \frac{1}{Z} \exp\left(-\frac{1}{T} (\text{SSE}(k))\right) \quad (2.3)$$

where  $\text{SSE}(k)$  is the minimum SSE associated with model  $k$ , and  $T$  scales the distribution. Appendix B contains details of calculating the minimum SSE for a given model. Under the assumption, this distribution places larger probability mass on those models which

include all the terms of the ‘true’ model. We now use the Metropolis-Hastings algorithm to construct a Markov chain with the distribution of equation 2.3 as its limiting distribution. A sequence of models is thus generated by the following procedure.

1. Begin with a random model, chosen such that each term in the  $V(N, K)$  expansion is included with probability  $1/2$ . Store this model in a list.
2. Choose one of the expansion terms at random.
3. Postulate a change to that term, and calculate  $\Delta\varepsilon$ , the change in SSE, from equation B.2 or B.4.
4. Accept this change if  $\exp(-(1/T)\Delta\varepsilon) > \text{rand}(0, 1)$  else reject the change. (Where  $\text{rand}(0, 1)$  is uniformly distributed between 0 and 1.)
5. Add the current model to the stored list, update  $\mathbf{R}_t^{-1}$  (see Appendix B). Goto stage 2.

Thus we use a proposal distribution which is uniform over all states except the current state in stage three of the algorithm, and the Metropolis acceptance function in stage 4.

Examination of the stored list allows the terms of the ‘true’ model to be determined.

The algorithm will pass through an initial transient phase, until all the terms of the ‘true’ model have been included. After this phase, under the assumption and the method of construction of the Markov chain, the ‘true’ terms will be included in (almost) every model in the stored list, as removing one of the ‘true’ model terms will cause a large increase in SSE, and such a change will almost always be rejected in step 4. The other terms will be included in half the models in the list, as changing the inclusion status of one of these terms will have little effect on the SSE, and hence will be accepted in step 4. The variance of the number of inclusions of these other terms will depend on  $N$  and the number of iterations.

### Convergence results

The assumption above has the following corollaries

1. After a term of the ‘true’ model has been ‘visited’, it will be included in (almost) all subsequent models.
2. When a term not in the ‘true’ model is ‘visited’, the change will (almost) always be accepted.

The first corollary allows a value for the length of the transient phase to be estimated. It is given by one of the classic ‘balls in urns’ problems of probability theory.

There are  $N$  urns. Each initially contains a ball with probability  $1/2$ . Balls are placed in the urns at random. What is the probability that after  $n_1$  balls have been placed, none of the urns are empty?

The probability that the transient phase has not been completed by iteration  $n_1$  is given approximately by<sup>2</sup>

$$p_{tr} = 0.5 \left( \frac{N-1}{N} \right)^{n_1} \quad (2.4)$$

The second corollary allows the length of run to ensure good separation of the ‘true’ and unwanted terms to be estimated. It says that the inclusion of an unwanted term follows a two state Markov Chain with symmetric transition probabilities, and transition matrix

$$\mathbf{T} = \frac{1}{N} \begin{bmatrix} N-1 & 1 \\ 1 & N-1 \end{bmatrix}$$

Appendix D derives results for the mean and variance of the state occupancies of each of these chains. After  $n_2$  transitions the mean is  $(n_2 + 1)/2$  and the variance is given by

$$v_1^v(n_2) \simeq n_2 \frac{(N-1)}{4} - \frac{N^2 - 4N + 2}{8} \quad (2.5)$$

A value for  $n_2$  can be chosen such that the difference between the mean inclusion value and  $n_2$  is a sufficiently large number of standard deviations, so that discrimination between wanted and unwanted terms is assured with a sufficiently high probability.

### Small models

If it is known a-priori that a model containing few terms is to be preferred then a bias may be introduced into equation 2.3, giving

$$p'(\text{model } k) = \frac{1}{Z'} \exp \left( -\frac{1}{T} (\text{SSE}(k) + \beta f(k)) \right) \quad (2.6)$$

where  $f(k)$  is the number of model terms included in model  $k$ . It can be seen that this will increase the discrimination between wanted and unwanted terms, although this is difficult to quantify, depending on the relative sizes of  $\Delta\varepsilon$  and  $\beta$ . After the transient phase the

---

<sup>2</sup>To the author’s knowledge there is no straightforward closed form solution for the exact probability value, the simplest expression found being too complex to be of practical use for this purpose.

desired terms will still be included in all models. The unwanted terms, however, will be excluded more often than they are included. Now in the equations for the mean and variance of the occupancy of state one, (equations D.1 and D.2),  $a < 1/N$  and  $b > 1/N$ . For example if  $a = 1/2N$ ,  $b = 2/N$ , the mean is now  $(n_2 + 1)/5$  and the variance is given by

$$v_1^v(n_2) \simeq n_2 \frac{16N - 20}{125} - \frac{N^2 - 23N + 20}{125} \quad (2.7)$$

and clearly a much smaller value of  $n_2$  will be required to ensure that discrimination is achieved.

## Experimental Results

### Circle problem

Figure 2.6 illustrates a circle classification problem, where the training examples are points on circles of radii 0.5 and 1, corrupted by additive gaussian noise with variance 0.08. The discriminant function to be learned is clearly a circle (or ellipse) situated between the two classes. The  $V(2, 3)$  expansion gives the following extended vector

$$\hat{\mathbf{x}} = [1 \ x_1 \ x_2 \ x_1^2 \ x_2^2 \ x_1x_2 \ x_1^3 \ x_1^2x_2 \ x_1x_2^2 \ x_2^3]^T$$

This has  $N = 10$  elements. By equation 2.4, the transient phase will not have been completed with probability  $< 0.001$  after  $n_1 = 60$  iterations, and by equation 2.5, a further  $n_2 = 100$  iterations will give good discrimination between the wanted and unwanted terms. The unwanted terms will be distributed around a frequency of inclusion of 50 with a standard deviation of  $\sim 15$ . Figure 2.7 shows the frequency of inclusion of each of the expansion terms for  $T = 0.3$ . The correct terms are identified, giving a discriminant function of  $d = w_0 + w_1x_1^2 + w_2x_2^2$ .

### Fifth order expansion problem

The algorithm is particularly applicable to large expansion orders due to the elimination of large matrix inverses. Two class data was generated from

$$f = 2x_1x_2x_3^3 + 4x_2^2x_3^2 - 7x_1x_3x_5 + 7x_2x_5 - 5x_3^4 + x_4$$

Values for  $x_1, x_2, x_3$  and  $x_5$  were drawn from a zero mean gaussian with unit variance, and  $x_4$  was then chosen such that  $f = 1$  for a class 1 vector and  $f = -1$  for a class 2 vector. A  $V(5, 5)$  expansion was used. This has  $N = 252$  terms. The first  $n_1 = 1000$  iterations were neglected, giving a probability  $< 0.01$  that the transient phase had not been completed,

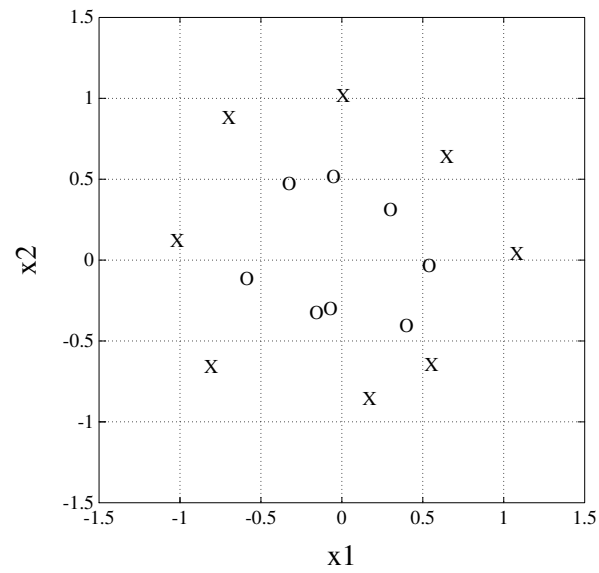


Figure 2.6: Training data for the circle problem

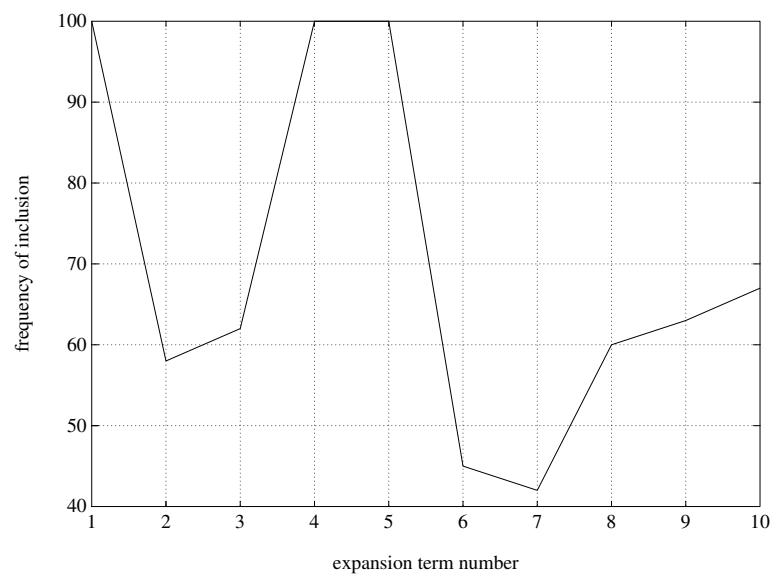


Figure 2.7: Frequency of inclusion of extension terms for circle problem

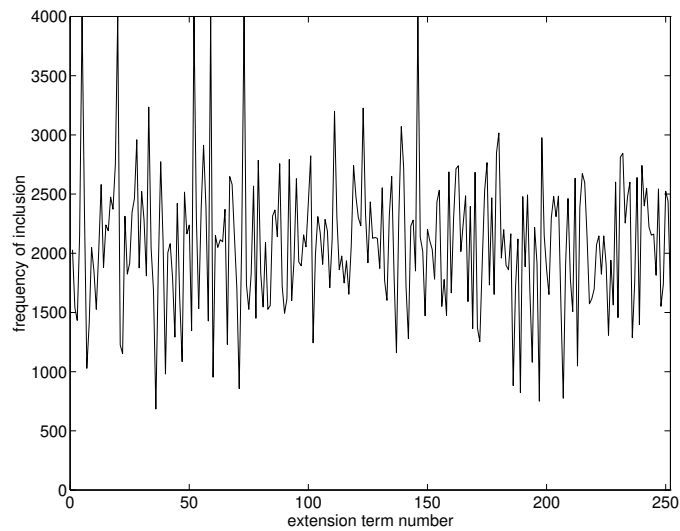


Figure 2.8: Frequency of inclusion of extension terms for fifth order expansion problem,  $\beta = 0$

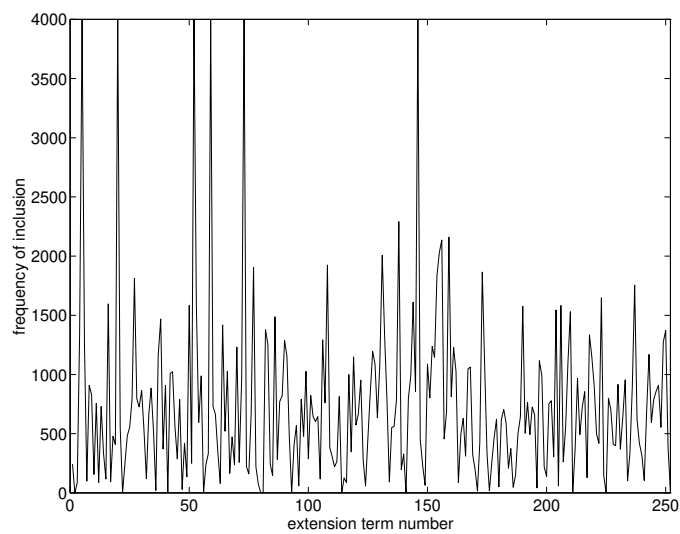


Figure 2.9: Frequency of inclusion of extension terms for fifth order expansion problem,  $\beta = 5/3$

and the chosen models were recorded for the subsequent  $n_2 = 4000$  iterations. This gives a standard deviation of  $\sim 500$ . Figure 2.8 shows the frequency of inclusion of each term for  $T = 1$ ,  $\beta = 0$ . The correct terms are included in every model. The incorrect terms are distributed around a frequency of inclusion of 2000, and none are included more often than 3300 times, approximately 2.6 standard deviations above the mean.

Because it is known that only a few of the expansion terms are required, the acceptance probabilities can be calculated on the basis of equation 2.6. Using  $T = 1$ ,  $\beta = 5/3$ , and the same values of  $n_1$  and  $n_2$  as above gives the frequencies of inclusion graph of figure 2.9. It can be seen that the mean value of the unwanted terms inclusion frequency is much reduced, and that the discrimination between the wanted and unwanted terms has increased. Values of  $T$  were  $\beta$  are chosen from exploratory experiments.

## 2.5 Discussion

So far in this chapter we have discussed the Gibbs sampler and the Metropolis-Hastings algorithm for discrete state spaces. When the state space is discrete, simple proofs of convergence of the two algorithms are available, and were presented in sections 2.3 and 2.4. When the state space is continuous, however, the corresponding proofs are more difficult. In [82] a reasonably straightforward proof of convergence for the continuous state space case is given.

These proofs of convergence, whilst guaranteeing theoretically that the chain will converge to the correct invariant distribution, do not give any indication of how rapid that convergence will be.

There are two approaches in the literature to dealing with this problem. A number of authors have developed theoretical rates of convergence. Roberts [81] has developed expressions for the parameters in the relation

$$\|f_n - \pi\|_i \leq M\rho^n$$

where  $f_n$  is the probability distribution of the chain at iteration  $n$ ,  $M < \infty$  and  $\rho$  is related to the transition kernel of the chain.  $\|\cdot\|_i$  denotes the  $L^i$  norm, where  $i$  was either 1 or infinity. Rosenthal [84] provides similar results for the total variation distance from  $f_n$  to  $\pi$ .

The second approach is to monitor the output of the chain, and to use some function of the realisation to determine convergence. As noted by Green [39] these must be used with some caution, as it is difficult to infer anything about parts of the distribution which

have not been visited, without a reasonable knowledge of the target distribution. In [77] a method was developed which reduces the chain's realisation to a binary sequence dependent on some function of the output. This binary sequence is then modelled as a two state Markov chain, and expressions for the length of 'burn in' required, and the subsequent number of iterations needed for results of a specified accuracy are developed. A second method of monitoring convergence is presented in [21]. This method monitors 'weights', defined as

$$w(\mathbf{x}) = \frac{\pi(\mathbf{x})}{f_n(\mathbf{x})}$$

where  $f_n(\mathbf{x})$  was approximated from the chain's kernel. At convergence the distribution  $w$  should be a delta function. The presence of outliers in this distribution is an indication that the chain has not converged.

Once convergence is established, time series methods based on the autocorrelation properties of the chain [99] or on 'batch means' [35] can be used to determine the uncertainty introduced into any estimates caused by the use of a finite number of samples.

A number of other practical considerations are important. In [31] the case was made for using multiple chains, initialised from an overdispersed distribution, to ensure that the distribution was adequately explored. This has the disadvantage that the burn in samples must be ignored for each chain. In [50] the use of coupled chains, running at different temperatures, was proposed, where greater mobility through the distribution was achieved by proposing swaps between the states of the different temperature chains.

The determination of the proposal distribution for the Metropolis-Hastings algorithm is also a focus of attention. A number of strategies are discussed in [99]. For the random walk Metropolis algorithm, where  $q(\mathbf{x}, \mathbf{x}') = q(\mathbf{x} - \mathbf{x}')$ , if the variance of the proposal distribution is small, most proposals will be accepted, but the chain will move very slowly. Conversely, a large proposal variance will result in very few accepted changes. In [30] it was shown that for a particular target distribution of moderate dimension it was optimal to adjust the proposal distribution such that approximately 25% of the proposals were accepted. This is a useful rule of thumb in many other cases.

## 2.6 Summary

In this chapter we have explained the rationale behind the use of Markov chain Monte Carlo simulation techniques. The Markov chain theory necessary for the development of the Markov chain Monte Carlo techniques was presented, and two algorithms, the Gibbs sampler and the Metropolis-Hastings algorithm were detailed. Examples of the use of each

---

algorithm were given, taken from the areas of Bayesian inference, where we showed how marginal densities could be estimated easily and accurately, and from the area of artificial neural networks, where we showed that the Metropolis-Hastings algorithm can form the basis for an efficient, effective self-structuring algorithm. Some implementation issues were also discussed.

---

## MARKOV RANDOM FIELDS AND SCRATCH REMOVAL FROM MOTION PICTURES

---

In this chapter we present the theory of Markov Random Fields (MRFs), and show how this theory can be used to construct algorithms for the detection of ‘blotches’ in motion pictures, and the subsequent interpolation of the detected areas. In section 3.2 Markov Random Field image models are discussed, and a number of examples of the images particular MRF specifications produce are given. In section 3.3 a review of previous work into image sequence restoration is presented, concentrating on attempts to remove scratches from motion picture material. We then construct an MRF which enables blotches to be detected, and show how the optimum detection field may be estimated by stochastic methods and deterministic approximations. In section 3.3.4 an MRF is developed which is matched to the problem of interpolating the detected regions. Again its optimisation by stochastic and deterministic methods is presented. For both the detection and interpolation algorithms qualitative, pictorial results are presented, and quantitative comparisons made.

### 3.1 Introduction

The work in this chapter concerns methods for detection and removal of one major type of defect in motion picture material. Due to the physical nature of the film material it becomes dirty very easily, and rough handling can cause the emulsion to be abraded. When dirt, fingerprints or hairs, for example, become attached to the film material they result in a dark defect; abrasion of the emulsion results in a bright defect. Together these defects are known as ‘Dirt and Sparkle’, ‘scratches’ or ‘blotches’. Figure 3.1 shows a frame which has been degraded with artificial, but realistic looking blotches. Clearly this type of degradation is local in nature, and so any processing method developed should respect this, and leave undegraded areas of the frame untouched - the quality and detail in undegraded areas of many black and white films is often very high. Later in this chapter algorithms will be given which address this problem, and these will be compared with existing solutions.

The algorithms have their basis in *Markov Random Field* (MRF) image models. It is widely recognised that image and image sequence processing algorithms perform more satisfactorily if they have a solid foundation in a mathematical image model. Image models



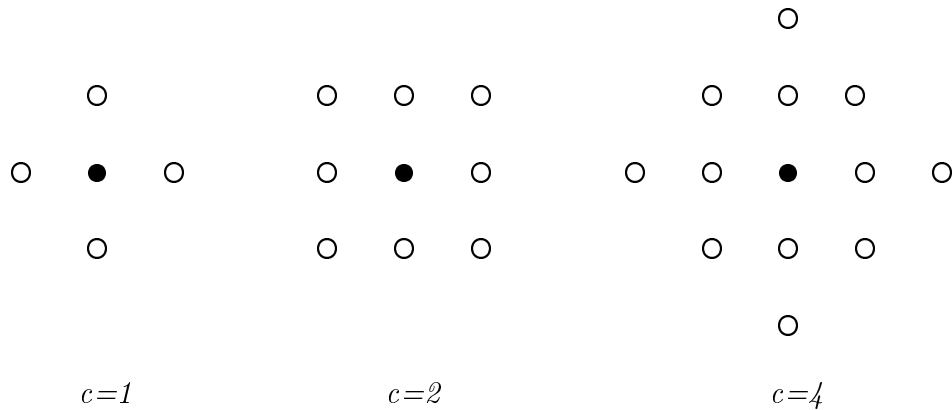
Figure 3.1: Degraded frame from sequence

in use vary from the simple models implicit in processes like Wiener filtering, to autoregressive models, which have been used with success for interpolation and restoration tasks [52, 65]. The MRF model is extremely flexible, indeed much of its power will not be used in this chapter, but will only be called upon in chapter 4. We now discuss the aspects of MRF models and associated optimisation techniques, which draw on the Markov chain Monte Carlo algorithms of chapter 2, which are used later for the scratch detection and interpolation algorithms.

## 3.2 Markov Random Field Image Models

Markov Random Field (MRF) models provide a very rich framework for modelling images and other spatial systems [7, 80]. Their more widespread use has come about since the paper by Geman and Geman [33], which drew together many ideas from the statistical and statistical physics literature in an expository manner, and proved a number of important theorems. In this section we present some of the main ideas, and in later sections show how these can be used for scratch detection and removal in motion pictures.

MRF models define a probability distribution over a set of interacting variables  $\{x_i\}$ , which, in this work, are taken as lying on the points of a regular lattice, defining the pixel grid of the image. Let  $S$  denote this set of sites. Each variable thus represents the grey level, or some other quantity of interest, at one pixel. The model then details the interactions between the pixel values in terms of local conditional probabilities, and assigns higher probabilities to images with characteristics which match the model. A number of concepts are fundamental to the use of MRFs, namely neighbourhoods, cliques, the Hammersley-Clifford theorem, Gibbs distributions and potential functions [6, 33].

Figure 3.2: Neighbourhoods of order  $c = 1, 2, 4$ 

### 3.2.1 Neighbourhoods

Neighbourhoods on the pixel lattice are defined in the following way. The set of neighbours of lattice point  $i$  is those points  $j$  such that the functional form of the local conditional probability,  $p(x_i | x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$ , depends only on  $x_j$ ; we also demand symmetry, such that if site  $i$  is a neighbour of site  $j$ , then site  $j$  is a neighbour of site  $i$ . Figure 3.2 illustrates a few neighbourhood structures of different orders, where the  $c^{\text{th}}$  order neighbourhood of site  $i$  is defined as  $\mathcal{N}_i^c = \{j \in S : 0 < |i - j|^2 \leq c\}$ . Clearly these must be modified at the image boundaries. Either toroidal or free boundary conditions may be used; the second of these being more natural.

### 3.2.2 Cliques

A clique is defined to be a set of sites such that the set contains either a single site, or every site in the set is a neighbour of every other site. Figure 3.3 shows the pair cliques for the first-order neighbourhood system, and some additional cliques available with the second order neighbourhood. Clearly the number and types of cliques grows very rapidly with neighbourhood size, and allows almost unlimited forms of interaction between pixels in a neighbourhood to be specified.

As stated above, MRF models are built by specifying the local conditional probability structure - how pixels in a neighbourhood interact - and this is a fairly natural, easy method of modelling. The development of MRF models was held up for some time by the lack of a formulation for specifying these conditional probabilities which leads to a consistent form for the joint probability. The Hammersley-Clifford theorem provides this formulation. Its original proof was developed in 1971, but went unpublished until 1990 [20]. This proof is involved and relies on what the authors call the “blackening algebra”. The following simple proof is due to Besag [6].

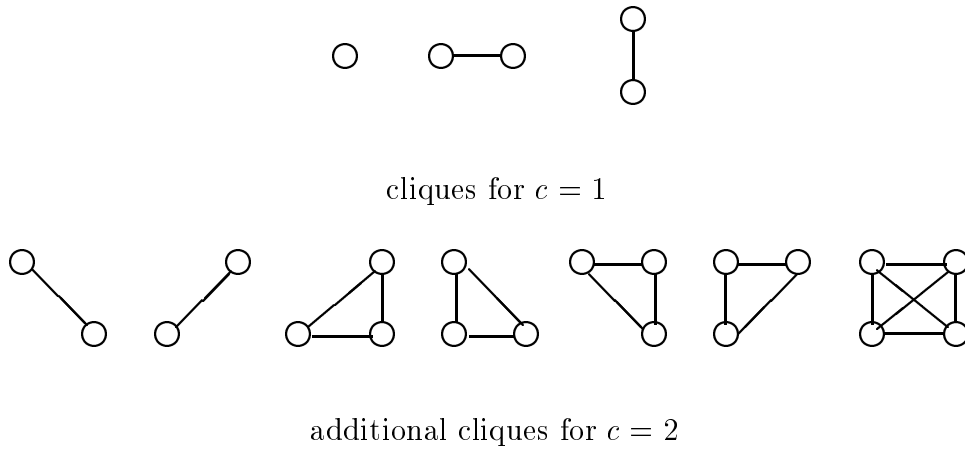


Figure 3.3: Examples of cliques for first and second order neighbourhoods

### 3.2.3 The Hammersley-Clifford Theorem

The Hammersley-Clifford theorem answers the question of consistency between the conditional and joint probability specifications. Assume each variable  $x_i$  takes values from a discrete state-space, of which zero is a member (by relabeling if necessary), and that all configurations have non-zero probability, *i.e.*  $p(\mathbf{x}) > 0$ . Also define  $Q(\mathbf{x}) \equiv \ln(p(\mathbf{x})/p(\mathbf{0}))$  and let  $\mathbf{x}_i$  denote  $(x_1, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n)$ . Determination of the form of  $Q(\mathbf{x})$  gives the form of the allowable conditional probability structure, since

$$\exp(Q(\mathbf{x}) - Q(\mathbf{x}_i)) = p(x_i | x_1 \dots x_{i-1}, x_{i+1} \dots x_n) / p(0 | x_1 \dots x_{i-1}, x_{i+1} \dots x_n) \quad (3.1)$$

$Q(\mathbf{x})$  may be expanded as

$$\begin{aligned} Q(\mathbf{x}) &= \sum_{1 \leq i \leq n} x_i G_i(x_i) + \sum_{1 \leq i < j \leq n} x_i x_j G_{i,j}(x_i, x_j) \\ &+ \sum_{1 \leq i < j < k \leq n} x_i x_j x_k G_{i,j,k}(x_i, x_j, x_k) + \dots + x_1 x_2 \dots x_n G_{1,2,\dots,n}(x_1, x_2, \dots, x_n) \end{aligned}$$

where the  $G$  functions are arbitrary. This can be shown to be a unique expansion by considering suitable choices of  $\mathbf{x}$ . From equation 3.1 the conditional probability specification says that  $Q(\mathbf{x}) - Q(\mathbf{x}_i)$  can only depend on the values of  $x_i$  and its neighbours. Considering just site 1 we have

$$\begin{aligned} Q(\mathbf{x}) - Q(\mathbf{x}_1) &= x_1 \left[ G_1(x_1) + \sum_{2 \leq j \leq n} x_j G_{1,j}(x_1, x_j) \right. \\ &\quad \left. + \sum_{2 \leq j < k \leq n} x_j x_k G_{1,j,k}(x_1, x_j, x_k) + \dots + x_2 x_3 \dots x_n G_{1,2,\dots,n}(x_1, x_2, \dots, x_n) \right] \end{aligned}$$

Now if site  $l$  is not a neighbour of site 1,  $Q(\mathbf{x}) - Q(\mathbf{x}_1)$  must be independent of  $x_l$ . Consider  $x_i = 0$  for  $i \neq 1, l$  and hence  $G_{1,l}(x_1, x_l) = 0$ . Other choices of  $\mathbf{x}$  show that the higher order  $G$  functions involving both  $x_1$  and  $x_l$  must be null. Consideration of any non-neighbour pair gives analogous results. Hence  $G_{i,j,\dots,s}$  is only non-null if  $i, j, \dots, s$  form a clique and we have the theorem:

For any  $1 \leq i < j < \dots < s \leq n$  the function  $G_{i,j,\dots,s}$  in the expansion of  $Q(\mathbf{x})$  may be non-null if and only if the sites  $i, j, \dots, s$  form a clique. Subject to this restriction the  $G$  functions may be arbitrary.

Clearly any joint distribution may be constructed if the neighbourhoods are made large enough.

### 3.2.4 Gibbs Distributions and Potential Functions

The allowable forms of  $Q(\mathbf{x})$  as determined above allows the joint distribution to be written in the form of a Gibbs distribution [33]. Thus

$$p(\mathbf{x}) = \frac{1}{Z} \exp\left(-\frac{1}{T}U(\mathbf{x})\right) \quad (3.2)$$

where  $Z$ , the partition function which normalises the distribution, and  $T$  are constants.  $U(\mathbf{x})$  is known as the *energy function*, and is of the form

$$U(\mathbf{x}) = \sum_{C \in \mathcal{C}} V_C(\mathbf{x}) \quad (3.3)$$

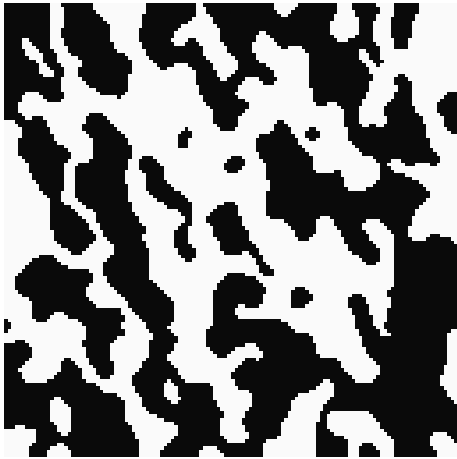
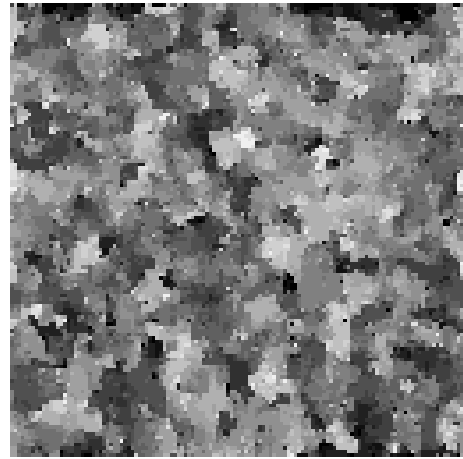
where the sum is over all the cliques on the lattice and  $V_C(\mathbf{x})$  is an arbitrary *potential function*, depending only on those variables  $x_i$  within the clique  $C$ .

Thus an MRF model is specified through the joint probability distribution as follows:

$$\begin{aligned} p(\mathbf{x}) &> 0 \\ p(x_i|x_j, j \neq i) &= p(x_i|x_j, j \in \mathcal{N}_i) \end{aligned}$$

and practically is built up hierarchically in the following manner.

1. Specify a neighbourhood system.
2. Determine the cliques associated with that neighbourhood system.
3. Specify potential functions associated with each clique. These may be position independent, leading to an homogeneous model, or position dependent.

Figure 3.4: Example of Ising Model,  $\beta = 2$ Figure 3.5: Example of Mixed Model,  $c = 1, \beta = 3, \delta = 2, 16$  states

$$4. \text{ Form } p(\mathbf{x}) = \frac{1}{Z} \exp\left(-\frac{1}{T} \sum_c V_C(\mathbf{x})\right),$$

The main modelling stages are 1 and 3, the choice of neighbourhood size and the choice of potential functions. By careful choice of these aspects of the model we can ensure that in stage 4 higher probabilities are assigned to configurations of the field (that is, images) with the desired characteristics. We thus have a model in the form of a complex probability distribution defined over as many variables as there are pixels in the image. The problem is now how to perform any computation with such a model, and the answer lies in the Markov chain Monte Carlo methods of chapter 2.

The conditional probability of any variable is, by construction, only a function of the fixed values of the pixels in its neighbourhood. For small state spaces or particular forms of potential function this distribution may be computed exactly, and hence sampled from easily. This allows straightforward application of the Gibbs sampler to the problem of sampling from an MRF - indeed, the Gibbs sampler as presented in section 2.3 was developed for just this purpose. In fact a stronger result was proved, that convergence is guaranteed for updating the variables in any random order, provided each site is updated infinitely often, allowing parallel, asynchronous updating.

Figures 3.4, 3.5 show samples from two MRFs. The first is the classic Ising model [51], which is defined for  $x_i \in \{-1, +1\}$ ,  $c = 2$  neighbourhood and  $V_C(x_i, x_j) = -\beta x_i x_j$  for two element cliques only. The second is defined over 16 states,  $c = 1$  and has potential function  $V_C(x_i, x_j) = \frac{\beta}{1+|(x_i-x_j)/\delta|}$  [34]. These figures give some indication of the flexibility of MRF image modelling.

### 3.3 Scratch Removal from Motion Pictures

In this section we first review previous work in the area of image sequence restoration, concentrating on work on restoring sequences degraded by scratch type artifacts, and then develop a new algorithm for the detection and subsequent interpolation of scratch defects in motion picture material.

#### 3.3.1 Introduction and Review

Early work in image sequence restoration focused on the problem of white noise removal. The usual approach was some form of temporal averaging. Simple frame averaging works well in stationary parts of the image, indeed noise reduction close to the theoretical improvement of a factor of  $\sqrt{n}$  is achievable, where  $n$  is the number of frames used in the averaging operation. In moving areas of the frames, however, simple frame averaging is a poor solution – the moving areas are blurred very badly, and ‘shadows’ may appear.

To counter this effect adaptive schemes were developed. These systems worked by detecting motion, and only applying the frame averaging operation in areas classed as stationary [23]. This reduced the blurring effect described above, but resulted in no noise reduction in the moving areas. This results in a number of visually annoying artifacts, mainly when objects begin to move.

The next stage from detecting motion, and modifying the filtering on the result of motion detection, was explicit motion *estimation* and *compensation*. Once moving objects can be tracked, and filtering performed along motion trajectories, the way is opened for more accurate algorithms. Purely temporal finite impulse response (FIR) and median filters have been applied to the noise reduction problem [49]. The use of the temporal median filter has the advantage that it is robust to motion estimation errors. Motion estimation errors will result in discontinuities when viewed along motion trajectories, and the one dimensional median filter is known to pass step changes.

The next stage in the development of white noise reducing filters was the recognition that once motion compensation has been performed then it is possible to treat the sequence as a statistically stationary three dimensional signal (at least within a small window). The development of the three dimensional Wiener filter is probably the best example. This has been used for noise reduction in motion picture material, exhibiting very general motion [57], and a variant has been used for the restoration of sequences of remote sensed imagery, where there is a single, global translation between successive frames [24].

The area of *median filtering* has also been active. The nonlinear nature of the median

operation enables filters to be designed which will provide noise attenuation without over smoothing small image features. The median filter rejects outliers and so is effective at removing impulsive, or ‘salt and pepper’ noise. Nieminen *et al.* [72] introduced the multilevel median filter. This is a hierarchical arrangement, where the inputs to a final stage median filter are the outputs from earlier median filters. This arrangement has a number of advantages. The filters at the first level can be constructed to have spatial supports which preserve a wide variety of image features of differing orientations. This allows the multistage filter to reject impulsive noise on the image with less smoothing than would result from a single filter with spatial support consisting of the union of the spatial supports of the first level filters.

Three dimensional median filters were discussed in [2, 12, 3]. These filters were not explicitly motion compensated. Instead the first level structures were made insensitive to motion. That is, if the first level filters were applied to moving areas then the pixels in the temporal support would be outliers and hence not be chosen to be passed to the second level filter. This can only be achieved by having a spatial support larger than the temporal support, so that the filtering operation reduces to a spatial one in moving areas of the images. Again there is a trade off between the number of first level filters used, which will enhance the detail preservation, and the amount of noise rejection achievable, which is reduced as more filters are introduced.

Because of the need to have more spatial than temporal support, for robustness against motion, there is a limit to the spatial extent of the distortion that can be rejected by the multilevel median filters discussed so far. Using supports in the current frame which are subsets of a  $3 \times 3$  pixel area centred on the pixel being processed limits the size of distortion which can be rejected to the size of that area.

The problem we are addressing in this chapter is that of restoring ‘scratches’ or ‘blotches’ which are a major defect in archived motion pictures. These can be formed by abrasion removing some of the emulsion, causing a bright defect, or dirt becoming attached to the film, resulting in a dark defect. Other forms of contamination, for example fingerprints, can also result in a shift in brightness.

The common feature of these scratch type defects is that they can be characterised as a *temporal discontinuity*. Given a single frame it is a very high-level processing task to determine which, if any, areas of the scene are scratches, and indeed this may not be possible. However in an image sequence, the scratches and blotches become immediately obvious as areas which ‘do not fit’ – that is, the scratch area does not match previous or subsequent frames, taking motion between the frames into account.

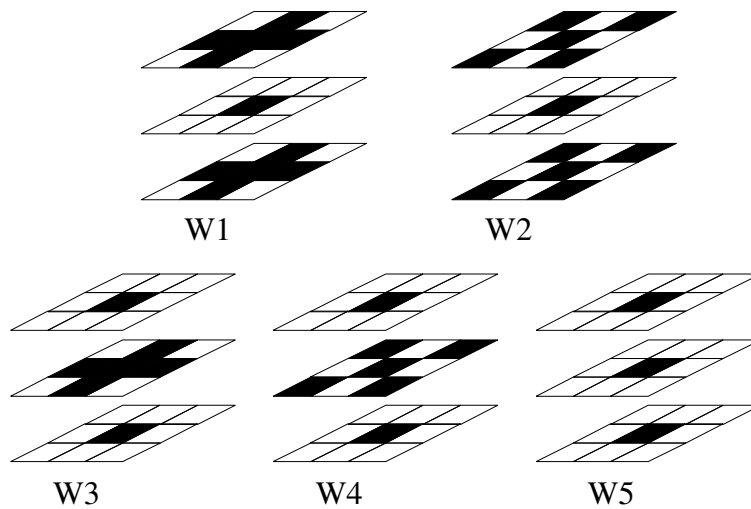


Figure 3.6: Spatio-temporal supports used in Kokaram's multilevel median filter

This provides the heuristic for the detection of scratches and blotches used in most of the detectors we will now discuss – *a scratch is a spatially connected area of a frame which is unpredictable from either the previous or following frames*. We will show later that the use of the theory of MRFs allows the spatial continuity of the scratch to be included in the detection algorithm, something not done explicitly by any of these earlier approaches.

Kokaram [56] proposed using explicit motion compensation with the filter structures used so far to enable greater fidelity to be achieved, due to the increased occurrence of good information in the temporal supports. Once motion compensation was being performed, Kokaram then realised that it was reasonable to include much more information from the surrounding frames. This would enable distortion of appreciable spatial extent to be rejected; this is the first example of a motion compensated filter being designed for the scratch removal problem. The supports used in the first level median filters were as shown in figure 3.6. The inputs to the second level median were the outputs of these filters.

So far all the filters discussed have been applied globally. The contamination under consideration (scratches or blotches) is, however, local in nature. Areas of the images not affected are often of very high definition, and it is thus desirable not to filter these areas, as filtering will inevitably introduce some distortion. In [95, 96] a detector for this type of contamination was introduced.

This detector worked by checking whether the observed grey level value lay within the range suggested by the previous and following frames. The detection equation is

$$\begin{aligned}
 D_{\text{BBC}} &= 1 \text{ if } (|f| > t) \text{ AND } (|b| > t) \text{ AND } (\text{sign}(f) == \text{sign}(b)) \\
 &= 0 \text{ otherwise}
 \end{aligned}$$

where  $f$  is the difference in the forwards direction,  $b$  the difference in the backwards direction and  $t$  a threshold.  $D_{\text{BBC}} = 1$  indicates the presence of a blotch. No motion compensation was used with this detector, due to hardware limitations at the time of its development. Instead a number of control measures were used to turn off the detector where it was decided that the frame differences were unreliable. These reduced the detection accuracy; blotches on moving areas could not be detected. In [95, 96] simple averaging of the previous and following frames was used to effect the restoration.

In [60] a similar detector working on motion compensated differences at each pixel was derived. This detector worked by producing a number, the Spike Detection Index (SDI), which could be thresholded to give a detection field. The SDI was defined as

$$\begin{aligned} \text{SDI} &= 1 - \frac{||f| - |b||}{||f| + |b||} \text{ for } |f| > t \text{ or } |b| > t \\ &= 0 \text{ otherwise} \end{aligned}$$

where  $f$  and  $b$  now indicate motion compensated differences. The threshold applied to these differences ensured that the SDI was set to sensible values as  $f$  and  $b$  tended to zero when the motion estimator found a good match. This detector was found to be effective in locating scratches in motion picture material. The multilevel median filter described above was then applied to the detected locations to restore the frame.

In [59] it was recognised that a motion compensated development of the detector in [95] would be the most simple detector possible. The SDIa was defined as

$$\text{SDIa} = (f^2 > t) \text{ AND } (b^2 > t)$$

That is, a scratch is indicated if the motion compensated frame differences in both directions are larger than some threshold. Estimates for this threshold can be made on the basis of the visibility of scratches of differing intensity differences.

Model based approaches to signal processing problems have generally proved more accurate than ad-hoc methods. The autoregressive (AR) model has proved extremely useful in many problems. In its one dimensional form it is a very good model for short segments of audio data, and has been used to good effect for audio restoration problems [38]. The 2-D AR model has been used to interpolate known missing areas of still images [101], and the 3-D AR model has been used to model image sequences. Motivated by the work on audio restoration, Kokaram developed a blotch detector based on the 3-D AR model [59].

If it is assumed that the sequence has been motion compensated, so that motion offset

terms do not appear in the equations, then the 3-D AR model is summarised by

$$\mathbf{x}(\mathbf{i}) = \sum_{k=1}^N a_k \mathbf{x}(\mathbf{i} + \mathbf{q}_k) + \mathbf{e}(\mathbf{i})$$

where  $\mathbf{i}$  indexes the pixel locations. This says that any pixel can be predicted as a weighted sum (weights  $a_k$ ) of pixels in some spatiotemporal neighbourhood (where the  $\mathbf{q}_k$ 's are the spatiotemporal offsets to the support locations). The term  $\mathbf{e}(\mathbf{i})$  is the residual from the model. Usually the support region is predefined, and then the weights are calculated by minimising some function of the residual. In [59] the mean square residual was minimised for ease of computation.

If the image is assumed to be corrupted by large amplitude additive noise, affecting a certain, randomly chosen proportion of the image, then the degraded image is given by

$$\mathbf{x}^d(\mathbf{i}) = \mathbf{x}(\mathbf{i}) + \mathbf{b}(\mathbf{i})$$

where  $\mathbf{b}(\mathbf{i}) = 0$  at the unaffected pixels, and takes large values consistent with the appearance of a scratch at the degraded pixels. Filtering  $\mathbf{x}^d$  with the model prediction filter gives

$$\begin{aligned} \mathbf{e}^d(\mathbf{i}) &= \mathbf{x}^d(\mathbf{i}) - \sum_{k=1}^N a_k \mathbf{x}^d(\mathbf{i} + \mathbf{q}_k) \\ &= \mathbf{x}(\mathbf{i}) + \mathbf{b}(\mathbf{i}) - \sum_{k=1}^N a_k \mathbf{x}(\mathbf{i} + \mathbf{q}_k) - \sum_{k=1}^N a_k \mathbf{b}(\mathbf{i} + \mathbf{q}_k) \\ &= \mathbf{e}(\mathbf{i}) + \mathbf{b}(\mathbf{i}) - \sum_{k=1}^N a_k \mathbf{b}(\mathbf{i} + \mathbf{q}_k) \end{aligned}$$

Thus the filter output will have small values, on the scale of the model residual, where  $\mathbf{b}(\mathbf{i}) = 0$ , and typically much larger values where the degradation is present. Thresholding  $|\mathbf{e}^d|$  gives the detection field.

Kokaram noted a number of practical considerations when using this detector. If the areas of the image sequence used to calculate the model coefficients contained a large proportion of scratch pixels then the model coefficients would be biased and begin to model the scratches. This limits the extent of the scratches that can be detected. It was also found to be advantageous to use two separate 3-D AR models, one a forwards predictor, and the other a backwards predictor, and to combine the detection fields. Using a single model with support in both directions was more prone to being affected by biased coefficients. Kokaram reported extremely accurate detection when the correct model parameters were used, but

when these parameters were estimated from the image data the detector's performance was found to be inferior to that of the SDIa discussed above.

Kokaram also used the 3-D AR model to interpolate the detected areas. Because the scratch locations had already been estimated, much of the bias could be removed from the coefficient estimation stage, by using a weighted error criterion. This removed any contribution to the error term from sites labelled as scratches. The main fault is then that the image data may not be stationary over the area used to estimate the parameters. The restoration was performed by minimising the mean squared error with respect to the interpolated pixel values. The prediction error can be written as

$$\mathbf{e} = \mathbf{A}_k \mathbf{x}_k + \mathbf{A}_u \mathbf{x}_u$$

and then

$$\mathbf{x}_u = -[\mathbf{A}_u^T \mathbf{A}_u]^{-1} \mathbf{A}_u^T \mathbf{A}_k \mathbf{x}_k$$

where  $\mathbf{A}_u$  and  $\mathbf{A}_k$  are partitions of the AR coefficient matrix corresponding to the known and unknown pixels,  $\mathbf{x}_k$  are the known data values and  $\mathbf{x}_u$  are the unknown values to be interpolated. This interpolator produced extremely good restorations.

Whilst there has been some work on the application of MRFs to image sequence restoration, most of this effort has been directed towards white noise removal (*e.g.* [8]). The only work in the literature which applies MRFs to the problem of restoring motion picture material is that of Geman *et al.* [32]. The algorithm developed in that paper does not use the detection and interpolation approach, but rather constructs an energy function which is a form of spatiotemporal smoothing. Only very rudimentary motion compensation was used – a single motion vector estimate was used for the entire frame, resulting in some motion artifacts in areas where the motion did not correspond to this estimate.

The energy function defined was

$$\begin{aligned} U(\mathbf{x}, \mathbf{y}) = & \alpha \sum_j \sum_i \phi(x_{i+2,j} - 2x_{i+1,j} + x_{ij}) \\ & + \alpha \sum_j \sum_i \phi(x_{i,j+2} - 2x_{i,j+1} + x_{ij}) \\ & + 2\alpha \sum_j \sum_i \phi(x_{i+1,j+1} + x_{i,j} - x_{i+1,j} - x_{i,j+1}) \\ & + \alpha^c \sum_j \sum_i \phi^c(x_{i,j} - y_{i,j}^c) \\ & + \alpha^p \sum_j \sum_i \phi^p(x_{i,j} - y_{i,j}^p) \end{aligned}$$

where  $y^c$  and  $y^p$  are the current and previous observed frames, and  $\mathbf{x}$  is the smoothed output. The potential function,  $\phi(u)$  was

$$\phi(u) = - \left( 1 + \left| \frac{u}{\delta} \right|^\gamma \right)^{-1}$$

with  $\gamma = 1$  for  $\phi(u)$  and  $\gamma = 2$  for  $\phi^c$  and  $\phi^p$ . The remaining parameters were chosen in a manner similar to the one we will use in section 3.3.2. To optimise  $U(\mathbf{x}, \mathbf{y})$  Besag's Iterated Conditional Modes algorithm was used [7]. This involves choosing the value of  $x_i$  which maximises the local conditional distribution each time site  $i$  is visited. To try to avoid artifacts due to this greedy updating scheme the lattice was partitioned into subsets such that no two members of a subset were neighbours, and each subset was updated in turn. The potential function used was chosen as for its finite asymptotic behaviour, which helps to avoid over smoothing. As discussed above, this smoothing approach did leave some visible artifacts in the processed frame.

In the next section we develop an MRF based algorithm for the scratch detection problem. This algorithm employs the same observation as the detectors described above, that scratches are unpredictable from both the previous and the following frames, but also encodes the spatial continuity of the scratch regions. Results comparing this new detector with the best of the previous detectors, the SDIa, are given in section 3.3.3.

It is clear from the discussion above that the detection is reliant on the estimation of motion between the frames. For all the experiments in this chapter a hierarchical block-matching algorithm was used – details can be found in [10, 15] and chapter 4. This algorithm has been found to be robust to the degradations and is able to estimate the large displacements often found in motion picture sequences. The detection and interpolation algorithms of this chapter take a set of motion vectors as an input, their source is of secondary importance. In chapter 4 we will discuss and develop other motion estimation algorithms for this task.

Once the scratches have been detected, the interpolation problem can be viewed as a missing data problem. The observed data at the scratch locations bears no relation to the actual scene information. In Bayesian terminology the likelihood is uniformly distributed over all the allowed values. Hence the interpolation problem is one of signal modelling. The missing areas are interpolated on the basis of an image sequence model. MRF models are used successfully for this problem.

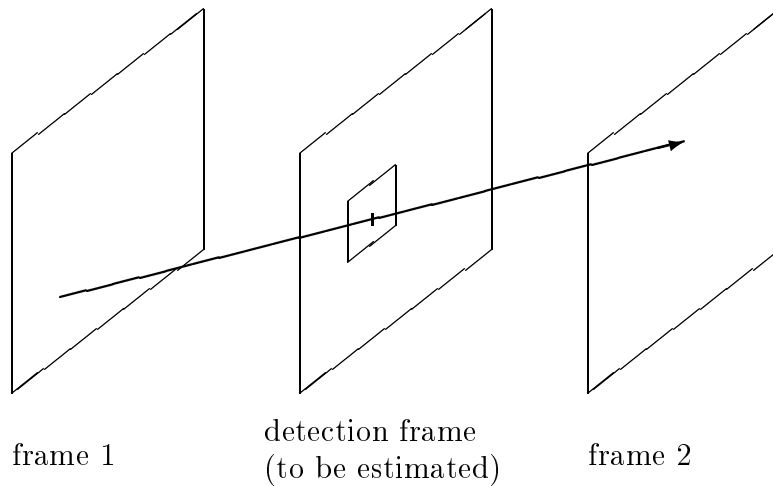


Figure 3.7: Illustration of the detection frame between two image frames

### 3.3.2 An MRF Based Scratch Detection Algorithm

The theory of MRFs allows the spatial continuity of the scratches to be incorporated into the detection algorithm. This section details the specific MRF used, and a number of statistical physics based optimisation techniques which can be used to find the optimum detection.

Consider two adjacent frames from a sequence. Let  $S$  denote the pixel lattice of these two frames taken together, and  $\mathbf{X}$  be the observed grey levels at these lattice points. Let  $\mathcal{N}_i$  denote the neighbourhood of site  $i$  which is within the frame, and  $\mathcal{T}_i$  denote the motion-compensated temporal neighbourhood in the other frame of the pair. Define a discontinuity frame,  $\mathbf{D}$ , between the two image frames, on a lattice  $S'$ , see figure 3.7. Let  $d_i \in \{-1, +1\}$ , where  $d_i = 1$  denotes a temporal discontinuity between the two frames, and where  $d_i = -1$  denotes no discontinuity. It is this frame  $\mathbf{D}$  which is to be estimated. Bayes theorem states that

$$p(\mathbf{D} = \mathbf{d} | \mathbf{X} = \mathbf{x}) = \frac{p(\mathbf{X} = \mathbf{x} | \mathbf{D} = \mathbf{d})p(\mathbf{D} = \mathbf{d})}{p(\mathbf{X} = \mathbf{x})} \quad (3.4)$$

The denominator is a normalising constant and is neglected in this work. The numerator consists of two terms: a likelihood model for the observed frames, given the discontinuity locations, and a prior model for the discontinuity locations. It is this prior model which encodes the spatial continuity of the scratches.

Using a first order ( $c = 1$ ) neighbourhood within the frame, one temporal neighbour, pair cliques, and writing  $\phi(\cdot)$  as the potential function for these pair cliques, the likelihood

function is

$$p_{\mathbf{x}|\mathbf{d}}(\mathbf{X} = \mathbf{x} | \mathbf{D} = \mathbf{d}) = \frac{1}{Z_X} \exp \left( -\frac{1}{T} \sum_{i \in S} \left[ \alpha' \sum_{j \in \mathcal{N}_i} \phi(x_i, x_j) + \alpha(1 - d_i) \sum_{j \in \mathcal{T}_i} \phi(x_i, x_j) \right] \right) \quad (3.5)$$

That is, the probability of observing a particular grey level at location  $i$  is a function of the neighbouring pixel values, with the temporal neighbour being excluded if a discontinuity is indicated.

The prior model on  $\mathbf{D}$  used is the Ising model, augmented with a term to bias towards no discontinuities, to avoid equation 3.4 being maximised by a solution with all  $d_i = 1$ , that is

$$p_{\mathbf{d}}(\mathbf{D} = \mathbf{d}) = \frac{1}{Z_D} \exp \left( -\frac{1}{T} \sum_{i \in S'} \left[ -\beta_1 \sum_{j \in \mathcal{N}_i} d_i d_j + \beta_2(1 + d_i) \right] \right)$$

Combining these equations, and dropping the term from equation 3.5 which is not a function of  $d$  gives

$$p_{\mathbf{d}|\mathbf{x}}(\mathbf{D} = \mathbf{d} | \mathbf{X} = \mathbf{x}) = \frac{1}{Z} \exp \left( -\frac{1}{T} \sum_{i \in S'} \left[ \alpha(1 - d_i) \sum_{j \in \mathcal{T}_i} \phi(x_i, x_j) - \beta_1 \sum_{j \in \mathcal{N}_i} d_i d_j + \beta_2(1 + d_i) \right] \right) \quad (3.6)$$

where the pixels in the temporal term are at positions in  $S$  corresponding to the position  $i$  in  $S'$  and its temporal neighbour. Finding the configuration which maximises this expression solves for the maximum a-posteriori (MAP) configuration of the discontinuity frame. Regions where temporal discontinuities are indicated separately in both the forwards and backwards directions are consistent with the heuristic for scratches and are classified as such.

### Parameter Estimation

This model depends on three parameters,  $\alpha, \beta_1, \beta_2$ , and on the function  $\phi(\cdot)$ . For simplicity  $\phi(x_i, x_j)$  is taken to be  $(x_i - x_j)^2$ . Other choices (*e.g.*  $\phi(u) = |u|$ ) are also suitable. Estimates for the parameter values may be found as follows.

The parameter  $\beta_1$  determines the strength of the self-interaction of the discontinuity field. In the context of restoring binary images a value of around 1 is suggested by Ripley [78], on the basis of considering the conditional probability assigned to a pixel when surrounded by different numbers of pixels in the same state.

Similar considerations can be used to estimate  $\alpha$  and  $\beta_2$ .  $\beta_2$  ‘balances’ the increase in probability due to introducing a discontinuity, which eliminates the effect of the first term in equation 3.6. Thus to balance a difference of  $e_1$  requires

$$\alpha e_1^2 \simeq \beta_2$$

Also consider an isolated pixel with error  $e_2$ . For this to be detected requires

$$\exp(-\beta_2) > \exp(-\alpha e_2^2 + 4\beta_1)$$

In [41] the importance of good parameter estimates for binary classification problems was demonstrated. For this scratch detection problem, by quantifying the heuristic that temporal discontinuities are indicators of blotches, the values of the parameters of the MRF used to detect them may be chosen in a consistent manner.

We now address the problem of finding the optimum configuration  $\hat{\mathbf{d}}$  for the discontinuity frame. As stated above, the configuration which maximises equation 3.6 gives the MAP solution. Other estimates, such as the minimum variance estimate [28, 68] are also suitable, and we shall show that other algorithms approximate this solution.

### Stochastic Simulated Annealing

In equation 3.2  $T$  was said to be a constant. If we allow the value of  $T$  to be varied, then it will alter the ‘peakiness’ of the distribution, and is therefore known as the ‘temperature’ by analogy with physical systems governed by Gibbs distributions [55]. Stochastic Simulated Annealing (SSA) uses this as a control parameter to enable the MAP configuration of the field to be found. For large values of  $T$  the distribution in equation 3.2 is essentially uniform; for  $T \rightarrow 0$  it becomes more and more peaked at the mode.

The Gibbs sampler can be used to draw samples from the distribution of equation 3.2 at any temperature. However the proof of convergence to the invariant distribution in section 2.3 required that the transition probabilities do not change. As the temperature of the distribution is altered the probabilities do change, causing the Gibbs sampler to sample from an inhomogeneous Markov chain. Geman and Geman [33] proved that if the temperature is reduced according to

$$T = \frac{C}{\log(1+k)} \tag{3.7}$$

where  $k$  denotes the number of full scans of the lattice and  $C$  is a fixed constant, then the Gibbs sampler will converge to the uniform distribution over all the configurations

with maximum probability. This enables the mode of the distribution, and hence the MAP estimate to be found - the configuration of the field as the temperature is reduced converges to the MAP estimate. Intuitively, at high temperatures the algorithm may escape local minima, as the distribution is fairly uniform; as  $T \rightarrow 0$  only changes which increase the probability will occur. Geman also showed that the constant  $C = N\Delta U$ , where  $N$  is the number of sites in the lattice, and  $\Delta U$  is the maximum difference in energy function for two configurations which differ at only one site. Clearly this is a huge number, too large to be used in practice. Also the logarithmic schedule in equation 3.7 is very slow, and

$$T = C'a^k, 0 < a < 1$$

is often used [61].

### First-Order Mean Field Approximation

Whilst SSA finds the MAP configuration, it is slow and computationally intensive. The mean field approximation (MFA) provides a deterministic optimisation technique which retains many of the features of SSA. Consider

$$\text{Var}_{\bar{\mathbf{d}}} = \sum_{\mathbf{d}} (\mathbf{d} - \bar{\mathbf{d}})^2 p(\mathbf{d})$$

that is, the variance of the field, given an estimate  $\bar{\mathbf{d}}$ , where the sum is over all configurations of  $\mathbf{d}$ . Minimising this variance gives

$$\bar{\mathbf{d}} = \sum_{\mathbf{d}} \mathbf{d} p(\mathbf{d})$$

which is the mean value of the field [28]. This summation is also over all possible states of the field, and so some simpler approximation must be made.

From equation 3.6 the energy function for the scratch detection problem is

$$U(\mathbf{d}) = \sum_{i \in S'} \left[ \alpha(1 - d_i) \sum_{j \in \mathcal{T}_i} (x_i - x_j)^2 - \beta_1 \sum_{j \in \mathcal{N}_i} d_i d_j + \beta_2(1 + d_i) \right] \quad (3.8)$$

Make the approximation that the influence of  $d_j, j \neq i$  in the calculation of  $\langle d_i \rangle$  is given by the influence of  $\langle d_j \rangle$ , where  $\langle \cdot \rangle$  denotes mean values or expectations [105].

Define a new energy function for site  $i$

$$\begin{aligned} U_i^{mf} &= U(\mathbf{d})|_{d_j=\langle d_j \rangle, j \neq i} \\ &= U_i^{mf'}(d_i) + R_i^{mf'}(\langle d_{S' \setminus \{i\}} \rangle) \end{aligned}$$

which separates the variable and fixed parts of the energy function under this approximation, where

$$U_i^{mf'}(d_i) = \alpha(1 - d_i) \sum_{j \in \mathcal{T}_i} (x_i - x_j)^2 - \beta_1 \sum_{j \in \mathcal{N}_i} d_i \langle d_j \rangle + \beta_2(1 + d_i)$$

and

$$R_i^{mf'}(\langle d_{S' \setminus \{i\}} \rangle) = U(\mathbf{d})|_{d_j=\langle d_j \rangle, j \neq i} - U_i^{mf'}(d_i)$$

Also define

$$\begin{aligned} Z_i^{mf} &= \sum_{d_i} \exp\left(-\frac{1}{T} U_i^{mf}(d_i)\right) \\ &= Z_i^{mf'} \exp\left(-\frac{1}{T} R_i^{mf'}(\langle d_{S' \setminus \{i\}} \rangle)\right) \end{aligned}$$

where

$$Z_i^{mf'} = \sum_{d_i} \exp\left(-\frac{1}{T} U_i^{mf'}(d_i)\right)$$

is the mean field local partition function. The first-order mean field approximation gives

$$\begin{aligned} \langle d_i \rangle &\simeq \frac{1}{Z_i^{mf}} \sum_{d_i} d_i \exp\left(-\frac{1}{T} U_i^{mf}(d_i)\right) \\ &= \frac{1}{Z_i^{mf'}} \sum_{d_i} d_i \exp\left(-\frac{1}{T} U_i^{mf'}(d_i)\right) \end{aligned} \quad (3.9)$$

Where in each of the last three equations the sum is formed over all possible values of  $d_i$ . This approximation thus results in an estimate formed by finding the mean value of each pixel, based on the current values of the pixels in its neighbourhood. This approximate mean value of  $d_i$  can be computed easily on the basis of the values  $\langle d_j \rangle$  in its neighbourhood, which are used to form the local conditional distribution. Especially for small state-spaces and small neighbourhoods the sum involved in equation 3.9 in calculating  $\langle d_i \rangle$  is over a small number of terms, and is not computationally intensive to calculate.

Typically for binary problems, such as the scratch detection problem, a zero initial field is chosen, and equation 3.9 is calculated for all  $i \in S'$ . The temperature is reduced, and

the iteration repeated. A similar cooling schedule as for SSA is usually used, but this mean field approximation often converges much more quickly than SSA, but with possibly poorer results. The values of  $\langle d_i \rangle$  computed from equation 3.9 are continuous valued, and are thresholded to give the final detection field.

### An Improved Mean Field Approximation

In this section the energy function of equation 3.8 is approximated by some much simpler energy function, one for which the state with maximum probability is trivial to find. For a binary system the energy function

$$U_0(\mathbf{d}) = - \sum_i m_i d_i \quad (3.10)$$

is convenient. Clearly  $U_0(\mathbf{d})$  is minimised for  $\mathbf{d} = \mathbf{m}$ . The remaining problem is how to choose the parameters  $m_i$  so that the energy function  $U_0(\mathbf{d})$  best approximates  $U(\mathbf{d})$  in equation 3.8 in some sense [45, 11, 1].

From equation 3.10

$$p_o(\mathbf{d}) = \frac{1}{Z_0} \exp \left( \frac{1}{T} \sum_i m_i d_i \right)$$

where

$$\begin{aligned} Z_0 &= \sum_{d_1 \dots d_N = \pm 1} \exp \left( \frac{1}{T} \sum_i m_i d_i \right) \\ &= 2^N \prod_i \cosh \left( \frac{m_i}{T} \right) \end{aligned}$$

$N$  being the number of sites in the lattice  $S'$ . Define

$$\langle A \rangle_m = \sum_{d_1 \dots d_N = \pm 1} A p_0(\mathbf{d})$$

Consider

$$Q = -\frac{1}{T}(U - U_0)$$

From the Taylor expansion

$$\langle \exp(Q) \rangle_m \geq \exp(\langle Q \rangle_m)$$

giving

$$-T \ln Z \leq -T \ln Z_0 + \langle U - U_0 \rangle_m \quad (3.11)$$

In statistical mechanics terms this is the Gibbs-Bogoliubov-Feynman bound [19], and clearly  $U_0$  best approximates  $U$  when the right hand side of equation 3.11 is minimised, such that the tightest bound is obtained, that is  $\nabla_m[-T \ln Z_0 + \langle U - U_0 \rangle_m] = 0$ .

In equation 3.8 define  $h_i = \beta_2 - \alpha \sum_{j \in \mathcal{T}_i} (x_i - x_j)^2$ , and remove constant terms to give

$$U(\mathbf{d}) - U_0(\mathbf{d}) = \sum_{i \in S'} \left( (h_i + m_i) d_i - \beta_1 \sum_{j \in \mathcal{N}_i} d_i d_j \right)$$

and

$$\begin{aligned} \langle U(\mathbf{d}) - U_0(\mathbf{d}) \rangle_m &= \frac{1}{Z_0} \sum_{d_1 \dots d_N = \pm 1} \left( \sum_i (h_i + m_i) d_i \right) \exp \left( \frac{1}{T} \sum_i m_i d_i \right) \\ &+ \frac{1}{Z_0} \sum_{d_1 \dots d_N = \pm 1} -\beta_1 \left( \sum_i \sum_{j \in \mathcal{N}_i} d_i d_j \right) \exp \left( \frac{1}{T} \sum_i m_i d_i \right) \end{aligned}$$

The first of these summations is straightforward. The second is more involved, but, because of the symmetry of the neighbourhood structure,  $i \in \mathcal{N}_j \Leftrightarrow j \in \mathcal{N}_i$ , terms involving  $d_i d_j$  occur in pairs and the summation can be performed, giving

$$\begin{aligned} -T \ln Z_0 &\leq -T \ln \left( 2^N \prod_i \cosh \left( \frac{m_i}{T} \right) \right) + \sum_i (h_i + m_i) \tanh \left( \frac{m_i}{T} \right) \\ &- \frac{\beta_1}{2} \sum_i \sum_{j \in \mathcal{N}_i} \tanh \left( \frac{m_i}{T} \right) \tanh \left( \frac{m_j}{T} \right) \end{aligned}$$

and

$$\frac{\partial}{\partial m_i} (-T \ln Z_0 + \langle U - U_0 \rangle_m) = \frac{1}{T} \left[ 1 - \tanh^2 \left( \frac{m_i}{T} \right) \right] \left[ h_i + m_i - \frac{\beta_1}{2} \sum_{j \in \mathcal{N}_i} \tanh \left( \frac{m_j}{T} \right) \right] \quad (3.12)$$

The optimum parameters of the approximation can be found by gradient descent using these gradients. Again equation 3.12 is used to minimise the bound in equation 3.11 at a series of temperatures associated with an annealing schedule. At each reduced temperature the converged result of the minimisation at the previous temperature is used as the starting point. The final result is thresholded to give the detection field.

### 3.3.3 Detection Results

To obtain an intuitive feel for the operation of the detector it was applied, with SSA optimisation, to the test pattern shown in figure 3.8. A sequence consisting of uniform grey level frames was constructed, and the test pattern was inserted in place of one of the

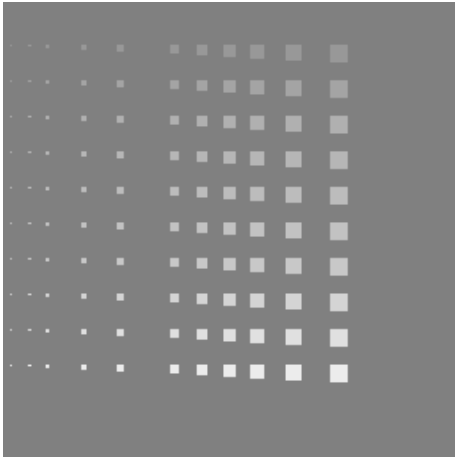


Figure 3.8: Test pattern showing discontinuities of size 1 through 10, of grey level difference from 4 to 18 (exaggerated grey scale).

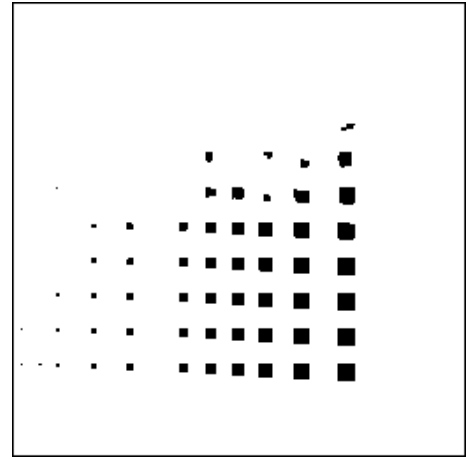


Figure 3.9: Detection frame for  $e_1 = 8, e_2 = 18$

frames. The test pattern shows different sized areas with different grey level differences. Figure 3.9 shows the detection results. The detector can be seen to act as a ‘soft’ threshold – as the size of the blotch increases, the grey level difference needed for its detection decreases. This matches well the visual impact of different sized blotches.

To quantify the action of the detector under the three optimisation schemes the algorithms were applied to an artificially degraded sequence. Figure 3.1 (page 28) shows a frame from the sequence, with artificial scratches added. The scratch locations were generated by running the Gibbs sampler on a biased Ising model for a small number of iterations<sup>1</sup>. Contiguous scratches were then coloured uniformly with a random grey level value. This results in the very realistic looking scratches visible in the figure.

As stated above the motion estimation algorithm used was a multiresolution block matching scheme [10]. Four levels were used, with a full-motion search of  $\pm 4$  pixels at each level. The pyramid was generated using a Gaussian filter with variance one and size  $9 \times 9$  pixels. Motion detection was incorporated as described in [15]. The final estimate was a motion vector at the centre of each  $9 \times 9$  block, and bilinear interpolation was used to produce a vector at each pixel.

The most succinct way of characterising the comparative performance of the detectors is by the use of a plot of Correct Detection rate vs False Alarm rate. This is shown in figure 3.10, where the rates are averaged over twelve frames from the sequence. A number of points relating to this plot need to be made. First, for this problem the two approximate optimisation schemes provide accurate approximations, their responses being very close

<sup>1</sup>The practice of using a large value of  $\beta$ , resulting in large correlations between neighbouring pixels, and running only a few iterations of the Gibbs sampler to compensate, is common.

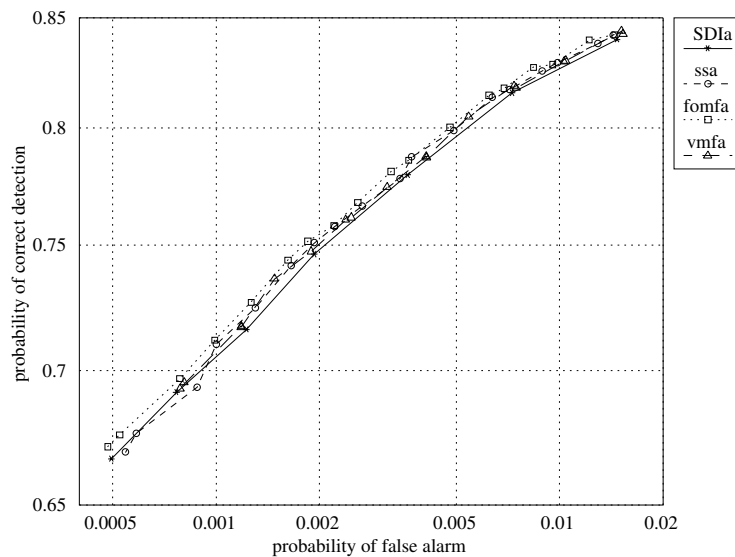


Figure 3.10: Detector operating characteristics – asterisk, SDIa detector; circle, MRF detector using SSA optimisation; square, MRF detector using first order mean field annealing optimisation; triangle, MRF detector using the improved mean field annealing optimisation.

to that of the Simulated Annealing optimisation. The solid curve on the graph indicates the characteristic of the ‘SDIa’ detector which was found to be the most accurate of the previous detectors [59]. This detector flags a pixel if the forwards and backwards motion compensated frame differences are both larger than some preset threshold.

It is seen that the new detector produces some improvement over this detector, but that the improvement is quite limited. Closer examination of equation 3.6 reveals why this is the case. If the term  $-\beta_1 \sum_{j \in \mathcal{N}_i} d_i d_j$  is neglected in equation 3.6, then the distribution becomes a simple product of terms, one for each pixel, and depending on the values of  $\alpha$ ,  $\beta_2$ , and the form of  $\phi(x_i, x_j)$  each term will be maximised by a particular value of  $d_i$  – this corresponds exactly to thresholding the motion compensated frame difference. The additional term in equation 3.6 will only become important if the other two terms are very closely balanced. This will happen for scratches of small grey level difference. Clearly these will only make up a small proportion of the total number of scratches, and so the improvement gained by the incorporation of the spatial continuity will be small. For a frame where the number of scratches with small grey level differences is large, however, the new detector will show a significant improvement.

Figure 3.11 shows the detector operating characteristics when the detectors are applied to frame 49 of the sequence. This demonstrates the improvement possible with the new algorithm when a number of scratches with small grey level difference are present. Also shown on figure 3.11 are operating characteristics for the autoregressive model based detectors described in [56]. These detectors work by thresholding the prediction error resulting

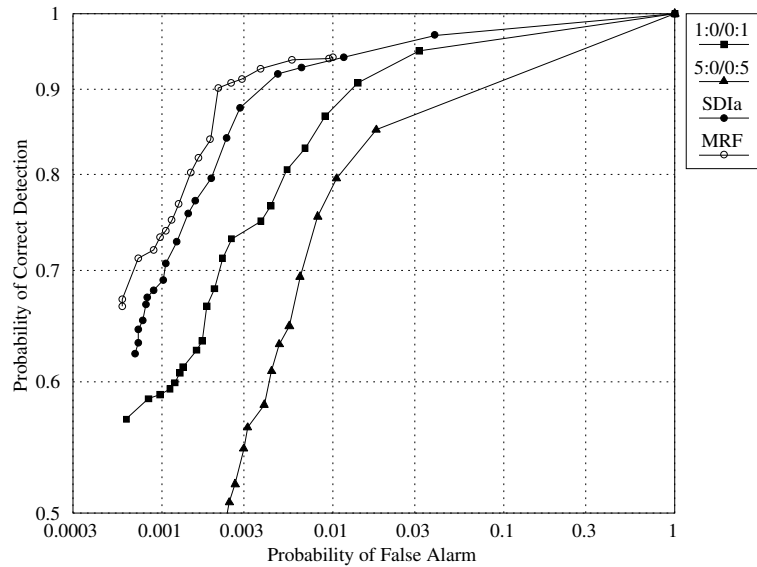


Figure 3.11: Detector operating characteristics - frame 49 – solid square, AR detector, one pixel support; solid triangle, AR detector, five pixels support; solid circle, SDIa detector; circle MRF detector, SSA optimisation.

from various supports in the previous and next frames (one pixel in each of the neighbouring frames (solid square on figure 3.11) and five pixels in each frame (solid triangle on figure 3.11)). The poor results from these algorithms are due to the coefficient estimation process being badly biased by the distortion on the frames. The new detector is seen to be the most effective detector developed.

It is instructive to consider separately the correctly detected scratches, the false detections and the scratches which were not detected. Figures 3.13, 3.14, 3.15 show these three results for parameters determined by  $e_1 = 24$ ,  $e_2 = 30$ .

### Correct detection

The detector correctly identifies almost all the visible scratches. It also identifies some scratches which are almost indiscernible, for example those at the bottom edge of the writing that is partly obscured by the actor's arm, and a region at the top edge of the window. In both these cases the detector finds a connected region as expected.

### False alarms

It is clear from figure 3.14 that most of the false detections occur at the edges of moving regions, for example along the boundary between the actor's arms and the background. This is a result of motion estimation errors, a consequence of the block size used in the block matching algorithm, and the use of only integer accurate motion estimates. The

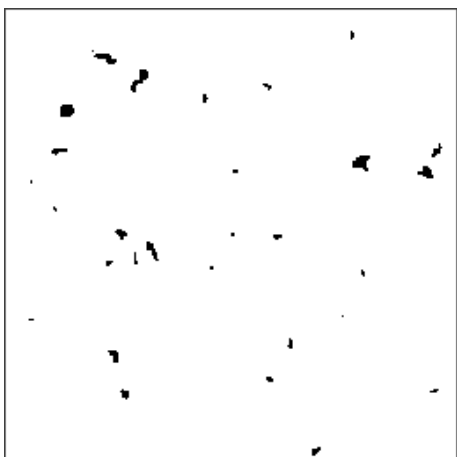


Figure 3.12: Blotch locations

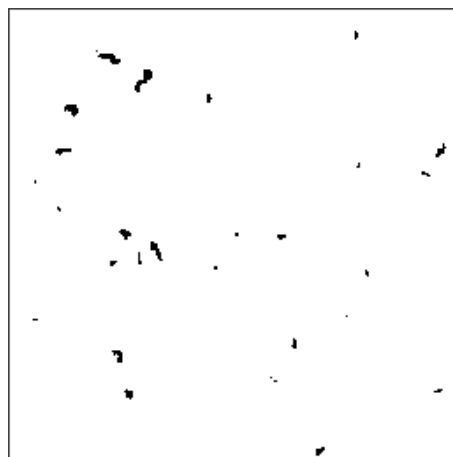


Figure 3.13: Blotches detected correctly

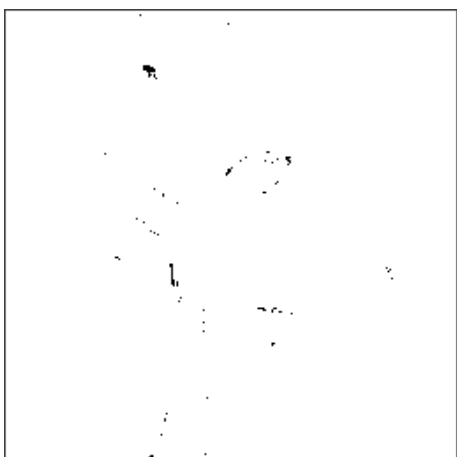


Figure 3.14: False detections

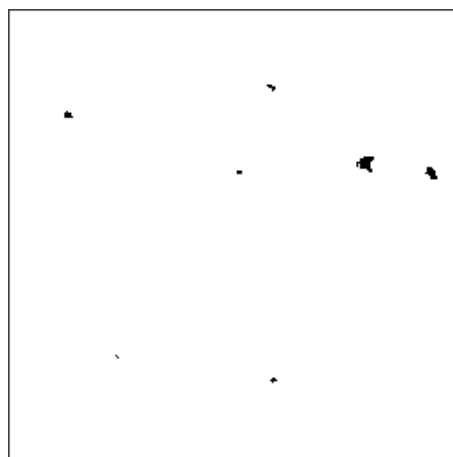


Figure 3.15: Blotches not detected

other main area of false alarms is where the lining of the actor's coat briefly enters the scene as the coat is ruffled. This is an unfortunate effect which is unavoidable with the current heuristic – this area fits the definition of a scratch as an area which has no match in either the previous or next frames. A more complex algorithm incorporating information from more than three frames may be able to overcome this problem.

### Scratches not detected

Close comparison of figures 3.15 and 3.1 shows that almost all the scratches which are not detected are those which are almost the same grey level as the background. This justifies the parameter estimation scheme described in section 3.3.2 for this application.

### 3.3.4 Missing Region Interpolation Using MRFs

As outlined earlier, the problem of interpolating into the detected scratch regions is a missing data problem. If the likelihood function for the frame, given the observed frame and the detection field is examined, it is

$$p(\mathbf{X} = \mathbf{x} | \mathbf{D} = \mathbf{d}) = \prod_{\{i:d_i=-1\}} \delta(x_i)$$

where  $\delta(\cdot)$  is the discrete delta function, so that the undegraded areas of the image are left unmodified, but in the areas marked as scratches, the observed data does not influence the restoration. This is due to the *replacement* nature of the degradation. In the degraded frame, the observed data at the scratch locations gives no information as to the original scene.

To interpolate into the missing regions it is necessary, therefore, to model the image sequence. An image sequence model, conditioned on the image at locations not marked as scratches must be constructed, and used for the interpolation. In this section a simple model based on a spatio-temporal MRF will be introduced for this problem, and details of its application via the Gibbs sampler and the mean field approximations presented. Many image sequence models are in current use [65, 52, 56]. The aim of this section is to show how MRFs can be used to interpolate the detected regions, not to provide a general comparison of image sequence models.

### 3.3.5 An MRF Based Interpolation Algorithm

We now develop the MRF image sequence model which we use for interpolating the areas detected as scratches. Again using a first-order neighbourhood, and pair cliques, the local conditional probability distribution may be written as

$$p(X_i = x_i) = \frac{1}{Z_{X_i}} \exp \left( -\frac{1}{T} \sum_{j \in \mathcal{N}_i} \phi(x_i, x_j) - \frac{\lambda}{T} \sum_{j \in \mathcal{T}_i} \phi(x_i, x_j) \right)$$

where the temporal neighbourhood  $\mathcal{T}_i$  now includes pixels from frames both previous to and following from the current frame. The restriction to a small neighbourhood and simple cliques, whilst not suitable for general modelling of complex images, is suitable for this purpose as the scratch regions tend to be fairly small, and the temporal neighbourhood provides much good image information. Thus the energy function for this problem is

$$U(\mathbf{x}) = \sum_{\{i:d_i=1\}} \left[ \sum_{j \in \mathcal{N}_i} (x_i - x_j)^2 + \lambda \sum_{j \in \mathcal{T}_i} (x_i - x_j)^2 \right] \quad (3.13)$$

where  $\phi(u) = u^2$  is used as the potential function to give a smooth interpolation.

### Interpolation by Gibbs Sampling

The Gibbs sampler can be used to draw a representative sample from a distribution such as that described by the energy function in equation 3.13. This sample is then used as the interpolation. Specifically, samples are drawn from  $p(X_k | X_{S \setminus \{k\}})$ , where  $k$  indexes successive elements from the set  $\{d_i = 1\}$ . Information from the edges of the scratch and from the temporal neighbourhood will propagate into the scratch regions. The restoration will, however, still be a sample from  $p(\mathbf{X} = \mathbf{x})$ , and as such may show unnecessary variations, for example, texture may appear in a uniform region of the image. To overcome this, annealing may be introduced into the Gibbs sampler, to cause the interpolation to converge to the maximum probability solution.

### Interpolation by First-Order Mean Field Approximation

The mean field approximation may also be used with the energy function of equation 3.13 to provide an interpolation of the missing regions.

As in subsection 3.3.2, the influence of  $x_j, j \in \mathcal{N}_i$  in the calculation of  $\langle x_i \rangle$  is approximated by the influence of  $\langle x_j \rangle$ . This results in the mean field energy function becoming

$$U_i^{mf'}(x_i) = \sum_{j \in \mathcal{N}_i} (x_i - \langle x_j \rangle)^2 + \lambda \sum_{j \in \mathcal{T}_i} (x_i - x_j)^2$$

and the estimate of  $\langle x_i \rangle$  is given by

$$\langle x_i \rangle = \frac{1}{Z_i^{mf'}} \sum_{x_i} x_i \exp \left( -\frac{1}{T} U_i^{mf'}(x_i) \right) \quad (3.14)$$

Where the summation is formed over all possible values of  $x_i$ . This is iterated over all  $\{i : d_i = 1\}$ , and the converged result is the interpolation. When  $x_i$  takes discrete values, the values generated from equation 3.14 must be rounded to the nearest discrete value at the end of the iterative procedure.

### An Improved Mean Field Interpolator

In a similar manner to section 3.3.2, in this section the energy function of equation 3.13 is approximated by some simpler energy function. For MRFs defined over a large number of grey levels, which may conveniently be approximated as a continuous state-space, the energy function

$$U_0(\mathbf{x}) = \sum_{\{i:d_i=1\}} (x_i - m_i)^2 \quad (3.15)$$

is a useful approximation [45]. Again the problem is to select the parameters  $m_i$  to minimise the error caused by using  $U_0(\mathbf{x})$  in place of  $U(\mathbf{x})$ .

From equation 3.15

$$p_0(\mathbf{x}) = \frac{1}{Z_0} \exp \left( -\frac{1}{T} \sum_{\{i:d_i=1\}} (x_i - m_i)^2 \right)$$

where

$$\begin{aligned} Z_0 &= \int_{\mathbb{R}^M} \exp \left( -\frac{1}{T} (x_i - m_i)^2 \right) dx_1 \dots x_M \\ &= (\pi T)^{M/2} \end{aligned}$$

and  $M$  is the cardinality of the set  $\{i : d_i = 1\}$ , that is, the number of locations classified as scratches. As in section 3.3.2 the ‘best’ approximation is when

$$\nabla_m [-T \ln Z_0 + \langle U - U_0 \rangle_m] = 0$$

and the calculation of  $\langle U - U_0 \rangle_m$  is now addressed.

$$\begin{aligned} \langle U_0 \rangle_m &= \frac{1}{Z_0} \int_{\mathbb{R}^M} \sum_{\{i:d_i=1\}} (x_i - m_i)^2 \exp \left( -\frac{1}{T} \sum_{\{i:d_i=1\}} (x_i - m_i)^2 \right) dx_1 \dots x_M \\ &= \frac{MT}{2} \end{aligned}$$

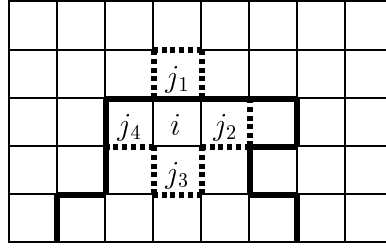


Figure 3.16: Some neighbours of  $x_i$  may be within the scratch area, denoted by the bold line (*i.e.*  $j_2, j_3, j_4$ ), others may not (*i.e.*  $j_1$ ).

$$\begin{aligned} \langle U \rangle_m &= \frac{1}{Z_0} \int_{\mathbb{R}^M} \sum_{\{i:d_i=1\}} \sum_{j \in \mathcal{N}_i} (x_i - x_j)^2 \exp \left( -\frac{1}{T} \sum_{\{i:d_i=1\}} (x_i - m_i)^2 \right) dx_1 \dots x_M \\ &+ \frac{\lambda}{Z_0} \int_{\mathbb{R}^M} \sum_{\{i:d_i=1\}} \sum_{j \in \mathcal{T}_i} (x_i - x_j)^2 \exp \left( -\frac{1}{T} \sum_{\{i:d_i=1\}} (x_i - m_i)^2 \right) dx_1 \dots x_M \quad (3.16) \end{aligned}$$

The second of these terms is straightforward to compute, as the values  $x_j, j \in \mathcal{T}_i$  are fixed known values from the temporal neighbourhood, that is data from previous and subsequent frames. The result of the second integral is

$$\lambda \sum_{\{i:d_i=1\}} \sum_{j \in \mathcal{T}_i} (T/2 + (m_i - x_j)^2)$$

The first term is more difficult due to the self-interaction of the  $x_i$ 's. Also a differentiation must be made between values of  $x_j, j \in \mathcal{N}_i$  which are *within* the scratch region, and hence variable, and those outside the scratch region, (*i.e.*  $\{i : d_i = -1\}$ ), which are fixed known values, see figure 3.16. Again the symmetry of the neighbourhood structure, and the use of the quadratic potential function enables the integral to be evaluated.

Rewrite the first integral in equation 3.16, making a distinction between these two types of neighbours.

$$\frac{1}{Z_0} \int_{\mathbb{R}^M} \sum_{\{i:d_i=1\}} \left[ \underbrace{\sum_{j \in \mathcal{N}_i \cap \{d_i=1\}} (x_i - x_j)^2}_{x_j \text{ within scratch}} + \underbrace{\sum_{j \in \mathcal{N}_i \cap \{d_i=-1\}} (x_i - x_j)^2}_{x_j \text{ not within scratch and of fixed value}} \right] \exp \left( -\frac{1}{T} U_0(\mathbf{x}) \right) dx_1 \dots x_M$$

and this can be evaluated to give

$$\sum_{\{i:d_i=1\}} \left[ \sum_{j \in \mathcal{N}_i \cap \{d_i=1\}} [T + (m_i - m_j)^2] + \sum_{j \in \mathcal{N}_i \cap \{d_i=-1\}} [T/2 + (m_i - x_j)^2] \right]$$

Let

$$R = -T \ln Z_0 + \langle U - U_0 \rangle_m$$

and hence

$$\frac{\partial R}{\partial m_i} = \sum_{j \in \mathcal{N}_i \cap \{d_i=1\}} 4(m_i - m_j) + \sum_{j \in \mathcal{N}_i \cap \{d_i=-1\}} 2(m_i - x_j) + \lambda \sum_{j \in \mathcal{T}_i} 2(m_i - x_j)$$

Again these gradients are used to find the optimum  $m_i$ 's to minimise the errors in the approximation. Once the optimum  $m_i$ 's have been found, the optimum interpolator is that given by  $x_i = m_i$  at the scratch locations.

### 3.3.6 Interpolation Results

To test the action of the interpolator under the three optimisation schemes, the same twelve frames as were used in the detection section were used here. To judge the action of the interpolator only, the interpolators were used to reconstruct the frames at the known scratch locations. Figure 3.17 shows how the mean squared error varies with the optimisation scheme, with a simple frame average interpolator as a benchmark for comparison. The results in this graph are for  $\lambda = 1$ . The strength of the temporal link is a compromise between the desire to include good information from the neighbouring frame, and the possibility of introducing erroneous information due to errors in the motion estimation. Some comments about these results follow.

#### Gibbs sampler interpolation

As was mentioned above, the Gibbs sampler restoration is a *sample* from the distribution describing the reconstructed frame. This will naturally show some variation each time the sample is drawn. The mean squared error will therefore be different each time the restoration is performed. Visual comparison of the restored frames with those restored by the simple frame averager showed a definite improvement in image quality.

#### First-order mean field approximation interpolator

The results for the first-order mean field approximation interpolator were not satisfactory. Their visual quality was poor. This is a result of the approximations not allowing spatial information to propagate into the centre of scratch regions. The large state-space also causes the reconstruction to be badly biased by the initialisation.

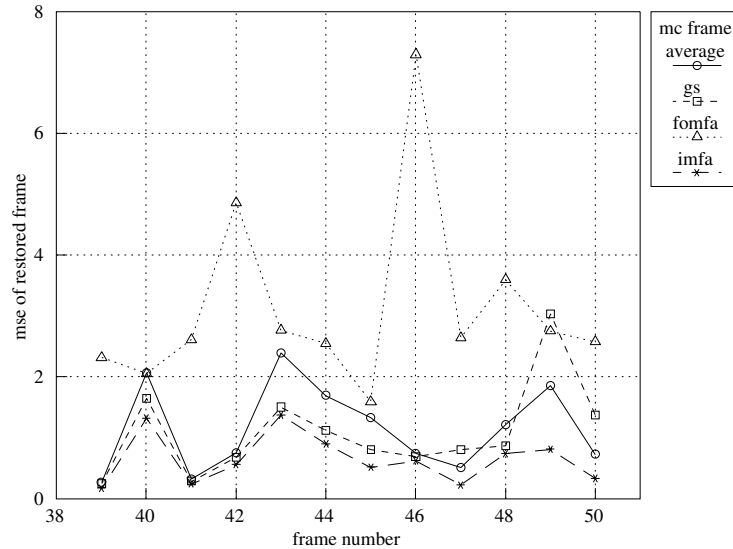


Figure 3.17: MSE for each restoration – circle, motion compensated frame averaging interpolator; square, MRF interpolator using the Gibbs sampler; triangle, MRF interpolator using first order mean field approx.; asterisk, MRF interpolation using the improved mean field approx.

### Improved mean field interpolator

Because the potential function used for the interpolator in this example was quadratic, the gradients of the improved mean field interpolator allow the algorithm to converge very closely to the optimum. This is the cause of the very low mean squared errors shown on figure 3.17. The use of gradient based optimisation also allowed this algorithm to converge much more quickly than the other two, based as they are on knowledge of the complete conditional distributions. The visual quality of the restored frames was also very good. Figure 3.19 shows the restored frame, based on knowledge of the true scratch locations. Comparison with figure 3.18 shows the quality of the restoration.

#### 3.3.7 Interpolating the Areas Detected by the Scratch Detection Algorithm

Figure 3.20 is the restored frame from the artificially degraded sequence, where stochastic simulated annealing has been used to detect the scratches, and the improved mean field interpolator has been used to interpolate into these regions. The artifacts due to the false alarm regions, for example in the actor’s hair, are not distracting and the overall visual quality is very good.

Figure 3.21 shows a frame from the ‘Frank’ sequence, exhibiting naturally occurring blotches. Figure 3.22 show the restoration using the methods described in this chapter (SSA detection followed by Improved Mean Field annealing restoration using a neighbourhood of 8 pixels in the current frame and 5 in each of the preceding and following frames).

### 3.4 Summary

In this chapter we have presented the theory of Markov Random Field image models necessary for the development of algorithms for the detection and removal of ‘blotches’ from degraded motion picture films. For each of these MRF specifications we showed how Stochastic Simulated Annealing could be used to find the MAP estimate, and how Mean Field Annealing could be used to find the minimum variance estimate. Comparisons between these estimates were made. Existing approaches to scratch removal were reviewed and comparisons with existing detectors showed the improvements available with the new detection algorithm. The efficacy of the interpolation algorithm was demonstrated by the quality of the restorations produced.



Figure 3.18: Undegraded frame



Figure 3.19: Interpolated frame, known scratch locations



Figure 3.20: Interpolated frame, scratch locations from SSA detector



Figure 3.21: Frame from real degraded sequence



Figure 3.22: Restored frame

---

## COMBINING THE MOTION ESTIMATION AND SCRATCH DETECTION

---

In this chapter we derive a motion estimation algorithm tuned to the problem of detecting scratches in motion picture material. In section 4.2 conventional approaches to motion estimation via block matching and gradient based algorithms are described. Extensions to the Markov Random Field theory of chapter 3 are presented in section 4.3. In section 4.4 a review of other approaches to motion estimation using Markov Random Fields is presented. In this section we also show how the extensions described in section 4.3 enable a field formulation to be built to give a motion estimation scheme more suited to the scratch detection problem - in fact occlusion detection is incorporated directly into the motion estimator. Section 4.5 gives results of experiments demonstrating the action of the combined motion estimator/scratch detector.

### 4.1 Introduction

In chapter 3 we presented an MRF formulation which enabled scratches to be detected in motion picture material. That algorithm took as an input a set of motion vectors estimated separately – for the experiments in chapter 3 multiresolution block matching was used. The detection algorithm thus essentially works on the *displaced frame differences* (DFDs), the residuals when motion compensation has been performed, and finds connected regions where the DFD has values which are not consistent with the motion estimator having been effective. The detection algorithm uses the motion estimator in a manner for which it was not designed – the motion estimation algorithm must not be able to match the scratch areas to produce low DFDs or the scratches will not be detected. Clearly it would be preferable to develop a detection algorithm which was integrated into the motion estimator, such that the motion compensation and occlusion detection were performed together, rather than detecting occlusion as a post-processing stage. In this chapter we show how MRFs enable this to be done.

Conventional motion estimation algorithms are of two main types – pixel matching, which try to find the corresponding pixels (or blocks of pixels) in the next frame [10], and pel-recursive algorithms [9] which operate on intensity gradients. The algorithm we develop

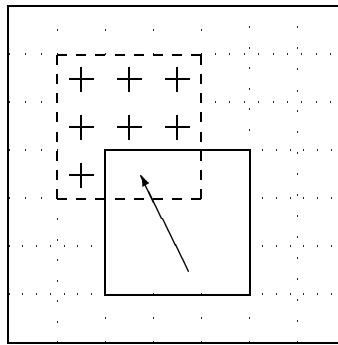


Figure 4.1: The origin of occlusion. The square moves up and left. The areas marked + have no corresponding pixels in the following frame.

in this chapter is of the pixel matching type. Motion estimation is an ill posed problem – a great many sets of motion vectors can be found which will produce a low DFD. For example a pixel in a uniform region can be matched to any pixel in that region in the next frame. Few of these sets of estimates will bear any relationship to the true motion of the scene, particularly in areas of low texture information. The only way to encourage a pixel matching algorithm to find an approximation to the true motion is to enforce ‘smoothness’ constraints on the motion estimates. Under the common assumption of translational rigid body motion the motion vectors will be uniform within the regions corresponding to each object, with discontinuities only at the object boundaries. Whilst natural scenes do not consist of rigid bodies undergoing pure translation this still provides a reasonable model; the motion vectors within objects tend to vary slowly and motion discontinuities still tend to occur at object boundaries. Together with the motion vectors and motion discontinuities we must also consider areas of occlusion. When an object in one frame moves there is an area of background along its edge which does not exist in the next frame. This area is known as the occlusion area – see figure 4.1. Scratch areas are areas which are occluded in both the forward and backward motion estimates, and so the inclusion of occlusion detection into the motion estimator is of great importance.

In section 4.3 we present the extensions to the theory of Markov Random Fields which enables the discontinuities to be modelled explicitly, by the inclusion of a so-called ‘line field’ [33]. In section 4.4 we use this hierarchical framework, together with the freedom to specify the state-space of each variable to construct a motion estimation algorithm which incorporates occlusion detection in an effective and consistent manner. First, however, we review conventional approaches to motion estimation via block matching and gradient based algorithms, and later in section 4.4.2 other motion estimation algorithms based on MRFs will be reviewed.

## 4.2 Conventional Approaches to Motion Estimation

Motion estimation has been a major research area in its own right for many years. One motivation was to enable image sequence compression schemes to remove further redundancy from the transmitted information, by making a more accurate prediction of the next frame than was possible without motion compensation. In parallel, workers in computer vision developed motion estimation algorithms to try to extract depth information from image sequences. Motion estimation has grown to be recognised as being beneficial to almost all image sequence processing tasks, enabling improvements in accuracy and picture quality over that which could be produced by processing each frame independently [56].

Ideally, a motion estimation algorithm would start from the knowledge that the motion in the image sequence is due to the projection of a moving three dimensional scene onto a two dimensional plane. Unfortunately such models are only tractable in controlled environments [70], and instead most approaches work by trying to find correspondences between areas of consecutive two-dimensional image frames. In general this will not result in an estimate which corresponds to the projection of the three dimensional motion [102]. If the displacement field is denoted by  $\mathbf{D}$ , then motion estimation reduces to estimating a configuration  $\mathbf{d}$ , which in some sense optimises the prediction equation

$$\mathbf{x}_2(\mathbf{i}) = \mathbf{x}_1(\mathbf{i} + \mathbf{d}) \quad (4.1)$$

As it stands this cannot correctly model certain types of motion. Zooms and object rotation cannot be fitted directly to this model. For very small zooms and rotations the model is adequate, however. It also fails to account for possible lighting changes between the two frames.

A number of methods have been proposed to estimate  $\mathbf{d}$  from equation 4.1. Block matching [10] is conceptually the simplest and most direct. Gradient based approaches [9] have found wide application due to their low computation cost. Many reviews of these approaches have been written [56], and so they will be only briefly outlined in the following sections. Transform based approaches have also been used [103]. There is also some work in the literature on the application of MRF models to this problem. This work will be reviewed in section 4.4.2, and its relation to the algorithm developed in section 4.4 discussed.

### 4.2.1 Block Matching Motion Estimation

Block matching starts with the prediction equation and tries to solve it by searching directly for the  $\mathbf{d}$  which gives best prediction. Because motion estimation is ill-posed, searching for a value of  $d$  at each pixel independently would lead to many of the motion estimates being

incorrect. To try to overcome this the displacements of *blocks* of pixels is estimated. From equation 4.1 we can define the DFD as

$$\text{DFD}(\mathbf{i}, \mathbf{d}) = \mathbf{x}_2(\mathbf{i}) - \mathbf{x}_1(\mathbf{i} + \mathbf{d}) \quad (4.2)$$

A block of pixels is chosen in frame 1, and the DFD calculated for that block when it is moved to all the positions in a pre-defined search area. The displacement at which the sum of the absolute DFD values is minimum is taken to be the motion estimate for that block. The averaging effect of matching blocks rather than individual pixels helps to regularise the solution. A number of problems still remain, and various methods have been proposed to overcome them.

If the displacement from one frame to the next is *large*, then the search area required would need to be correspondingly large. This increases enormously the computation required, and greatly increases the chance that a spurious match will be found, which has no relation to the true motion. To overcome this, in [10] a *multiresolution* approach was proposed.

If the original image frames are low-pass filtered, and then spatially subsampled (usually by a factor of two), then any motion between the frames will also be reduced by the same factor. The low-pass filter will also tend to reduce spurious matches by reducing any noise on the images. A block matching estimate on the subsampled images can then be propagated down to the larger images. At this level the estimate is *refined* – it is used to define the centre point of a region about which a block is displaced to find the final motion estimate – see figure 4.2. To estimate very large displacements a number of resolution levels can be employed, at each stage low-pass filtering and subsampling the higher resolution images, and propagating estimates to be refined from the lower to the higher resolution images.

Whilst this multiresolution framework allows large displacements to be estimated, it suffers from what is termed the ‘vector halo’ effect. At a low resolution level a block may be assigned a motion vector due to it overlapping the edge of a moving object. At the higher resolution levels this will result in the blocks outside the object being assigned motion vectors as if they were within the object, and the refinement search will not, in general, be able to correct this. In [10] the following solution was proposed. At the end of the estimation, compare the sum of the absolute DFD values for the motion estimate with the sum for the zero-motion case. If the latter is smaller, reset the motion vector for that block to zero.

A further refinement was also implemented. The sum absolute DFD was first calculated

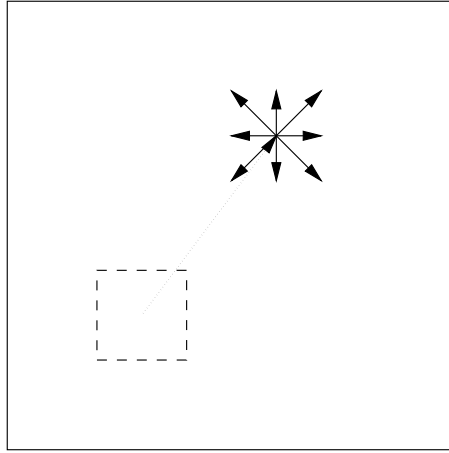


Figure 4.2: Illustration of multiresolution block matching - dotted arrow, estimate from subsampled image; solid arrows, search area for refining the estimate

for the zero displacement case. If this was below a threshold based on the assumed noise level on the image, then it was assumed that the block was not moving. This can reduce the number of spurious estimates caused by noise on the images. It also eliminated the computation needed to evaluate the sum absolute DFD at the other displacement values. For sequences where much of the frame is not moving this can provide a significant saving. Boyce [15] also reset estimated vectors to zero if the sum absolute DFD ratio of the no-motion estimate to the best motion estimate found was smaller than that suggested by the noise on the image.

#### 4.2.2 Gradient Based Motion Estimation

Block matching as described above is computationally intensive, and also only estimates motion to integer pixel accuracy, unless some form of interpolation is used. Both these factors were motivation for the development of gradient based approaches. The most complete gradient based algorithm which does not use explicit constraints on the motion or a more complex prediction model than equation 4.1 is that of Biemond *et al.* [9].

Consider equation 4.2. Expanding about an estimate  $\mathbf{d}^j$  gives, at the  $(j+1)$ th iteration,

$$\begin{aligned} \text{DFD}(\mathbf{i}, \mathbf{d}^j) &= \mathbf{x}_2(\mathbf{i}) - \mathbf{x}_1(\mathbf{i} + \mathbf{d}^j) \\ &= \mathbf{u}^{jT} \nabla \mathbf{x}_1(\mathbf{i} + \mathbf{d}^j) + \mathbf{e}(\mathbf{i} + \mathbf{d}^j) \end{aligned} \quad (4.3)$$

where  $\mathbf{u}^j = \mathbf{d} - \mathbf{d}^j$  is the update to the displacement vector,  $\nabla$  is the two dimensional gradient operator, and  $\mathbf{e}$  is the truncation error from the Taylor series expansion.

In equation 4.3 there are two unknowns at each site – an underdetermined system. By collecting data at  $N$  points,  $N$  variations of equation 4.3 can be collected into the matrix

equation

$$\mathbf{z} = \mathbf{G}\mathbf{u} + \mathbf{e} \quad (4.4)$$

where  $\mathbf{z}$  is a column vector of DFD values,  $\mathbf{G}$  is a  $2 \times N$  matrix of spatial gradients and  $\mathbf{e}$  a column vector of noise realisations, considered to be white Gaussian with variance  $\sigma_{ee}^2$ . The optimum linear least-squares estimate of the update,  $\mathbf{u}$ , is given by the Wiener estimator as

$$\begin{aligned} \mathbf{u} &= [\mathbf{G}^T \mathbf{G} + \mu \mathbf{I}]^{-1} \mathbf{G}^T \mathbf{z} \\ \mu &= \sigma_{ee}^2 / \sigma_{uu}^2 \end{aligned} \quad (4.5)$$

where  $\sigma_{uu}^2$  is the variance of the update. The term  $\mu \mathbf{I}$  tends to stabilise the matrix inverse and regularise the solution. Equation 4.5 is iterated from a zero initial displacement until the updates found are below a threshold.

The gradient based algorithm of equation 4.5 suffers from a number of limitations. The main limitation is due to the use of the Taylor expansion. This expansion is only valid for small updates,  $\mathbf{u}$ , and this limits the range of the displacements that allow the algorithm to converge successfully. As in the case of block matching a multiresolution approach can be used, with the propagated vectors forming the initial displacement estimates. Low-pass filtering the images when generating the subsampled images will also improve the validity of the Taylor expansion and aid convergence.

Different formulations of the gradient based motion estimation algorithm have been proposed. Most of these aim to overcome the ‘aperture problem’ – if the pixels used to construct equation 4.4 cross the boundary of an object which is moving parallel to the boundary, the motion in the direction of the boundary cannot be estimated. Various solutions have been proposed, from identifying the areas where this occurs, and only estimating motion perpendicular to the edges at these location [13], to the use of ‘oriented smoothness’ constraints [70]. It is beyond the scope of this brief review to detail these further. A more complete discussion can be found in [56, 4].

### 4.3 Hierarchical Markov Random Fields

This section contains extensions to the theory of MRFs which will be used later in the construction of the new motion estimation algorithm.

In section 3.2 we presented some of the theory of Markov Random Fields. In that section an MRF was shown to be governed by a joint distribution which had the form of

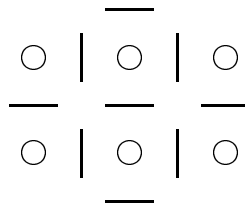


Figure 4.3: The pixel and line neighbours of a line element

a Gibbs distribution. This gave the probability of each configuration of the field, and in the practical sections of chapter 3 the variables were taken to be either the elements of the binary discontinuity field, or the image grey level values.

The interpolation algorithm of section 3.3.4 used an MRF with the quadratic potential function, which encourages global smoothness. For the typically small areas to be interpolated this is adequate, but clearly larger areas of images do not exhibit such smoothness – the edges of objects provide very obvious discontinuities, and it would be useful to include these boundaries in the model. The formulation of MRF models places no restriction on the types of variables involved; the variables need not have the same state space. Thus we may define auxiliary variables to model the discontinuities.

Typically edge variables are placed between each pair of pixel variables as illustrated in figure 4.3, suitably modified at the image boundary. These line variables are usually binary, and if set to one serve to break the link between the two pixel variables on either side. Thus the potential function for the two element cliques becomes

$$V_C(x_i, x_j) = \phi(x_i, x_j)(1 - l_{i,j}) \quad (4.6)$$

where  $\phi(x_i, x_j)$  is an arbitrary function of  $x_i$  and  $x_j$ , and  $l_{i,j}$  is the line element between the pixels at sites  $i$  and  $j$ .

If the potential function of equation 4.6 was used in equation 3.2 then the energy would be minimised, and hence the probability maximised, for a field with all line elements taking values of one. To prevent this, potential functions for the various line cliques associated with the neighbourhood of figure 4.3 can be defined. These can serve to organise the lines. Typically we expect lines to be continuous, with few terminations or junctions. This can be explicitly specified in the values the potential function takes for each line configuration. Figure 4.4 illustrates the various configurations possible (up to a rotation), together with the associated potential function values. These configurations and costs (or similar costs) are commonly used [33, 61, 105]. Also included in the cliques and costs illustrated in figure 4.4 is a cost which serves to exclude isolated pixels, which rarely occur in real images.

We now have a model which specifies the joint probability of two fields, the pixel field,

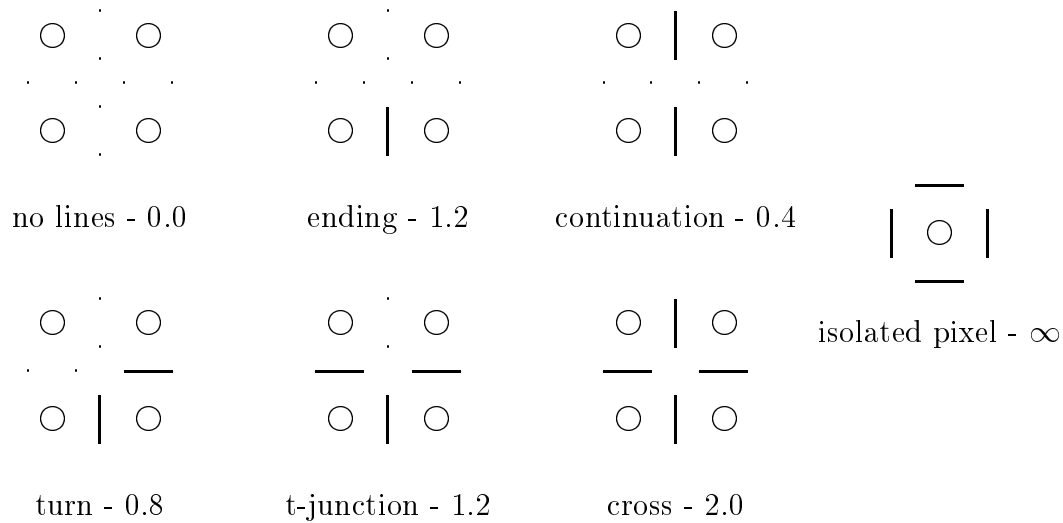


Figure 4.4: Cliques and costs associated with a line element

$\mathbf{X}$ , and the line field,  $\mathbf{L}$ , and the distribution is governed by the energy function

$$U_{\mathbf{x},\mathbf{l}}(\mathbf{x}, \mathbf{l}) = \sum_{\mathcal{c}} \phi(x_i, x_j)(1 - l_{i,j}) + \lambda_1 \sum_{\mathcal{L}} V_C(\mathbf{l})$$

The first summation being over all two element pixel cliques and the second being over all the line cliques specified. The parameter  $\lambda_1$  weights the relative importance of the two terms. There have been some efforts made to eliminate the line field from the model, by choosing potential functions with particular characteristics [32]. The motivation was to try to reduce the computation involved [28]. However to process the line field takes much less time than to process the displacement field, and the flexibility of being able to specify line costs to enable the lines to self-organise means that a separate line field remains attractive.

We now describe the approach using this framework which has been used by other researchers to construct motion estimation algorithms, and then use it in the development of a new algorithm which performs motion estimation with occlusion detection.

#### 4.4 Estimating Motion, Discontinuities and Occlusion

The framework of section 4.3 allows discontinuities to break the smoothness encouraged by certain types of potential function. In the discussion of motion estimation in section 4.1 we stated that the translation model of motion comprises areas of uniform motion with discontinuities between them. It can be seen that the hierarchical MRF framework can model this well.

#### 4.4.1 Estimating Motion and Discontinuities

The algorithm we are developing is of the pixel correspondence type. We wish to find the optimum displacement field  $\mathbf{D}$ , where each variable  $d_i$  has two components, the displacement in the vertical and horizontal directions from the pixel in the first frame to the corresponding pixel in the second frame. The closeness of this match is usually measured using the displaced frame difference (DFD), which is  $\text{DFD}(i) = x_{1i} - x_{2(i+d_i)}$ . As discussed earlier, there are many configurations of  $\mathbf{D}$  which will result in a low DFD across the frame. The purpose of the MRF model on  $\mathbf{D}$  is to specify which configurations fit our a-priori model of smoothness broken by organised boundaries.

Thus we are trying to estimate the displacement field  $\mathbf{D}$  and the line field  $\mathbf{L}$  from the data provided by the two frames  $\mathbf{x}_1$  and  $\mathbf{x}_2$ . Bayes theorem gives

$$p(\mathbf{d}, \mathbf{l} | \mathbf{x}_1, \mathbf{x}_2) \propto p(\mathbf{x}_2 | \mathbf{x}_1, \mathbf{d}, \mathbf{l}) p(\mathbf{d}, \mathbf{l} | \mathbf{x}_1) \quad (4.7)$$

The first term on the right hand side is a measure of how well the second frame is predicted from the first using the displacement field  $\mathbf{d}$ . Because we have no real information as to the form of this distribution, a Gaussian assumption is commonly made [26], giving an energy function of

$$U_{\mathbf{x}_2 | \mathbf{x}_1, \mathbf{d}, \mathbf{l}}(\mathbf{x}_2 | \mathbf{x}_1, \mathbf{d}, \mathbf{l}) = \lambda \sum_S |x_{2i} - x_{1(i+d_i)}|^2 \quad (4.8)$$

This energy function is not an explicit function of  $\mathbf{l}$  – the influence of the line field is implicit in the particular configuration of the vector field.

We decompose the second term as

$$p(\mathbf{d}, \mathbf{l} | \mathbf{x}_1) = p(\mathbf{d} | \mathbf{l}, \mathbf{x}_1) p(\mathbf{l} | \mathbf{x}_1)$$

The first of these is the prior on the displacement process. We can infer very little about the displacements from the single frame  $\mathbf{x}_1$ , and so we use just  $p(\mathbf{d} | \mathbf{l})$ . This must encode the smoothness of the displacement field within objects defined by the particular configuration of the line field. The energy function

$$U_{\mathbf{d} | \mathbf{l}}(\mathbf{d} | \mathbf{l}) = \lambda_d \sum_c \|d_i - d_j\|^2 (1 - l_{i,j})$$

where  $d_i$  is the *vector* displacement at location  $i$  will encourage this behaviour – when the vectors are not separated by an active line element any difference is penalised. No penalty is incurred for differing vectors either side of a discontinuity. The quadratic potential function does allow some slow variation of vectors within objects which will allow the motion of real

world objects to be modelled more closely.

The second term is the prior on the line field. Again the cliques and costs in figure 4.4 are used to organise the lines. Motion discontinuities tend to occur at object boundaries, where we also tend to have a grey level discontinuity. This is modelled by the inclusion of a single element clique which is a function of the intensity gradient, resulting in

$$U_{\mathbf{l}|\mathbf{x}_1}(\mathbf{l}|\mathbf{x}_1) = \lambda_1 \sum_{\mathcal{L}} V_L(\mathbf{l}) + \alpha \sum_L \frac{1}{(\nabla \mathbf{x}_1)^2}$$

where the second summation is over all the line elements. Finally we have

$$p(\mathbf{d}, \mathbf{l}|\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{Z} \exp \left( -\frac{1}{T} \left[ U_{\mathbf{x}_2|\mathbf{x}_1, \mathbf{d}, \mathbf{l}} + U_{\mathbf{d}|\mathbf{l}} + U_{\mathbf{l}|\mathbf{x}_1} \right] \right) \quad (4.9)$$

#### 4.4.2 MRF Based Approaches to Motion Estimation

In the discussion of conventional motion estimation algorithms in section 4.2 it was stated that motion estimation is an *ill posed* problem. The solution of ill posed problems generally involves regularisation, or from a Bayesian viewpoint, the incorporation of any prior knowledge of the solution. The MRF model has been shown to be suitable for modelling spatial coherence, and has been used by a number of authors to model the expected displacement fields. It provides an alternative to the smoothness constraints often used with gradient based motion estimation algorithms.

In [61], Konrad and Dubois derived the formulation discussed in section 4.4.1 above. They produced a more general scheme, aimed at estimating motion vectors for motion-compensated interpolation tasks. This was done by estimating the displacements at a time between the image frames. They also used line cliques which discouraged the formation of double edges. For the discrete state-space case (a fixed number of possible motion vectors at each site) either simulated annealing or minimum expected cost (MEC) estimation was used. The second of these involves simulating the field at  $T = 1$  and averaging the samples at each site to form the estimate.

In an attempt to reduce the computation required by the discrete state-space sampling schemes Konrad and Dubois also proposed an approximate continuous-space approach. In a similar manner to the gradient based motion estimation algorithm of section 4.2.2 they linearised the energy term involving the DFD (equation 4.8) using the Taylor expansion about the current estimate. The distribution which results from this simplification was

fitted to a bivariate Gaussian distribution at each site,

$$p(d_i) = \frac{1}{2\pi|\mathbf{M}|^{1/2}} \exp\left(-\frac{1}{2}(d_i - \mathbf{m})^T \mathbf{M}^{-1}(d_i - \mathbf{m})\right)$$

In this expression  $\mathbf{m}$  was a value similar to the mean of the neighbouring displacements and the covariance matrix was very similar to the  $[\mathbf{G}^T \mathbf{G} + \mu \mathbf{I}]$  matrix of the gradient based motion estimation algorithm, the differences mainly being due to the inclusion of edge variables in Konrad and Dubois' formulation. Once the parameters of the bivariate Gaussian have been calculated it is a simple matter to generate new values by sampling from this distribution.

In [22] the same authors presented extensions to this algorithm, again aimed at improved spatiotemporal interpolation. There were two main changes from the algorithm described above. More frames were used to estimate the motion, taking into account occlusion and uncovering to determine which frames could be used to predict areas of the current frame, and the motion was assumed to be some parametric form, not necessarily uniform, non-accelerated motion. It was always assumed that somewhere in at least one of the surrounding frames a matching area could be found, so this did not model occlusion in the manner required for the scratch detection problem.

Stiller [94] used a very basic MRF to smooth the estimates from a block matching estimator. His energy function was based on the eight nearest neighbours, and was of the form

$$U_{d_i}(d_i) = \sum_{j \in \mathcal{N}_i} \frac{1}{l} \|d_i - d_j\|$$

where  $l$  was the distance between the neighbours (1 for vertical/horizontal or  $\sqrt{2}$  for diagonal neighbours). Instead of optimising the combined DFD/smoothness function exactly only the relative probabilities of a number of plausible candidate vectors were calculated. These candidate vectors were based on the current values of the vectors in the neighbourhood. Initial values came from a reduced-search block matching estimator. The vector chosen as the new value was that with highest relative probability. This significantly reduced the computation required, and did not affect the estimates unduly, as the vectors neglected would in general have very low probability.

In [44], Heitz and Bouthemy propose a very complete MRF based motion estimation algorithm, combining gradient based and feature based [14] motion estimation algorithms. One element of their estimator was a cost function based on the standard gradient based motion estimator constraint. They also, however, looked explicitly for moving edges and used the motion of the edges as further constraints on the estimated motion. This often

helped overcome the aperture problem. Occlusion areas were also detected by testing the hypothesis that the first order spatial derivatives at a point were the same on both frames. If this was not the case then the area was assumed to be occluded. The directions of motion of the moving edges were also constrained by the occlusion areas. Integrating all these constraints produced a very accurate estimator. The authors report that the occlusion detection part of the algorithm had most effect in enabling accurate motion vector estimates to be found.

The mean field approximations discussed in chapter 3 have also been applied to motion estimation. The mean field approach based on the Gibbs-Bogoliubov-Feynman bound (equation 3.11) was used by Abdelquader *et al.* [1]. Their formulation used an energy function of the form of equation 4.8 for the likelihood, and for the prior used the energy function

$$U_{\mathbf{d}}(\mathbf{d}) = -\lambda_d \sum_c \exp(-\|d_i - d_j\|^2/2\tau)$$

This penalises variations in neighbouring vectors *very* severely, and tends to enforce piecewise constancy. The result of the mean field approximation was an iterative estimator based on spatial and temporal gradients.

Zhang and Hanauer [105] applied the first-order mean field approximation. Their formulation included an explicit line field, together with an occlusion field, which indicated the validity of the motion estimates at each pixel. Using this occlusion estimation approach with the mean field estimate avoids the issue of defining motion vectors at sites where occlusion is indicated, by taking the occlusion field elements to be continuous valued, between 0 and 1, indicating the relative plausibility of the motion vector estimate. The final estimate is made by thresholding the occlusion field values and then discarding the appropriate motion estimates. They also showed how to derive mean field equations for line costs based on look-up tables.

### 4.4.3 Modelling the Occlusion

In the derivation in section 4.4.1 we assumed that it was reasonable to define a displacement at each pixel. As figure 4.1 clearly shows, there may be areas of frame one, the occlusion areas, which simply have no corresponding area in frame two. In these areas it makes no sense to define a displacement vector. In [61, 105] an occlusion detection framework was developed which *did* define a displacement vector at these areas, but then ignored its effect if a separate occlusion field indicated the presence of occlusion at that site. Here we develop a formulation where the occlusion is treated directly with the displacement, by defining the variables at each site to take values from a set of states, one of which is the

occlusion state. In this manner we avoid having to define and then ignore displacements, and treat the motion and occlusion in a unified framework.

For the formulation in [105] it is unclear how to sample from the distribution for the displacement at location  $i$  if occlusion is indicated at that location, if the mean field approximation is not being used. The cost function involving the DFD becomes

$$U_{\mathbf{x}_2|\mathbf{x}_1, \mathbf{l}, \mathbf{o}}(\mathbf{x}_2|\mathbf{x}_1, \mathbf{l}, \mathbf{o}) = \sum |x_{2i} - x_{1(i+d_i)}|^2(1 - o_i)$$

where  $\mathbf{o}$  is the separate occlusion field. Consider the case when  $o_i = 1$ , indicating occlusion at site  $i$ . The DFD term is eliminated, and a value for  $d_i$  sampled just on the basis of the smoothness constraint. This motion vector is, however, meaningless, as the occlusion field is indicating that there is no valid motion at that point. It is an artifact of the algorithm. We now formulate an MRF where the occlusion is included in the same field as the displacements. This produces a much more coherent framework

Equation 4.7 becomes

$$p(\mathbf{s}, \mathbf{l}|\mathbf{x}_1, \mathbf{x}_2) \propto p(\mathbf{x}_2|\mathbf{x}_1, \mathbf{s}, \mathbf{l})p(\mathbf{s}, \mathbf{l}|\mathbf{x}_1) \quad (4.10)$$

where  $\mathbf{s}$  is the field of states, where the state space consists of states corresponding to the displacement vectors and a state which is the occlusion state. We now specify each term in this equation in detail.

The first term in equation 4.10 is again the likelihood, the probability of observing frame 2 given frame 1 and the particular configuration of the state field and line fields. Again we use a Gaussian distribution based on the DFD values, except now the DFD is only defined for those sites where the state field takes a value corresponding to a motion vector. Where occlusion is present we can make no prediction of frame 2, and cannot define a cost based on the DFD. The influence of the line field is implicit in the configuration of the state field, giving

$$U_{\mathbf{x}_2|\mathbf{x}_1, \mathbf{s}}(\mathbf{x}_2|\mathbf{x}_1, \mathbf{s}) = \lambda \sum_{\mathbf{s} \neq \text{occlusion state}} |x_{2i} - x_{1(i+v_i)}|^2 \quad (4.11)$$

Where  $v_i$  is the displacement corresponding to the value of the state field at site  $i$ .

The second term in equation 4.10 is decomposed as  $p(\mathbf{s}, \mathbf{l}|\mathbf{x}_1) = p(\mathbf{s}|\mathbf{l}, \mathbf{x}_1)p(\mathbf{l}|\mathbf{x}_1)$ . The first of these is the prior on the field of states, and in a similar manner to  $p(\mathbf{d}|\mathbf{l}, \mathbf{x}_1)$  is taken

to be independent of  $\mathbf{x}_1$ . The energy function which governs  $p(\mathbf{s}|\mathbf{l})$  has the following form

$$\begin{aligned}
 U_{\mathbf{s}|\mathbf{l}}(\mathbf{s}|\mathbf{l}) &= \lambda_d \sum_{\substack{\text{all cliques not in-} \\ \text{cluding an occlu-} \\ \text{sion}}} \|v_i - v_j\|^2 (1 - l_{i,j}) \\
 &+ \lambda_d \sum_{\substack{\text{all cliques including} \\ \text{one occlusion state}}} (1 - l_{i,j}) \mu \\
 &+ \lambda \sum_{\text{all occlusion states}} \vartheta
 \end{aligned} \tag{4.12}$$

It can be seen that this energy function has the desired characteristics with regard to organising the displacement states and the occlusion states. The first term is a smoothness penalty, which encourages neighbouring displacements to have similar values, unless separated by an active line element. The second term defines a ‘distance’ from an occlusion state to a vector state. This distance,  $\mu$ , is the same for all vector states, as occlusion occurs at the edge of any moving object, regardless of the magnitude of its motion<sup>1</sup>. Thus we have no cost for a clique containing the same state in each element of the clique, be they displacement states or occlusion states. This organises the displacement states and vector states into regions. The final term in equation 4.12 is the cost for introducing an occlusion state, to avoid  $p(\mathbf{s}, \mathbf{l}|\mathbf{x}_1, \mathbf{x}_2)$  being maximised by a field comprising solely of occlusion states.

The final term in  $p(\mathbf{l}|\mathbf{x}_1)$  is the same prior on the line process as was described earlier, with

$$U_{\mathbf{l}|\mathbf{x}_1}(\mathbf{l}|\mathbf{x}_1) = \lambda_l \sum_{\mathcal{L}} V_L(\mathbf{l}) + \alpha \sum_L \frac{1}{(\nabla_{\mathbf{x}_1})^2} \tag{4.13}$$

to organise the lines and encourage line formation at positions of high spatial gradient.

The combined energy function is thus

$$U_{\mathbf{s}, \mathbf{l}|\mathbf{x}_1, \mathbf{x}_2} = U_{\mathbf{x}_2|\mathbf{x}_1, \mathbf{x}} + U_{\mathbf{s}|\mathbf{l}} + U_{\mathbf{l}|\mathbf{x}_1} \tag{4.14}$$

#### 4.4.4 Parameter Estimation

The distribution  $p(\mathbf{s}, \mathbf{l}|\mathbf{x}_1, \mathbf{x}_2)$  is governed by the energy function of equation 4.14. This can be seen to depend on six parameters,  $\lambda$ ,  $\lambda_d$ ,  $\lambda_l$ ,  $\mu$ ,  $\vartheta$  and  $\alpha$ . The relative values of which determine the weight given to each term of  $U_{\mathbf{s}, \mathbf{l}|\mathbf{x}_1, \mathbf{x}_2}$  – the trade off between low displaced frame differences, smooth vectors, discontinuities between vectors and at sites of high spatial gradient, and the introduction and spatial organisation of occlusion.

---

<sup>1</sup>It is possible to constrain the occlusion on the *direction* of the edge’s motion, but this is not done here.

The estimation of hyperparameters of Markov Random Fields directly from the image data is extremely complex [18, 37]. Indeed in [105] the parameters were chosen on an ad-hoc basis. Reasonable estimates may, however, be made by considering the relative magnitudes of the terms involved.

### Occlusion vs. DFD

When we introduce an occlusion state, we remove from  $U_{\mathbf{s}, \mathbf{l} | \mathbf{x}_1, \mathbf{x}_2}$  the term  $\lambda \text{DFD}^2$ , and introduce in its place the term  $\lambda \vartheta$ . Hence we need to choose a value for  $\vartheta$  which is equal to the smallest value of the  $\text{DFD}^2$  which we believe to be from a pixel which has no good match. Hence we let

$$\vartheta = \text{DFD}_0^2$$

Where  $\text{DFD}_0$  is the smallest DFD which is too large to be accounted for by noise on the image or by slight lighting changes. This is around  $\text{DFD}_0 = 10$  for low noise, eight bit images.

### Discontinuities vs. Smoothness

Now consider the changes to  $U_{\mathbf{s}, \mathbf{l} | \mathbf{x}_1, \mathbf{x}_2}$  when a discontinuity is introduced between two displacement states. The term  $\lambda_d \|v_i - v_j\|^2$  is removed, and  $\lambda_l V_L(l)$  is introduced. From figure 4.4,  $V_L(l)$  takes values from 0.4 to 2 (and infinity). We wish to allow some variation in the displacements within a moving object to allow for the shape changes seen in real object motion and so must not try to introduce discontinuities between displacements which are similar but not identical. We claim that similar implies that  $\|v_i - v_j\|^2 \simeq 5$ . Thus we remove a term  $\simeq 5\lambda_d$  and include a term  $\simeq \lambda_l$ ; hence we choose

$$\lambda_l \simeq 5\lambda_d$$

### Smoothness vs. DFD

The ratio  $\lambda : \lambda_d$  determines the trade off between smooth motion estimates and low DFD values. High relative values of  $\lambda$  encourage low DFD values; high relative values of  $\lambda_d$  encourage locally smooth displacement estimates. For eight bit images DFD values lie in the range 0 to  $\pm 255$ . In the discussion of the value of  $\vartheta$  we claimed that a good match would result in a DFD value less than 10 (depending on the noise on the image). We also claimed above that smoothness within a moving object should result in  $\|v_i - v_j\|^2 \simeq 5$ .

Balancing these two contributions gives

$$\lambda_d \simeq 5\lambda$$

### Smoothness and Occlusion

The value of  $\mu$  can be considered to be equivalent to the  $\|v_i - v_j\|^2$  term for a clique including one occlusion state. It thus serves as the ‘distance’ from any displacement state to an occlusion state. In a similar manner to the smoothness:discontinuity trade off, the value of  $\mu$  must not be made too small, else it may be easier to introduce an occlusion than to introduce a discontinuity. This gives

$$\mu \simeq 5$$

### Introducing Discontinuities

The parameter  $\alpha$  controls by how much spatial gradients encourage the formation of discontinuities, and  $\alpha/(\nabla \mathbf{x}_1)^2$  is traded off against  $\lambda_l V_L(l)$ . If we assume that a spatial gradient of  $\simeq 20$  (for eight bit images) should encourage line formation, this must balance  $\lambda_l$ , as described, giving

$$\alpha \simeq 400\lambda_l$$

### Simulated Annealing Parameters

To find an estimate of the MAP configuration of the state and line fields, simulated annealing was used. The local conditional distribution for each element of the state field can be calculated easily, as only first order neighbourhoods are used. This discrete distribution over the candidate motion vector states and the occlusion state can then be sampled using simple rejection sampling. The line field is defined over binary variables and is thus not computationally intensive to sample from. Simulated annealing depends on a number of parameters; the initial temperature  $T_0$ , the cooling schedule and the number of iterations, where an iteration is a full update of both the state and line fields.

The initial temperature was chosen such that when updating the state field using rejection sampling, the number of rejections recorded was low, indicating a fairly uniform distribution. An exponential cooling schedule was used, reducing the temperature after each update of both fields by a factor  $a < 1$ . The value of  $a$  and the number of iterations was chosen to give a rejection rate almost equal to the number of states by the end of the

	$\lambda_d$	$\lambda_l$	$\lambda$	$\alpha$	$\vartheta$	$\mu$	$T_0$	$a$	<i>niter</i>
‘square’	8	20	4	3000	6	1.5	2	0.987	40
‘western’	10	50	1	18000	100	7	3	0.987	50

Table 4.1: Parameter values used in the experiments

run.

## 4.5 Experimental Results

To assess the action of the combined motion estimation/occlusion detection algorithm two sets of experiments were performed. The first was on a ‘best case’ artificial sequence, and the second on a real sequence, which had been degraded by adding artificial scratches and blotches.

### 4.5.1 ‘Square’ Sequence

The artificial sequence consisted of an unchanging, textured background, across which a textured square underwent translational motion, with integer motion between frames; a situation which matches closely the ideal case of figure 4.1. In this case a set of displacements exists which give low (actually zero) displaced frame difference and are smooth (actually uniform) within the moving object. The occlusion region is also clearly defined along the edge of the moving region.

Figure 4.5 shows one frame from the sequence, together with the estimated displacements, discontinuities and occlusion regions. The parameters used are listed in the first row of table 4.1. These parameter values do not follow the estimation scheme above precisely. Because we know that the motion is uniform within the moving region a higher value of  $\lambda_d$  can be used to enforce smoothness. Because of the perfect matches available in the non occluded regions a lower value of  $\vartheta$  can be used. Clearly the estimates are extremely good. The motion is estimated exactly within the moving region, with discontinuities around this region. Almost all of the occlusion region was correctly identified, save for a small number of false matches at the side of the moving region.

### 4.5.2 ‘Western’ Sequence

The second sequence used in the experiments was the ‘Western’ sequence, three frames of which are shown in figures 4.6, 4.7 and 4.8. The main motion is in the actor’s hands. His left hand moves upwards and rotates slightly. His right hand also moves forwards and

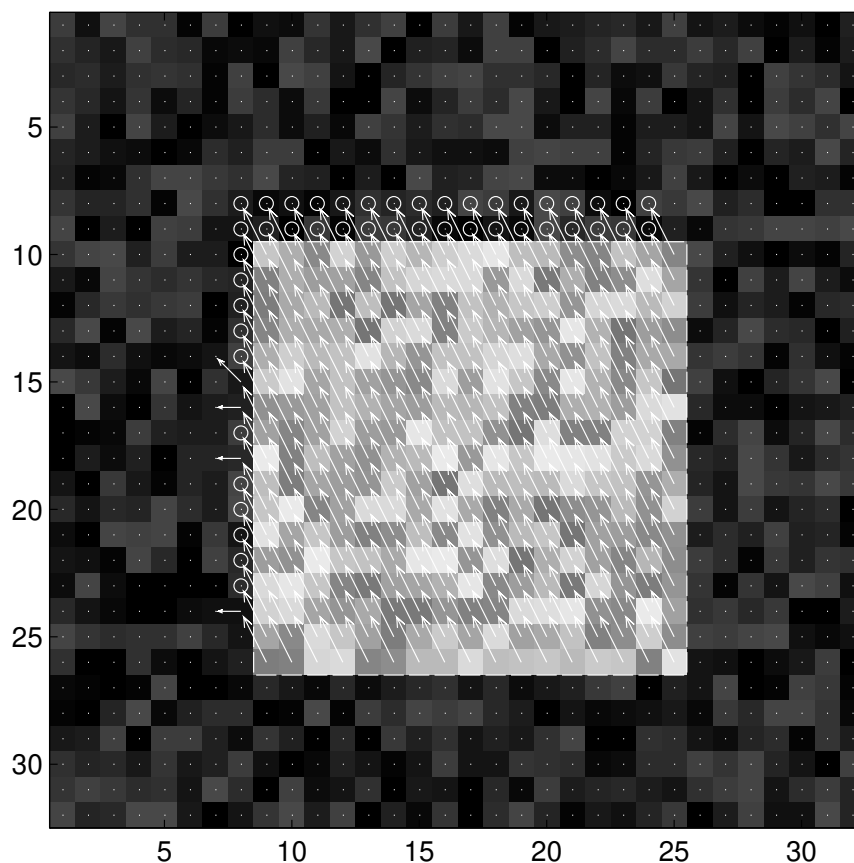


Figure 4.5: Motion vectors, lines and occlusion estimated for 'Square' sequence

upwards, and his right arm moves upwards. The actor's leg enters the bottom of the scene. The figures also show the artificial scratches added to the frames.

As discussed in chapter 3 the scratch regions can be detected as regions having no match in either the previous or following frames. Therefore to detect scratches on frame two, we must estimate motion and occlusion from frame two to frame one (backwards estimation), and separately from frame two to frame three (forwards estimation) and then classify as scratches areas marked as occlusion in both estimates.

The second line of table 4.1 lists the parameters used. These parameters follow closely the discussion in section 4.4.4, the discrepancies being small and found experimentally to give *slightly* better results.

Figure 4.12 shows the displacement, discontinuity and occlusion estimates for the backwards estimate from frame two to frame one. The moving areas, the actor's hands, have been assigned displacements in the correct direction, as has the actor's leg. A small occlusion area has been identified at the edge of the actor's right hand. Most of the visible scratch areas have been assigned occlusion labels, with two notable exceptions. The area behind the actor's head has been assigned displacements which match it to the smaller

scratch area in frame one. The scratch area on the actor's belt has been matched to a small scratch area almost invisible in frame 1. Increasing the strength of the smoothness constraint through the value of  $\lambda_d$  may eliminate this type of problem, by preventing the whole of a large scratch from being matched to a small scratch area. However the artificial contamination on the three frames illustrated is particularly severe, and such situations are likely to be infrequent in real degraded sequences.

Figure 4.13 shows the forward estimates from frame two to frame three. Again the main moving areas have been assigned reasonable displacements. In this case, however, all the visible scratches have been classified as occlusion regions.

Figure 4.9 shows the actual scratch locations, figure 4.10 the detected scratches and figure 4.11 those not detected. There were *no* false detections. Apart from the two areas discussed above which were not detected, the remaining scratch areas are either not visible or only just visible in frame two (figure 4.7). The combined motion estimation/occlusion detection framework thus provides a potentially very accurate scratch detection system.

## 4.6 Summary

In this chapter we have reviewed briefly conventional and MRF approaches to motion estimation. We have presented the theory of hierarchical Markov Random Fields, incorporating discontinuity elements to break the spatial coherence. We have also shown how, by using state descriptions which are very general, an algorithm may be developed for simultaneous estimation of displacements and occlusion in image sequences. This algorithm was combined with the heuristic of chapter 3 and used for the detection of scratch regions on motion picture frames, and found to be effective, especially in the suppression of false detections, a problem with previous detection algorithms.



Figure 4.6: Frame 1



Figure 4.7: Frame 2



Figure 4.8: Frame 3

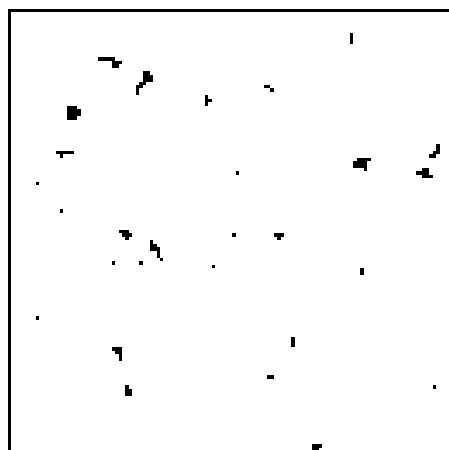


Figure 4.9: Scratch Locations

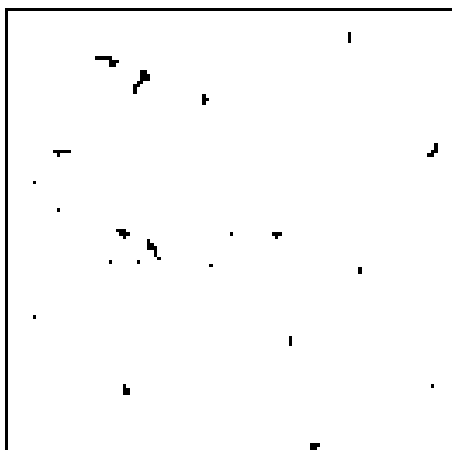


Figure 4.10: Detected scratches

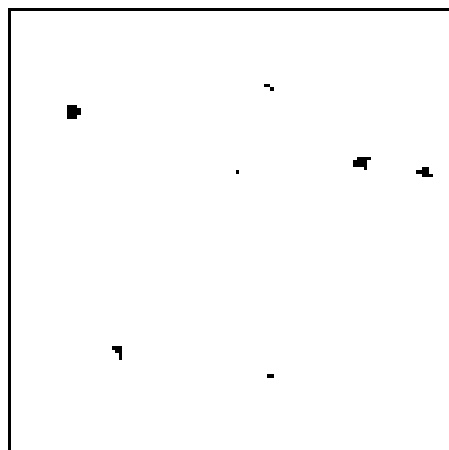


Figure 4.11: Scratches not detected

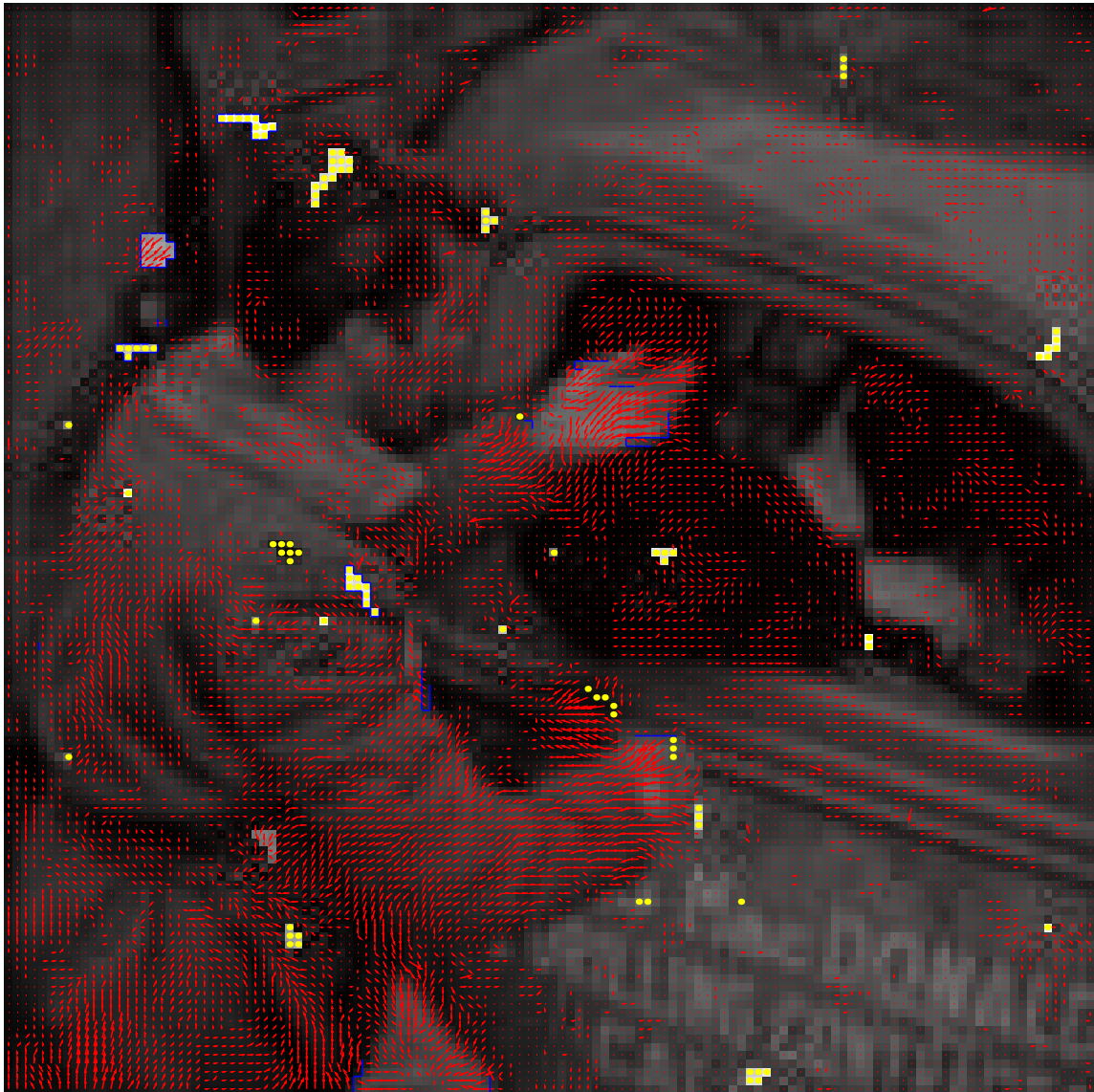


Figure 4.12: Motion (red arrows), discontinuities (blue lines) and occlusion (yellow dots) – frame 2 to frame 1

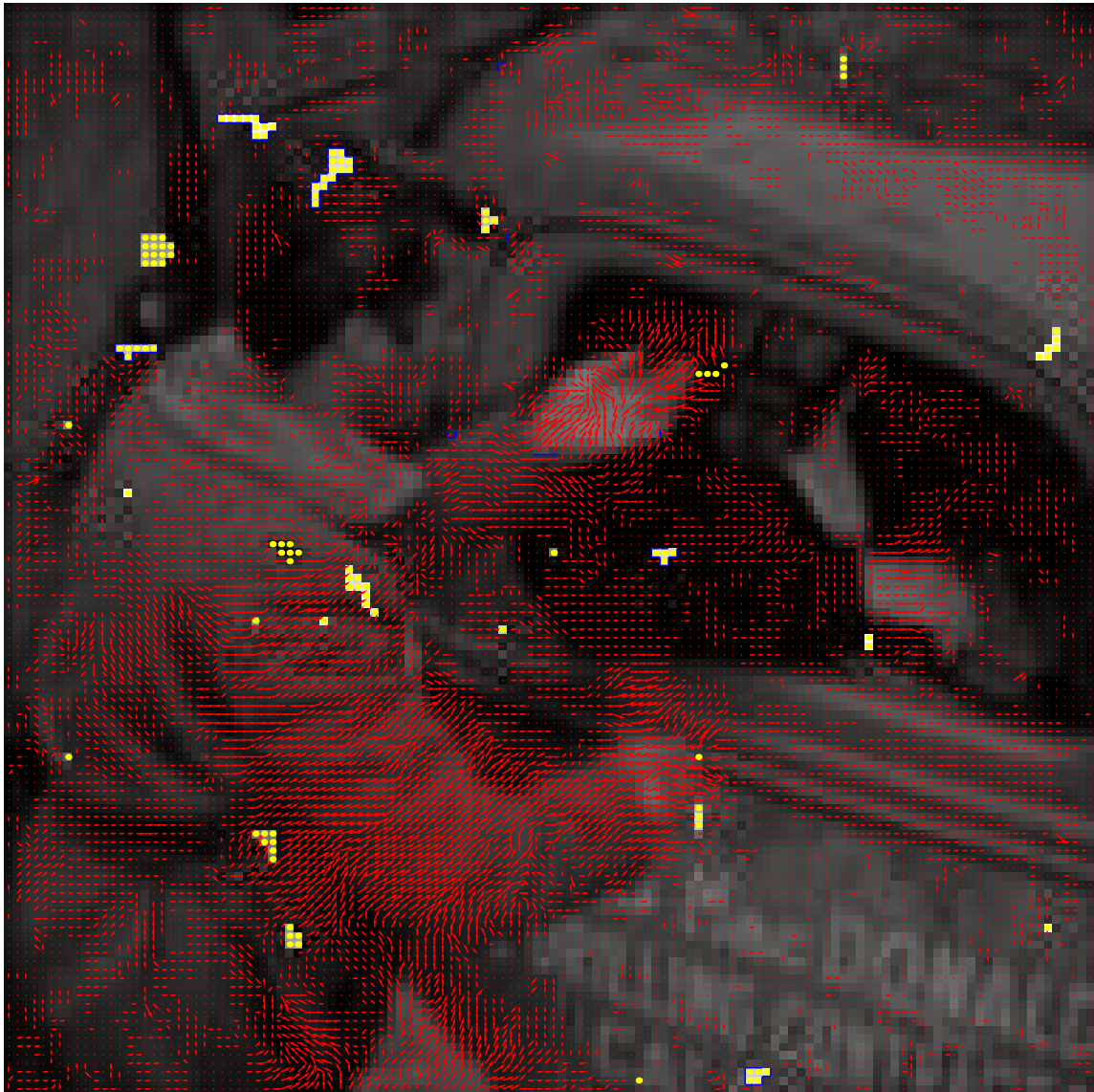


Figure 4.13: Motion (red arrows), discontinuities (blue lines) and occlusion (yellow dots) – frame 2 to frame 3

---

## REVERSIBLE JUMP MCMC AND LINE SCRATCH REMOVAL

---

In this chapter extensions to the Metropolis Hastings algorithm, discussed in chapter 2, are described. The Metropolis Hastings algorithm is applicable only when the dimensionality of the problem is fixed. Thus it cannot be used directly for the problem addressed in this chapter – detecting and removing an *unknown* number of line scratches. The method of Reversible Jump MCMC, which samples from a distribution defined over a union of subspaces of differing dimensionality, is detailed in section 5.2. In section 5.3 the Reversible Jump MCMC framework is used to extract the number of line scratches present in a degraded motion picture frame, together with the lines’ parameters. A restoration of the frame is then performed, and results given.

### 5.1 Introduction

In chapter 3 an algorithm was developed for removing ‘blotch’ type defects from motion picture material. Another major defect type, but one with very different characteristics, is ‘line scratches’. These are caused by the film material running against a stationary foreign object somewhere within the camera or projection equipment, causing the vertical structure. Line scratches persist for many frames, and are often at the same location in consecutive frames. A line scratch manifests itself as a narrow, bright or dark, vertical (or near-vertical) line, extending for most, if not all, of the height of the frame. Figure 5.1 shows a clear example of this type of defect.

Because the line scratches persist across many frames they cannot be detected in the same manner as ‘blotch’ type defects, by looking for pixels at the sites of large temporal discontinuities. Instead more spatial information must be used, and the line scratch’s effect on the appearance of the frame modelled more strongly.

If the number of line scratches present is known, then a standard MCMC algorithm could be constructed, using either the Gibbs sampler or the Metropolis Hastings algorithm to sample from the posterior distribution of the parameters, based on the line scratch model and the image information. Because, however, the true number of line scratches is unknown, a standard MCMC algorithm cannot be constructed. The Reversible Jump



Figure 5.1: Frame exhibiting ‘line scratch’ distortion

MCMC framework of Green [40] provides a solution. It enables a sampler to be constructed which has as its stationary distribution a joint distribution of the form  $p(k, \theta^{(k)} | \mathbf{i})$ , where  $k$  indexes the models of differing dimensionality,  $\theta^{(k)}$  are the parameters of the  $k$ th model and  $\mathbf{i}$  is the data; it provides a method of moving between the different models which maintains the invariant distribution and samples efficiently from the different models by allowing information about the parameter values to be transferred from one model to the next.

## 5.2 Reversible Jump MCMC

The area of Bayesian model determination is one of wide interest. The evidence framework provides a conceptually elegant method of determining which of a set of models is best supported by the available data [16, 26]. For most models of even moderate complexity, however, the integrals needed to calculate the evidence are intractable, and numerical simulation methods need to be used [85]. The usual method is to calculate the evidence for each of the models under consideration separately, using a Monte Carlo estimate of the evidence integral. It would be conceptually more elegant if the model determination could be included directly in the Monte Carlo simulation of the parameter values for each model. This would involve constructing some form of algorithm which sampled from a space of varying dimensionality. In [42], Grenander and Miller used a *jump diffusion* sampler, whereby the transitions between models of different orders were made at random exponential times, and between jumps the parameters of the model were simulated to satisfy a stochastic differential equation. Within the limits of the discretisation used to simulate the stochastic differential equation this maintains the correct invariant distribution. Phillips and Smith

[74] used the same jump diffusion sampler for a number of model choice problems, one of which, their ‘toy’ object recognition problem, shares a number of characteristics with the problem addressed in this chapter. The jump diffusion sampler relies on a number of difficult concepts for its construction, and is not pursued further here. Another sampling based approach to this problem is that of Carlin [17], which requires all the models to be simulated concurrently. In this chapter we concentrate on the approach of Green [40], who constructs a sampler of great generality and simple practical applicability.

In the discussion of Markov chain theory in section 2.2 conditions for the chain to converge to a unique invariant distribution were given. This derivation was limited to the discrete state space case. In the case of continuous variables the sufficient condition of *detailed balance* becomes

$$\int_A \int_B \pi(d\mathbf{x})T(\mathbf{x}, d\mathbf{x}') = \int_B \int_A \pi(d\mathbf{x}')T(\mathbf{x}', d\mathbf{x})$$

where again  $\pi(d\mathbf{x})$  is the limiting distribution and now  $T(\mathbf{x}, d\mathbf{x}')$  is known as the *transition kernel*, the probability of moving from a state  $\mathbf{x}$  to a region of parameter space  $d\mathbf{x}'$ . The detailed balance equation states that the probability of being in a region  $A$ , and moving to a region  $B$ , is the same as the probability of starting in region  $B$  and moving to  $A$ . It is this transition kernel which is to be constructed to maintain detailed balance even when moving between the different subspaces.

When the current state is  $\mathbf{x}$ , choose to make a move of type  $m$  which will take the current state to  $d\mathbf{x}'$  with probability  $q_m(\mathbf{x}, d\mathbf{x}')$  (this includes the probability of choosing this move type). The requirement now is to determine the acceptance probability  $A_m(\mathbf{x}, \mathbf{x}')$  to maintain detailed balance.

In appendix E the acceptance probability is derived as

$$A_m(\mathbf{x}, \mathbf{x}') = \min \left( 1, \frac{\pi(d\mathbf{x}')q_m(\mathbf{x}', d\mathbf{x})}{\pi(d\mathbf{x})q_m(\mathbf{x}, d\mathbf{x}')} \right)$$

It remains to explain how to use this in practice; in particular how to construct the proposals to ensure that  $\pi(d\mathbf{x})q_m(\mathbf{x}, d\mathbf{x}')$  does indeed have finite density with respect to a symmetric measure, as required in the derivation.

This is most easily done by constructing the proposals such that the dimensionality of  $\pi(d\mathbf{x}')q_m(\mathbf{x}', d\mathbf{x})$  is equal to the dimensionality of  $\pi(d\mathbf{x})q_m(\mathbf{x}, d\mathbf{x}')$  (even though  $\mathbf{x}$  and  $\mathbf{x}'$  are of differing dimensionality), by making the  $q_m$ ’s functions of a number of new random variables, which are introduced to match the dimensions. In many cases the dimensionality will be taken to be the larger of the dimensionalities of  $\mathbf{x}$  and  $\mathbf{x}'$ , so that in one direction

no new random variables will be needed. Thus if  $\mathbf{x}'$  is of higher dimension than  $\mathbf{x}$ ,  $\mathbf{x}'$  is made to be some function of  $\mathbf{x}$  and  $u$ , the new random variables. This gives the acceptance probabilities as

$$\min \left( 1, \frac{p(k+1, \theta^{(k+1)} | \mathbf{i})}{p(k, \theta^{(k)} | \mathbf{i}) q_k(u^{(k)})} \left| \frac{\partial(\theta^{(k+1)})}{\partial(\theta^{(k)}, u^{(k)})} \right| \frac{p(\text{choosing to propose the reverse jump})}{p(\text{choosing to propose the forwards jump})} \right)$$

where  $q_k(\cdot)$  is the probability density function of the new random variables,  $\mathbf{i}$  the data (in this case the degraded image) and  $\left| \frac{\partial(\theta^{(k+1)})}{\partial(\theta^{(k)}, u^{(k)})} \right|$  is the Jacobian of the transformation from  $\{\theta^{(k)}, u^{(k)}\}$  to  $\{\theta^{(k+1)}\}$ . This is used later to derive the acceptance probabilities for the line scratch detector.

## 5.3 Line Scratch Removal

### 5.3.1 Introduction

In this section the problem of restoring motion picture frames suffering from ‘line scratch’ degradation is addressed. These degradations are visible as bright or dark lines on the frames. Here we use a similar style of approach to that used for the problem of restoring frames with ‘blotch’ type defects. In this chapter the approach of using a detector to identify which, if any, areas of the frame are degraded is again adopted. A restoration algorithm is then applied only at the affected areas. For the ‘blotch’ detection algorithm in chapter 3 an approach was adopted which determined the presence or absence of a blotch at each pixel. For the ‘line scratch’ problem much more information about the *structure* of the defects is available and so an approach which tries to find these structures within the frames is used. This has parallels with earlier work on deformable templates [104, 80]; the work of Storvik [97], where a probabilistic approach is taken is most similar to the material in this chapter.

To the best of the author’s knowledge there is no material in the literature which addresses the problem of line scratches. Kokaram [58] has developed a method based on the Hough Transform [88] which can detect line scratches. It is, however, very susceptible to false detections if there are vertical features in the image which are not line scratches. This approach is discussed further in section 5.4.

### 5.3.2 A Model for Line Scratches

As discussed above, line scratches have a very strong structure and effect on the appearance of the frame, which must be incorporated into the detection algorithm. Each scratch can be parameterised spatially by its starting position,  $y$ , and its width  $w$ . (Here we assume

that the line extends the full height of the frame and is vertical; relaxing these constraints is straightforward.) The visual effect of the line scratch can be determined by considering the physical mechanism of the scratch's formation.

The scratches are formed by the film material running against an object in the camera or projection equipment. If the abrasion causes the removal of a proportion of the thickness of the emulsion, and the silver particles are uniformly distributed within the emulsion, then the effect is to reduce the density by a multiplicative factor. If the line scratch is formed on a positive print then reducing the thickness of the emulsion will result in a bright line; if the line scratch affects a negative print then in the subsequent positive copy a dark line will be visible. Clearly the same algorithm can be used to detect both types, by applying it to both positive and negative versions of the same image. Dark lines are more common, and so the factor by which the original grey level at the line scratch location has been multiplied is parameterised by calling it the 'depth',  $d$ , where  $0 \leq d < 1$ , and in practice we take  $0 \leq d < d_{max}$  where  $d_{max}$  is the largest value which results in a visible line scratch. (In the experiments described later  $d_{max} = 0.95$  was used.)

This discussion allows us to derive the likelihood function for the observed frame, given the line scratch parameters — it is simply the image model chosen, with the line scratch areas modified as described above. This is given in detail in the next section.

### 5.3.3 Constructing the Reversible Jump Sampler

To perform inference with regard to the number of line scratches and the scratches' parameters, it is necessary to construct a sampler which has as its invariant distribution  $p(k, \{y, w, d\}^{(k)} | \mathbf{i})$ , where  $k$  is the number of scratches,  $\{y, w, d\}^{(k)}$  are the  $k$  triples of parameters for the  $k$  scratches, and  $\mathbf{i}$  is the image data. This can be written using Bayes' Theorem as

$$p(k, \{y, w, d\}^{(k)} | \mathbf{i}) = p(\mathbf{i} | k, \{y, w, d\}^{(k)}) p(\{y, w, d\}^{(k)} | k) p(k) / p(\mathbf{i}) \quad (5.1)$$

So, up to a normalising constant, the target distribution is specified by the likelihood, the prior on the line scratches' parameters conditioned on the number of scratches, and the prior on the number of scratches.

#### The likelihood

As discussed briefly above, the effect of the line scratch is to multiply the original grey scale value by the factor  $d$ . Thus the likelihood is the probability of observing the image,

where the affected areas have their values multiplied by  $d$ . In this work, for computational reasons, we choose to use the simplest image model consistent with accurate detection. This turned out to be the simple model that assumes that the image is a sample from a space-varying Gaussian process, where the means and variances are estimated from the image data in a block-based manner. Then, when a pixel lies within a line, it is taken to be drawn from  $N(d \times \mu, d \times \sigma)$  rather than from  $N(\mu, \sigma)$ . It is acknowledged that the presence of line scratches on the image will tend to reduce the estimated values of  $\mu$ , and raise the estimated values of  $\sigma$ . This will weaken the detection, but in the experiments in section 5.4 it was found to be a sufficiently robust assumption.

### The priors

The prior on  $k$  gives information about the expected number of line scratches present. This was taken to be a Poisson distribution, conditioned on the maximum possible number of line scratches being known. The parameter  $\lambda$  gives the expected number of scratches.

$$p(k) \propto \exp(-\lambda) \frac{\lambda^k}{k!}, \quad k = 0, 1, 2, \dots, k_{max}$$

Note that this *includes* the case  $k = 0$  to deal with the situation of no line scratches being present in the same framework.

The remaining priors, on the line scratches' parameters, are all conditioned on  $k$ .

The prior on  $\{y\}$  tells us where the line scratches are expected to appear in the image. The lines are independent and may occur at any position, so a uniform prior on  $y$  is used.

For simplicity the prior on the widths of the line scratches was taken to be a uniform distribution, with each  $w$  being drawn independently from  $[1, w_{max}]$ . The 'depth' parameters were also taken to be from a uniform distribution on  $[0, d_{max}]$ , where  $d_{max} = 0.95$  as discussed above.

### The proposal and acceptance probabilities

To finally construct the reversible jump sampler two things must be achieved – the proposal distributions for each move type, conditioned on  $k$  must be specified, and the corresponding acceptance probabilities determined.

The available move types are

1. to change the position of an existing line

2. to change the width of an existing line
3. to change the depth of an existing line
4. to introduce a new line – a *birth*
5. to remove an existing line – a *death*

The proposal probabilities are respectively  $p_k$ ,  $w_k$ ,  $d_k$ ,  $B_k$ ,  $D_k$  where  $p_k + w_k + d_k + B_k + D_k = 1$ . In [40] Green proposed choosing  $B_k$  and  $D_k$  to be proportional to the stationary distribution based just on the prior for the number of scratches, *i.e.*

$$B_k = c \min \left( 1, \frac{p(k+1)}{p(k)} \right)$$

$$D_k = c \min \left( 1, \frac{p(k)}{p(k+1)} \right)$$

where  $c$  was chosen to be the largest value that gave  $B_k + D_k \leq 0.9$  for all  $k$ . That is, if the target distribution was  $p(k)$  as defined above, then  $B_k$  and  $D_k$  are chosen to give an acceptance probability of one for the simple sampler which just moves through the index of the model orders. The remaining probability was distributed equally between the remaining moves, conditioned on  $B_{k_{max}} = 0$  and  $p_0 = w_0 = d_0 = D_0 = 0$ . For the line scratch detection problem this resulted in an inefficient sampler; better results were obtained by decreasing the chances of proposing a birth, and significantly decreasing the chances of proposing a death. The reasons for doing this will be made clear in the discussion of the action of the algorithm in section 5.4.

At each iteration, once the type of move has been selected by sampling from the distribution over the available move types, the acceptance probability must be determined. For proposed changes to the position, width and depth of an existing line scratch, changes which do not involve changing the dimensionality of the parameter space, the acceptance probabilities are as for the standard Metropolis Hastings algorithm, given by equation 2.2 and repeated here for clarity.

$$A(\mathbf{x}, \mathbf{x}') = \min \left( 1, \frac{\pi(\mathbf{x}')q(\mathbf{x}', \mathbf{x})}{\pi(\mathbf{x})q(\mathbf{x}, \mathbf{x}')} \right)$$

**1. Position** A new value for the position of one of the scratches is drawn from a distribution uniform on  $[y_j - \Delta y, y_j + \Delta y]$ , constrained such that the lines do not overlap<sup>1</sup>. In the experiments  $\Delta y = 0.3$  was used. The acceptance probability from the posterior

---

<sup>1</sup>This constraint means that we do not sample exactly from  $p(k, \{y, w, d\}^{(k)} | \mathbf{i})$

distribution is

$$A(y_i, y'_i) = \min(1, (\text{likelihood ratio}))$$

**2. Width** A new value for the width of scratch  $j$  is chosen uniformly from  $[w_j - \Delta w, w_j + \Delta w]$  (conditioned on the minimum and maximum permissible values of  $w$ .)  $\Delta w = 0.3$  was used in the experiments. Because the prior on the widths is uniform the acceptance probability is

$$A(w_j, w'_j) = \min(1, (\text{likelihood ratio}))$$

**3. Depth** A new depth for scratch  $j$  is again proposed by perturbing the current value by drawing from  $[d_j - \Delta d, d_j + \Delta d]$ , again conditioned on  $d > 0, d < d_{max}$ . The experiments used  $\Delta d = 0.1$ . The acceptance probability is also

$$A(d_j, d'_j) = \min(1, (\text{likelihood ratio}))$$

**4. Birth** The birth step increases the number of parameters by three – a value must be assigned to the  $\{y, w, d\}$  triple associated with the new line scratch. Because the nature of the line scratch detection problem is such that each scratch occurs independently of any other, the existing parameter values cannot be used to guide the generation of the new parameters, and so  $\{y', w', d'\}$  are drawn from their respective priors, conditioned on the new line not overlapping any existing one.

In section 5.2 the acceptance probability for a change in state from  $\mathbf{x}$  to  $\mathbf{x}'$  involving a change in dimensionality was derived as

$$A(\mathbf{x}, \mathbf{x}') = \min \left( 1, \frac{p(k+1, \theta^{(k+1)} | \mathbf{i})}{p(k, \theta^{(k)} | \mathbf{i}) q_k(u^{(k)})} \left| \frac{\partial(\theta^{(k+1)})}{\partial(\theta^{(k)} u^{(k)})} \right| \frac{D_{k+1}}{k+1} \frac{1}{B_k} \right) \quad (5.2)$$

where the final two terms,  $D_{k+1}/(k+1)$  and  $1/B_k$  are the probabilities of proposing the *specific* changes in dimensionality. (The choice of *which* of the  $k+1$  lines to remove is made randomly.) This is essentially

(the probability of jumping *back*) divided by (the probability of jumping *there*).

Figure 5.2 illustrates this proposed jump.

In terms of the line scratch detection problem these are

$$p(k+1, \theta^{(k+1)} | \mathbf{i}) = p(\mathbf{i} | k+1, \{y, w, d\}^{(k+1)}) p(\{y, w, d\}^{(k+1)} | k+1) p(k+1)$$

$$p(k, \theta^{(k)} | \mathbf{i}) = p(\mathbf{i} | k, \{y, w, d\}^{(k)}) p(\{y, w, d\}^{(k)} | k) p(k)$$

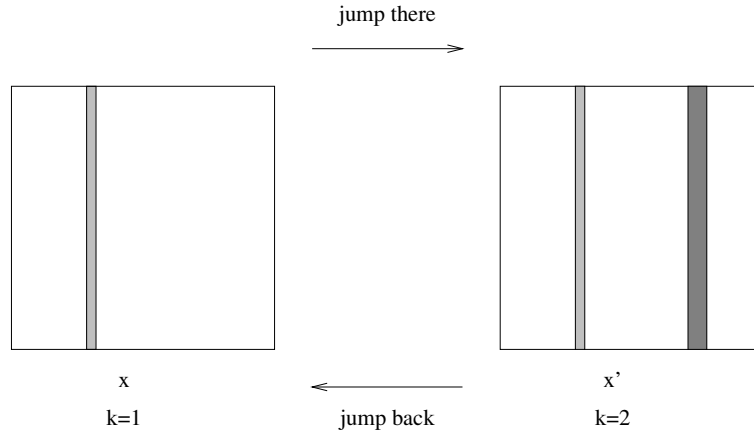


Figure 5.2: Illustration of the jumps involved in the birth step

and

$$q_k(u^{(k)}) = p(y', w', d') = \frac{1}{L} \times \frac{1}{W} \times \frac{1}{d_{max}}$$

from the uniform priors on  $y$ ,  $w$  and  $d$  which these proposals are drawn from. The Jacobian term,  $\left| \frac{\partial(\theta^{(k+1)})}{\partial(\theta^{(k)}u^{(k)})} \right| = 1$  as each of the variables in the  $k + 1$  dimensional case is equivalent to exactly one of either the existing variables or the newly introduced variables.

Substituting into equation 5.2 results in the acceptance probability for the birth move being

$$A(\mathbf{x}, \mathbf{x}') = \min(1, R)$$

where

$$R = \frac{(\text{likelihood})^{(k+1)}}{(\text{likelihood})^{(k)}} \times \frac{p(k+1)}{p(k)} \times \frac{D_{k+1}}{(k+1)B_k}$$

**5. Death** The death move must be constructed to undo exactly the birth move. In this case this involves choosing a line to remove, at random. A similar derivation to that above gives the acceptance probability as

$$A(\mathbf{x}, \mathbf{x}') = \min(1, R^{-1})$$

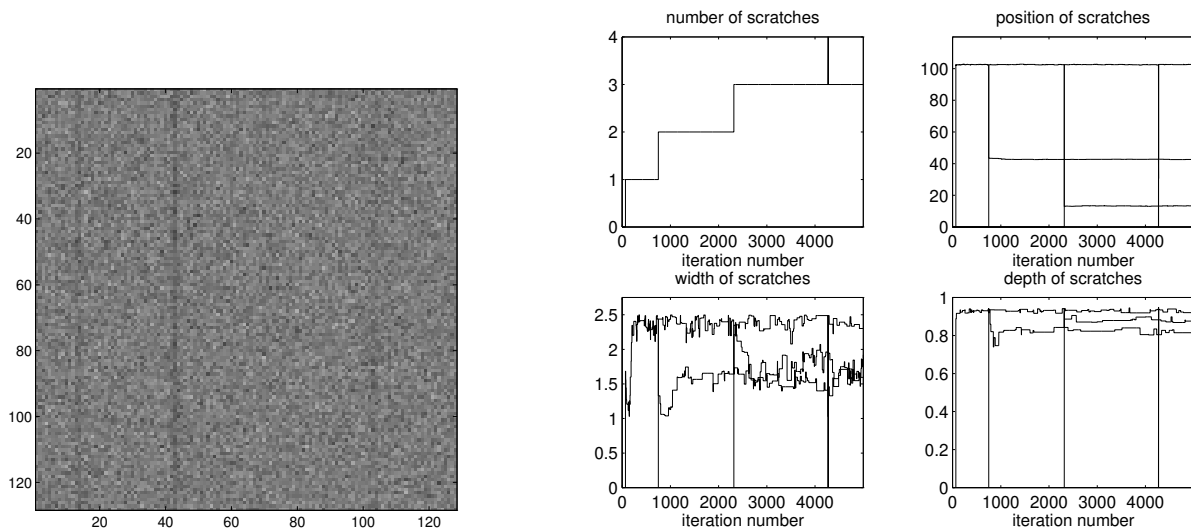


Figure 5.3: Artificial image

Figure 5.4: Results

## 5.4 Results

The line scratch detection algorithm was applied to three examples. The first was a best case artificial image, where the image statistics matched the likelihood model described in section 5.3.3 exactly, and line scratches were added to match the line scratch model of section 5.3.2. The other two examples are frames digitised from real motion picture material. One exhibits a dark line, the other a bright line. The action of the detection algorithm on these examples is discussed, and used to suggest methods of developing a much faster, but somewhat more *ad-hoc* algorithm.

### Best case frame

Figure 5.3 shows an artificial image, where each pixel in the background is drawn independently from a normal distribution. Three line scratches have been added. Two of these are clearly visible, the third (to the right of the frame) much less so. Table 5.1 lists the parameters of the line scratches. Figure 5.4 shows the results for running the reversible jump MCMC detector on this image starting from  $k = 0$ .

The correct number of line scratches has been identified, and their parameters estimated accurately - the parameter values estimated from forming a histogram of the samples are also listed in table 5.1.

Figure 5.4 requires more comment to explain the action of the algorithm. Because of the nature of the line scratches, that they are isolated, independent features, the algorithm functions much as a random search - lines are proposed in the birth step and are not accepted until one happens to be proposed at the location of an actual scratch. Once

Scratch Number	1	2	3
true y	13.3	42.8	102.6
true w	1.85	1.47	2.41
true d	0.88	0.84	0.93
estimated y	13.3	42.7	102.5
estimated w	1.75	1.55	2.38
estimated d	0.88	0.82	0.93

Table 5.1: Actual and estimated line scratch parameters for the artificial example

the line is accepted, subsequent proposals to alter its parameters cause their values to explore the posterior distribution – the estimates in table 5.1 are maximum a posteriori values estimated by smoothing the histograms of the sample values. Proposals to remove a true line were always rejected. Figure 5.4 also illustrates that it is possible for lines to be accepted which are not true lines, but these are rapidly removed.

This points the way to developing a very efficient and accurate line scratch detection algorithm. The Hough transform based detector of Kokaram [58], which suffers badly from false detections, can be used to propose  $k_{max}$  possible line scratches. These can then be tested using a version of the birth acceptance probability, suitably modified to take account of the new proposal distributions, to determine whether the proposed lines are indeed line scratches. The parameters of the confirmed line scratches can then be optimised using gradient descent on the posterior. This is left for future work.

### Cricket

Figure 5.5 shows a frame from the ‘cricket’ sequence. There is a very clear dark line scratch to the left of the stumps. The frame also has much strong vertical information, and this is the cause of many false detections using Kokaram’s [58] detector. Running the detection algorithm on this frame resulted in a line being introduced after  $\simeq 1400$  iterations, and figures 5.7, 5.8, 5.9 are the histograms of the parameter values for the next  $\simeq 6500$  iterations. The histograms for the position and width variables show that these are quite well defined. More iterations would have resulted in smoother histograms.

Figure 5.9, the histogram of the values of  $d$ , does not indicate that this parameter is well defined by the data. This is confirmed by trying to effect a restoration by inverting the line scratch model with parameters as estimated from the histograms. The restoration is unsatisfactory and is not shown. The line scratch model is adequate for detection but not for restoration. If naturally occurring line scratches are viewed in detail, the dark



Figure 5.5: Original frame from cricket sequence

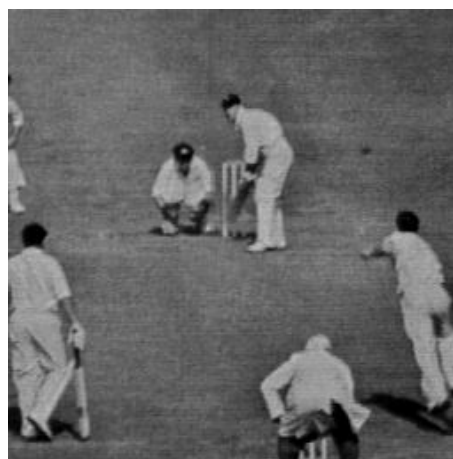


Figure 5.6: Restored frame

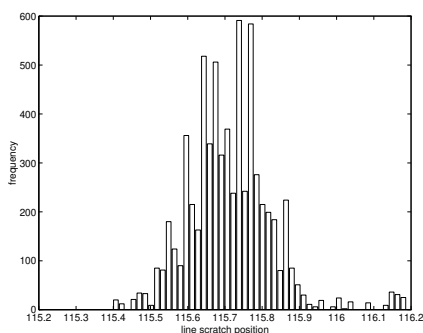


Figure 5.7: Histogram of estimated position variable

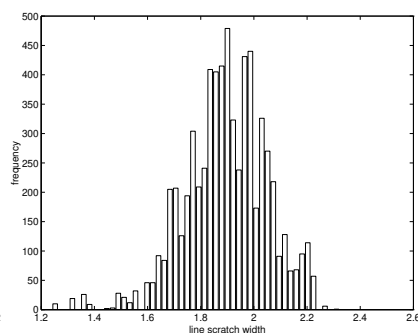


Figure 5.8: Histogram of estimated width variable

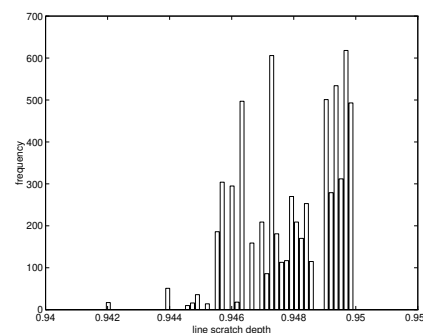


Figure 5.9: Histogram of estimated depth variable

scratch tends to have a bright area either side of it, and vice versa for a bright scratch. The restoration in figure 5.6 was performed using a spatial version of the IMFA interpolator of section 3.3.5 to replace the areas affected by the line scratch, after detection using the reversible jump detector.

### Knight

Figure 5.10 shows a frame from the ‘knight’ sequence, which is degraded by the presence of a bright line. Applying the line scratch detection algorithm to an ‘negative’ version of this image correctly located the single line at position  $\hat{y} = 166.6$ , with width  $\hat{w} = 1.2$ . Figure 5.11 shows the results of restoring this frame as described above.



Figure 5.10: Frame from the ‘knight’ sequence



Figure 5.11: Restored frame

## 5.5 Summary

In this chapter we have presented the theory of reversible jump MCMC samplers, which enable distributions defined over an unknown number of parameters to be sampled. This theory was used as the computational engine behind the development of an algorithm for the detection of ‘line scratch’ defects on motion picture material. The workings of this algorithm on artificially and naturally degraded images was explored. The detection algorithm proved accurate in determining the number and locations of the line scratches, and enabled visually pleasing restorations to be achieved.

---

## CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

---

This thesis has addressed a number of problems concerned with image sequence restoration. Markov Random Field image models have been applied to the problem of detection and removal of blotch type defects from archived motion picture material. An algorithm for the detection and removal of line scratch defects has also been developed. We have discussed the necessary theoretical material, and in some cases used this theory to develop effective algorithms outside the main thrust of the material.

Markov chain Monte Carlo methods were discussed. The use of these algorithms is becoming more widespread, especially in the area of Bayesian inference. The example of marginalised Bayesian inference presented was a simple one, but representative of a useful class of problems. The algorithm for the self structuring of generalised single layer networks which was developed was found to be effective. This algorithm does, however, depend on two parameters,  $T$  and  $\beta$ . In the examples in chapter 2 these were chosen on the basis of experience and experiment. The optimal, automatic choice of these parameters is an area for future research. Consideration of the maximum and minimum sum squared error resulting from a full model and a model containing only one term can be used to estimate  $T$ . It is believed that developments of the work in [46] could be used to bound  $\Delta\varepsilon$ , and thus estimate  $\beta$ .

The scratch detection algorithm presented in chapter 3 was based on the theory of Markov Random Fields. It used the same heuristic as had previously been used to detect scratches, but incorporated the spatial continuity of the scratch to improve the detection capability. The estimate used was the Bayesian maximum a-posteriori estimate. This detector relied on a number of parameters, and heuristic arguments were given which estimated useful values. In the Bayesian paradigm we can place prior distributions on these parameters, and estimate their values directly from the data, via the joint posterior  $p(\mathbf{d}, \alpha, \beta_1, \beta_2 | \mathbf{x}_1, \mathbf{x}_2)$ . Estimating the hyperparameters in this way, and defining physically meaningful priors is an obvious area for future work.

The interpolation algorithm presented, whilst providing visually pleasing restorations, is not very sophisticated. Again the joint estimation of the parameters of an adaptive model together with the interpolation is an area in need of development.

In the experiments on the combined motion estimation/occlusion detection algorithm of chapter 4, the experiments were performed at one resolution level. This is acceptable for image sequences where the motion is very limited. For sequences exhibiting large displacements this would not be a reasonable estimation strategy, as the state space would need to be very large to encompass the large number of possible displacements, and the possibility of false matches would be very high. Some form of multiresolution processing would be needed. Unfortunately using MRF models in a consistent multiresolution framework is extremely complex [36], and in many cases ad hoc formulations are used [61]. Recently some work has been done on simpler consistent forms of multiresolution estimation [62], but this relies on the state space being numbers and not labels, so that arithmetic on the state values is permissible. The consistent extension of the combined displacement/occlusion estimation framework is a topic for future research.

In the discussion of the line scratch detection algorithm in chapter 5 it was demonstrated that the ‘depth’ part of the model was not very accurate. It would be useful if the precise effect on the film of a line scratch could be determined, possibly from more detailed study of the mechanism of the scratches’ production. This would allow improvements to the accuracy of the detector, and improve the quality of the restorations. Indeed, in the results in chapter 5 a small amount of vertical structure remains. The development of an interpolator tuned to the problem of removing long, narrow features is another area for future work. The development of the fast, ad-hoc line scratch detection algorithm mentioned in chapter 5 is also left for future work.

In the area of motion picture restoration there are many other unsolved problems. Degradation caused by camera shake is very obvious, and should be amenable to a signal processing solution. The whole area of *colour* image sequence processing has only had preliminary investigations performed [100, 91]. How best to use the colour information in the detection algorithms developed is an area for future research, as is how to use the colour information to improve the accuracy of the motion estimation algorithm. Colour image modelling for interpolation problems is a difficult area in need of much research effort.

---

**BIBLIOGRAPHY**

---

- [1] I.M. Abdelquader, S.A. Rajala, W.E. Snyder, and G.L. Bilbro. Energy minimization approach to motion estimation. *Signal Processing*, 28:291–309, 1992.
- [2] B. Alp, P. Haavisto, T. Jarske, and K. Öistämö. Median-based algorithms for image sequence processing. *SPIE VCIP*, 1360:122–134, 1990.
- [3] G.R. Arce. Multistage order statistic filters for image sequence processing. *IEEE Trans Signal Processing*, 39(5):1146–1163, May 1991.
- [4] J.L. Barron, D.J. Fleet, and S.S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.
- [5] J.M. Bernardo and A.F.M. Smith. *Bayesian Theory*. Wiley, 1994.
- [6] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *J. Royal Statistical Soc. B*, 36:192–326, 1974.
- [7] J. Besag. On the statistical analysis of dirty pictures. *J. Royal Statistical Soc. B*, 48(3):259–302, 1986.
- [8] M.R. Bhatt and U.B. Desai. Robust image restoration algorithm using Markov random field model. In *Proceedings of the International Symposium on Circuits and Systems*, pages 2473–2476, 1992.
- [9] J. Biemond, L. Looijenga, D.E. Boekee, and R.H.J.M Plompen. A pel-recursive Wiener based displacement estimation algorithm. *Signal Processing*, 13:399–412, 1987.
- [10] M. Bierling. Displacement estimation by hierarchical block matching. *SPIE: Visual Communications and Image Processing*, 1001:942–951, 1988.
- [11] G.L. Bilbro, W.E. Snyder, and R.C. Mann. Mean-field approximation minimises relative entropy. *J. Opt. Soc. Am.*, 8(2):290–294, February 1991.
- [12] M. Bilge Alp and Y. Neuvo. 3-dimensional median filters for image sequence processing. In *Proceedings of ICASSP 91*, pages 2917–2920, 1991.

- 
- [13] L. Böröczky. *Pel-recursive motion estimation for image coding*. PhD thesis, Technical University of Budapest, 1990.
- [14] P. Bouthemy. A maximum-likelihood framework for determining moving edges. *IEEE Trans PAMI*, 11(5):499–511, May 1989.
- [15] J. Boyce. Noise reduction in image sequences using adaptive motion compensated frame averaging. In *Proceedings of IEEE ICASSP '92*, pages 461–464, 1992.
- [16] D. Burton, G.J. Moore, and W.J. Fitzgerald. Bayesian model selection and parameter estimation. *Signal Processing V: Theories and Applications*, pages 221–224, 1990.
- [17] B.P. Carlin and S. Chib. Bayesian model choice via Markov chain Monte Carlo. Technical report, University of Minnesota, 1994.
- [18] B. Chalmond. Image restoration using an estimated Markov model. *Signal Processing*, 15:115–129, 1988.
- [19] D. Chandler. *Introduction to Modern Statistical Mechanics*. OUP, 1987.
- [20] P. Clifford. Markov random fields in statistics. In G. Grimmett and D.J. Welsh, editors, *Disorder in Physical Systems*. Clarendon, Oxford, 1990.
- [21] L. Cui, M.A. Tanner, D. Sinha, and W.J. Hall. Comment: monitoring convergence of the gibbs sampler: Further experience with the gibbs stopper. *Statistical Science*, 7(4):483–486, 1992.
- [22] E. Dubois and J. Konrad. Estimation of 2-d motion fields from image sequences with application to motion-compensated prediction. In M.I. Sezan and R.L. Lagendijk, editors, *Motion Analysis and Image Sequence Processing*, chapter 3, pages 53–87. Kluwer, 1993.
- [23] E. Dubois and S. Sabri. Noise reduction in image sequences using motion compensated temporal filtering. *IEEE Trans Communications*, 32:826–831, 1984.
- [24] A.T. Erdem, M.I. Sezan, and M.K. Ozkan. Motion compensated multiframe Wiener restoration of blurred and noisy image sequences. In *Proceedings of ICASSP*, pages III – 293–296, 1992.
- [25] S.E. Fahlman and C. Lebiere. The Cascade-Correlation Learning Architecture. In D.S. Touretzky, editor, *Neural Information Processing Systems 2*, pages 524–532. Morgan-Kaufmann, 1990.

- 
- [26] W.J. Fitzgerald. Bayesian inference and signal processing. Technical Report CUED/F-INFENG/TR80, Cambridge University Engineering Department, 1991.
- [27] A.D.M. Garvin and P.J.W. Rayner. Fast Iterative Self-structuring and Training of Artificial Neural Networks. Submitted to *Neural Computation*, January 1994.
- [28] D. Geiger and F. Girosi. Parallel and deterministic algorithms from MRF's: Surface reconstruction. *IEEE Trans PAMI*, pages 401–412, 1991.
- [29] A.E. Gelfand and A.F.M. Smith. Sampling-based approaches to calculating marginal densities. *J. American Statistical Association*, 85:398–409, 1990.
- [30] A. Gelman, W. Gilks, and G.O. Roberts. Efficient metropolis jumping rules. Technical Report 94-10, Statistical Laboratory, University of Cambridge, 1994.
- [31] A. Gelman and D.B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4):457–511, 1992.
- [32] D. Geman and G. Reynolds. Constrained restoration and the recovery of discontinuities. *IEEE Trans PAMI*, 14(3):367–383, March 1992.
- [33] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Trans PAMI*, 6(6):721–741, November 1984.
- [34] S. Geman, D.E. McClure, and D. Geman. A nonlinear filter for film restoration and other problems in image restoration. *CVGIP: Graphical Models and Image Processing*, 54(4):281–289, July 1992.
- [35] C.J. Geyer. Practical markov chain monte carlo. *Statistical Science*, 7(4):473–483, 1992.
- [36] B. Gidas. A renormalization group approach to image processing problems. *IEEE Trans PAMI*, 11(2):164–180, February 1989.
- [37] B. Gidas. Parameter estimation for Gibbs distributions from fully observed data. In R. Chellappa and A. Jain, editors, *MRFs, Theory and Applications*. Academic Press, 1993.
- [38] S.J. Godsill. *The restoration of degraded audio signals*. PhD thesis, Cambridge University Engineering Department, 1993.
- [39] P.J. Green. Monte carlo methods: an overview. In *Proceedings of the IMA Conference on Complex Stochastic Systems and Engineering*, September 1993.

- 
- [40] P.J. Green. Reversible jump MCMC computation and Bayesian model determination. Presented at the workshop on Model Criticism in Highly Structured Stochastic Systems, Wiesbaden, September 1994.
- [41] D.M. Greig, B.T. Porteous, and A.H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society - B*, 51(2):271–279, 1989.
- [42] U. Grenander and M.I. Miller. Representations of knowledge in complex systems. *JRSS-B*, 56(4):549–603, 1994.
- [43] W.K. Hastings. Monte Carlo sampling methods using Markov Chains and their applications. *Biometrika*, 57:97–109, 1970.
- [44] F. Heitz and P. Bouthemy. Multimodal estimation of discontinuous optical flow using markov random fields. *IEEE Trans PAMI*, 15(12):1217–1232, December 1993.
- [45] H.P. Hiriannaiah, G.L. Bilbro, and W.E. Snyder. Restoration of piecewise-constant images by mean-field annealing. *J. Opt. Soc. Am.*, pages 1901–1912, 1989.
- [46] S.B. Holden. *On the theory of generalization and self structuring in linearly weighted connectionist networks*. PhD thesis, Cambridge University, 1993.
- [47] S.B. Holden and P.J.W. Rayner. Generalization and PAC learning: some new results for the class of generalized single layer networks. *IEEE Transactions on Neural Networks*, 1993.
- [48] R.A. Howard. *Dynamic Probabilistic Systems, Volume 1: Markov Models*. John Wiley, NY, 1971.
- [49] T. Huang. *Image Sequence Analysis*. Springer-Verlag, 1981.
- [50] M. Hurn and C. Jennison. Multiple-site updates in maximum a-posteriori and marginal posterior modes image estimation. Technical Report 93:03, University of Bath, April 1993.
- [51] E. Ising. . *Zeitschrift Physik*, 31:253, 1925.
- [52] A.K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall, 1989.
- [53] H. Jefferys. *Theory of probability*. OUP, 1939. Reprinted 1985.
- [54] J.G. Kemeny and J.L. Snell. *Finite Markov Chains*. Springer-Verlag, New York, 1960. reprinted 1976.

- 
- [55] S. Kirkpatrick, C.D. Gellat, and M.P. Vecchi. Optimisation by simulated annealing. *Science*, 220(4598):671–680, May 1983.
- [56] A.C. Kokaram. *Motion Picture Restoration*. PhD thesis, Cambridge University, 1993.
- [57] A.C. Kokaram. 3D Wiener filtering for noise suppression in motion picture sequences using overlapped processing. In *Signal Processing VII: Theories and Applications*, pages III – 1780–1783. EUSIPCO, 1994.
- [58] A.C. Kokaram. Personal communication. 1995.
- [59] A.C. Kokaram, R.D. Morris, W.J. Fitzgerald, and P.J.W. Rayner. Detection of missing data in image sequences. *IEEE Trans Image Processing*, November 1995.
- [60] A.C. Kokaram and P.J.W. Rayner. A system for the removal of impulsive noise in image sequences. *SPIE: Visual Communications and Image Processing*, 1818:322–331, 1992.
- [61] J. Konrad and E. Dubois. Bayesian estimation of motion vector fields. *IEEE Trans PAMI*, 14(9), September 1992.
- [62] J.-M. Laferté, P. Pérez, and F. Heitz. Global non-linear multigrid optimization for image analysis tasks. In *Proceedings of ICASSP*, pages V533–536. IEEE, 1994.
- [63] Y. LeCun, J.S. Denker, and S.A. Solla. Optimal Brain Damage. In D.S. Touretzky, editor, *Neural Information Processing Systems 2*, pages 598–605. Morgan-Kaufmann, 1990.
- [64] P.M. Lee. *Bayesian statistics, an introduction*. OUP, 1989.
- [65] J.S. Lim. *Two-dimensional signal and image processing*. Prentice-Hall, 1990.
- [66] D. Lowe. Adaptive radial basis function nonlinearities and the problem of generalization. *Proc. First IEE International Conference on Artificial Neural Networks*, pages 171–175, 1989.
- [67] M.R. Lynch and P.J.W. Rayner. Properties and implementation of the nonlinear vector space connectionist model. In *Proc. IEE First International Conference on Artificial Neural Networks*, London, 1989.
- [68] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. *Journal of the American Statistical Association*, 82(397):76–89, March 1987.

- 
- [69] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.
- [70] H. Nagel. On a constraint equation for the estimation of displacement rates in image sequences. *IEEE Trans PAMI*, 11:13–30, January 1989.
- [71] R. Neal. Probabilistic inference using Markov Chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, 1993.
- [72] A. Nieminen, P. Heinonen, and Y. Neuvo. A new class of detail-preserving filters for image processing. *IEEE Trans PAMI*, 9(1):74–90, January 1987.
- [73] P.H. Peskun. Guidelines for choosing the transition matrix in Monte Carlo methods using Markov chains. *J. Computational Physics*, 40:327–344, 1981.
- [74] D.B. Phillips and A.F.M. Smith. Bayesian model comparison via jump diffusions. Technical report, Imperial College of Science, Technology and Medicine, September 1994.
- [75] K.J. Pope. *Time Series Analysis*. PhD thesis, Cambridge University, 1993.
- [76] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical recipes in C (2nd ed.)*, pages 77–78. Cambridge University Press, Cambridge, 1992.
- [77] A.E. Raftery and S.M. Lewis. The number of iterations, convergence diagnostics and generic metropolis algorithms. Technical report, University of Washington, 1992.
- [78] B. D. Ripley. *Statistical inference for spatial processes*. CUP, 1988.
- [79] B.D. Ripley. *Stochastic Simulation*. Wiley, New York, 1987.
- [80] B.D. Ripley and A.I. Sutherland. Finding spiral structures in images of galaxies. *Philosophical Transactions of the Royal Society - A*, 332:477–485, 1990.
- [81] G.O. Roberts and N.G. Polson. On the geometric convergence of the Gibbs sampler. *JRSS-B*, 56:377–384, 1994.
- [82] G.O. Roberts and A.F.M. Smith. Simple conditions for the convergence of the gibbs sampler and metropolis-hastings algorithms. *Stochastic Processes and their Applications*, 49(2):207–216, 1994.
- [83] R.D. Rosenkrantz, editor. *E. T. Jaynes: Papers on probability, statistics and statistical physics*. Kluwer, 1989.

- 
- [84] J.S. Rosenthal. Theoretical rates of convergence for markov chain monte carlo. In *Proceedings of Interface '94*, 1994.
- [85] J.J.K Ó Ruanaidh. *Numerical Bayesian methods applied to signal processing*. PhD thesis, University of Cambridge, 1994.
- [86] J.J.K. Ó Ruanaidh and W.J. Fitzgerald. The restoration of degraded audio recordings using the Gibbs sampler. Technical Report CUED/F-INFENG/TR153, Cambridge University Engineering Department, 1993.
- [87] A. Sankar and R. J. Mammone. Growing and Pruning Neural Tree Networks. *IEEE Trans. Computers*, 42(3):291–299, 1993.
- [88] R.J. Schalkoff. *Digital Image Processing and Computer Vision*. Wiley, 1989.
- [89] L. Scharf. *Statistical Signal Processing*, pages 53–54. Addison-Wesley, New York, 1991.
- [90] M. Schetzen. *The Volterra and Wiener Theories of Nonlinear Systems*. John Wiley and Sons, 1980.
- [91] R.R. Schultz and R.L. Stevenson. Stochastic modelling and estimation of multispectral image data. In *Proceedings of ICASSP*, pages V–373–376, 1994.
- [92] A.F.M. Smith and A.E. Gelfand. Bayesian statistics without tears: a sampling-resampling approach. *The American Statistician*, 46(2):84–88, May 1992.
- [93] A.F.M Smith and G.O. Roberts. Bayesian computation via the Gibbs sampler and related Markov-chain Monte-Carlo methods. *JRSS-B*, 55(1):3–23, 1993.
- [94] C. Stiller. Motion estimation for coding of moving video at 8kbit/sec with gibbs modelled vector field smoothing. *SPIE VCIP*, 1360:468–476, 1990.
- [95] R. Storey. Electronic detection and concealment of film dirt. Technical report, Research Department, Engineering Division, The British Broadcasting Corporation, 1985.
- [96] R. Storey. Electronic detection and concealment of film dirt. *SMPTE Journal*, pages 642–647, June 1985.
- [97] G. Storvik. A Bayesian approach to dynamic contours through stochastic sampling and simulated annealing. *IEEE Trans. PAMI*, 16(10):976–986, October 1994.
- [98] C.W. Therrien. *Decision, estimation and classification*. Wiley, NY, 1989.

- [99] L. Tierney. Markov chains for exploring posterior distributions. Technical Report 560, School of Statistics, University of Minnesota, 1991.
- [100] P.E. Trahanias and A.N. Venetsanopoulos. Multispectral image processing. In *Image Processing: Theory and Application*, pages 41–44. Kluwer, 1993.
- [101] R. Veldhuis. *Restoration of lost samples in digital signals*. Prentice-Hall, 1980.
- [102] A. Verri and T. Poggio. Motion field and optical flow: qualitative properties. *IEEE Trans PAMI*, 11(5):490–498, May 1989.
- [103] R. Young and N. G. Kingsbury. Video compression using lapped transforms form motion estimation/compensation and coding. *SPIE VCIP*, pages 276–288, 1992.
- [104] A.L. Yuille, P.W. Hallinan, and D.S. Cohen. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8(2):99–111, 1992.
- [105] J. Zhang and G.G. Hanauer. The application of mean field theory to image motion estimation. *IEEE Trans Image Processing*, 4(1):19–33, January 1995.

**Papers by the author**

R.D. Morris and W.J. Fitzgerald. Detection and Correction of Speckle Degradation in Image Sequences Using a 3D Markov Random Field. In *Proceedings of International Conference on Image Processing: Theory and Applications*, San Remo, June 1993.

R.D. Morris and W.J. Fitzgerald. Replacement Noise in Image Sequences, Detection and Interpolation by Motion Field Segmentation. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, Adelaide, April 1994.

R.D. Morris and W.J. Fitzgerald. Stochastic and Deterministic Methods in Motion Picture Restoration. In *Proceedings of International Workshop on Image Processing*, Budapest, June 1994.

R.D. Morris. Image Sequence Restoration via Gibbs Distributions. A dissertation submitted to the Trinity College Research Fellowship Competition, August 1994.

R.D. Morris and W.J. Fitzgerald. Discontinuous Motion and Occlusion Estimation - Theory and Application. To appear in *Proceedings of the International Conference for Young Computer Scientists*, Beijing, July 1995.

A.C. Kokaram, R.D. Morris, W.J. Fitzgerald and P.J.W. Rayner. Detection of Missing Data in Image Sequences. To appear in *IEEE Transactions on Image Processing*, November 1995.

A.C. Kokaram, R.D. Morris, W.J. Fitzgerald and P.J.W. Rayner. Interpolation of Missing Data in Image Sequences. To appear in *IEEE Transactions on Image Processing*, November 1995.

R.D. Morris and A.D.M. Garvin. Fast Probabilistic Self-structuring of Generalised Single Layer Networks. Submitted to *IEEE Transactions on Neural Networks*, April 1994.

---

## RESULTS USED IN PROVING THE CONVERGENCE OF REGULAR MARKOV CHAINS

---

The results of this appendix are used in proving the convergence of regular Markov chains.

Consider an  $r \times r$  transition matrix  $\mathbf{T}$ , whose smallest element is  $\eta > 0$ . Let  $\mathbf{x}$  be an  $r$  component column vector with minimum element  $m_0$  and maximum element  $M_0$ . Let  $m_1$  and  $M_1$  be the smallest and largest elements of  $\mathbf{T}\mathbf{x}$ .

1.  $M_1 \leq M_0$

If any of the elements of  $\mathbf{x}$  are less than  $M_0$ , then, since the rows of  $\mathbf{T}$  sum to one, the largest element of  $\mathbf{T}\mathbf{x}$  must be smaller than  $M_0$ . Equality occurs when all the elements of  $\mathbf{x}$  are equal to  $M_0$ . Specifically  $M_1 \leq M_0 - \eta(M_0 - m_0)$ .

2.  $m_1 \geq m_0$

Similarly, as the minimum element of  $\mathbf{T}\mathbf{x}$  will contain some contribution from  $M_0$ , giving  $m_1 \geq m_0 + \eta(M_0 - m_0)$ .

3.  $M_1 - m_1 \leq (1 - 2\eta)(M_0 - m_0)$

Combining the two limits above gives this result.

---

## TRAINING AND UPDATING THE VOLTERRA CONNECTIONIST MODEL

---

### B.1 Training the Volterra Connectionist Model (VCM)

Given a set of  $M$  training examples,  $\{\mathbf{x}^i, d_i : i = 1 \dots M\}$ , and a set of  $t$  nonlinear functions, taken from the  $V(n, k)$  expansion, make the following definitions

$$\begin{aligned}\mathbf{X}_t &= [\hat{\mathbf{x}}_t^1 \hat{\mathbf{x}}_t^2 \dots \hat{\mathbf{x}}_t^M] \\ \mathbf{d} &= [d_1 \ d_2 \ \dots \ d_M] \\ \mathbf{R}_t &= \mathbf{X}_t \mathbf{X}_t^T \\ \mathbf{p}_t &= \mathbf{X}_t \mathbf{d}^T \\ \varepsilon_t &= \sum_{i=1}^M (d_i - \mathbf{w}_t^T \hat{\mathbf{x}}_t^i)^2 \\ &= \mathbf{d} \mathbf{d}^T - 2 \mathbf{w}_t^T \mathbf{p}_t + \mathbf{w}_t^T \mathbf{R}_t \mathbf{w}_t\end{aligned}$$

where  $\hat{\mathbf{x}}_t^i$  denotes the nonlinear extension of the  $i^{\text{th}}$  training vector by the  $t$  terms chosen from the expansion. Taking the derivative of  $\varepsilon_t$  with respect to  $\mathbf{w}_t$  and setting it equal to zero gives

$$\frac{d\varepsilon_t}{d\mathbf{w}_t} = -2\mathbf{p}_t + 2\mathbf{R}_t \mathbf{w}_t = 0$$

and the optimal value of  $\mathbf{w}_t$  is

$$\mathbf{w}_t = \mathbf{R}_t^\# \mathbf{p}_t$$

where  $\mathbf{R}_t^\#$  is the Moore-Penrose pseudo-inverse (equivalent to the standard inverse  $\mathbf{R}_t^{-1}$  if  $\mathbf{R}_t$  is of full rank). Substituting this in the equation for the SSE gives

$$\begin{aligned}\varepsilon_t &= \mathbf{d} \mathbf{d}^T - 2(\mathbf{p}_t^T \mathbf{R}_t^{-1}) \mathbf{p} + (\mathbf{p}_t^T \mathbf{R}_t^{-1}) \mathbf{R}_t \mathbf{R}_t^{-1} \mathbf{p}_t \\ &= \mathbf{d} \mathbf{d}^T - \mathbf{p}_t^T \mathbf{R}_t^{-1} \mathbf{p}_t\end{aligned}$$

## B.2 Updating the VCM for a change in the included expansion terms

Application of the Metropolis Hastings algorithm to the self-structuring problem requires the calculation of  $\varepsilon_{t+1}$  and  $\mathbf{R}_{t+1}^{-1}$  when a term is added to the nonlinear expansion, and  $\varepsilon_{t-1}$  and  $\mathbf{R}_{t-1}^{-1}$  when a term is removed.

### B.2.1 Adding a term

Define  $\tilde{x}^i$  to be the nonlinear function of the input vector  $\mathbf{x}^i$  to be added to the network and let

$$\begin{aligned}\tilde{\mathbf{r}} &= \sum_{i=1}^M \hat{\mathbf{x}}_t^i \tilde{x}^i \\ \tilde{q} &= \sum_{i=1}^M (\tilde{x}^i)^2 \\ \tilde{p} &= \sum_{i=1}^M \tilde{x}^i d_i\end{aligned}$$

Using the Woodbury matrix update formulae (see appendix C) gives

$$\begin{aligned}\mathbf{R}_{t+1}^{-1} &= \begin{bmatrix} \mathbf{R}_t^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \gamma^{-1} \begin{bmatrix} -\mathbf{R}_t^{-1} \tilde{\mathbf{r}} \\ 1 \end{bmatrix} \begin{bmatrix} -\tilde{\mathbf{r}}^T \mathbf{R}_t^{-1} & 1 \end{bmatrix} \\ \gamma &= \tilde{q} + \tilde{\mathbf{r}}^T \mathbf{R}_t^{-1} \tilde{\mathbf{r}}\end{aligned}\tag{B.1}$$

and thus

$$\begin{aligned}\varepsilon_{t+1} &= \mathbf{d}\mathbf{d}^T - \mathbf{p}_t^T \mathbf{R}_{t+1}^{-1} \mathbf{p}_t \\ &= \mathbf{d}\mathbf{d}^T - \begin{bmatrix} \mathbf{p}_t^T & \tilde{p} \end{bmatrix} \left( \begin{bmatrix} \mathbf{R}_t^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \gamma^{-1} \begin{bmatrix} -\mathbf{R}_t^{-1} \\ 1 \end{bmatrix} \begin{bmatrix} -\tilde{\mathbf{r}}^T \mathbf{R}_t^{-1} & 1 \end{bmatrix} \right) \begin{bmatrix} \mathbf{p}_t \\ \tilde{p} \end{bmatrix} \\ &= \mathbf{d}\mathbf{d}^T - \mathbf{p}_t^T \mathbf{R}_t^{-1} \mathbf{p}_t - \gamma^{-1} [-\mathbf{p}_t^T \mathbf{R}_t^{-1} \tilde{\mathbf{r}} + \tilde{p}] [-\tilde{\mathbf{r}}^T \mathbf{R}_t^{-1} \mathbf{p}_t^T + \tilde{p}] \\ &= \varepsilon_t - \gamma^{-1} [\tilde{p} - \mathbf{p}_t^T \mathbf{R}_t^{-1} \tilde{\mathbf{r}}]^2\end{aligned}$$

hence

$$\Delta\varepsilon = -\gamma^{-1} [\tilde{p} - \mathbf{p}_t^T \mathbf{R}_t^{-1} \tilde{\mathbf{r}}]^2\tag{B.2}$$

If the term being added to the expansion is a linear combination of terms currently included then  $\mathbf{R}_{t+1}$  will not be of full rank. In this case  $\gamma$  will be zero, and the above update formulae

will not apply. This situation can be avoided by always rejecting such a change to the network in stage 4 of the algorithm.

### B.2.2 Removing a term

A derivation is given in appendix C of the inverse of a matrix which is a sub-matrix of a matrix with known inverse. Using the same definitions as above, except letting  $\tilde{x}^i$  now denote the nonlinear term being removed, gives

$$\mathbf{R}_t = \begin{bmatrix} \mathbf{R}_{t-1} & \tilde{\mathbf{r}} \\ \tilde{\mathbf{r}}^T & \tilde{q} \end{bmatrix}$$

Define the inverse as

$$\mathbf{R}_t^{-1} = \begin{bmatrix} \mathbf{R}'_{t-1} & \tilde{\mathbf{r}}' \\ \tilde{\mathbf{r}}'^T & \tilde{q}' \end{bmatrix}$$

and then the updated inverse is

$$\mathbf{R}_{t-1}^{-1} = \mathbf{R}'_{t-1} - \frac{\tilde{\mathbf{r}}' \tilde{\mathbf{r}}'^T}{\tilde{q}'} \quad (\text{B.3})$$

Thus

$$\begin{aligned} \varepsilon_{t-1} &= \mathbf{d}\mathbf{d}^T - \mathbf{p}_{t-1}^T \mathbf{R}_{t-1}^{-1} \mathbf{p}_{t-1} \\ &= \mathbf{d}\mathbf{d}^T - \mathbf{p}_{t-1}^T \left( \mathbf{R}'_{t-1} - \frac{\tilde{\mathbf{r}}' \tilde{\mathbf{r}}'^T}{\tilde{q}'} \right) \mathbf{p}_{t-1} \\ &= \mathbf{d}\mathbf{d}^T - \mathbf{p}_{t-1}^T \mathbf{R}'_{t-1} \mathbf{p}_{t-1} + \mathbf{p}_{t-1}^T \frac{\tilde{\mathbf{r}}' \tilde{\mathbf{r}}'^T}{\tilde{q}'} \mathbf{p}_{t-1} \end{aligned}$$

and

$$\begin{aligned} \varepsilon_t &= \mathbf{d}\mathbf{d}^T - \begin{bmatrix} \mathbf{p}_{t-1} & \tilde{p} \end{bmatrix} \begin{bmatrix} \mathbf{R}'_{t-1} & \tilde{\mathbf{r}}' \\ \tilde{\mathbf{r}}'^T & \tilde{q}' \end{bmatrix} \begin{bmatrix} \mathbf{p}_{t-1} \\ \tilde{p} \end{bmatrix} \\ &= \mathbf{d}\mathbf{d}^T - \mathbf{p}_{t-1}^T \mathbf{R}'_{t-1} \mathbf{p}_{t-1} - 2\mathbf{p}_{t-1}^T \tilde{\mathbf{r}}' \tilde{p} - \tilde{q}' \tilde{p}^2 \end{aligned}$$

hence

$$\Delta\varepsilon = \frac{(\mathbf{p}_{t-1}^T \tilde{\mathbf{r}}')^2}{\tilde{q}'} + 2\mathbf{p}_{t-1}^T \tilde{\mathbf{r}}' \tilde{p} + \tilde{q}' \tilde{p}^2 \quad (\text{B.4})$$

The updated matrix  $\mathbf{R}_{t-1}$  will always be of full rank as it is a submatrix of the full rank matrix  $\mathbf{R}_t$ .

---

## MATRIX INVERSE UPDATING RESULTS

---

This appendix contains results for updating matrix inverses, to reduce the amount of computation needed.

### C.1 Increasing the matrix size

Given  $\mathbf{R}$  partitioned as

$$\mathbf{R} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}$$

then Woodbury's matrix inverse formula [89] states that  $\mathbf{R}^{-1}$  may be written as

$$\mathbf{R}^{-1} = \begin{bmatrix} \mathbf{A}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{F} \\ \mathbf{I} \end{bmatrix} \mathbf{H}^{-1} \begin{bmatrix} \mathbf{G} & \mathbf{I} \end{bmatrix}$$

where

$$\begin{aligned} \mathbf{F} &= -\mathbf{A}^{-1}\mathbf{B} \\ \mathbf{G} &= -\mathbf{C}\mathbf{A}^{-1} \\ \mathbf{H} &= \mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B} \end{aligned}$$

Defining  $\mathbf{A} = \mathbf{R}_t$ ,  $\mathbf{B} = \tilde{\mathbf{r}}$ ,  $\mathbf{C} = \tilde{\mathbf{r}}^T$ ,  $\mathbf{D} = \tilde{q}$  and  $\mathbf{R} = \mathbf{R}_{t+1}$  gives equation B.1 in Appendix B.

### C.2 Decreasing the matrix size

Some manipulation of the inverse by partitioning formulae in [76] implies that if

$$\mathbf{R}_t^{-1} = \begin{bmatrix} \mathbf{P} & \mathbf{Q} \\ \mathbf{R} & \mathbf{S} \end{bmatrix}$$

and

$$\mathbf{R}_t = \begin{bmatrix} \tilde{\mathbf{P}} & \tilde{\mathbf{Q}} \\ \tilde{\mathbf{R}} & \tilde{\mathbf{S}} \end{bmatrix}$$

then

$$\tilde{\mathbf{P}}^{-1} = \mathbf{P} - \mathbf{Q}\mathbf{S}^{-1}\mathbf{R}$$

Defining  $\tilde{\mathbf{P}} = \mathbf{R}_{t-1}$ ,  $\tilde{\mathbf{Q}} = \tilde{\mathbf{r}}$ ,  $\tilde{\mathbf{R}} = \tilde{\mathbf{r}}^T$ ,  $\tilde{\mathbf{S}} = \tilde{q}$ ,  $\mathbf{P} = \mathbf{R}'_{t-1}$ ,  $\mathbf{Q} = \tilde{\mathbf{r}}'$ ,  $\mathbf{R} = \tilde{\mathbf{r}}'^T$  and  $\mathbf{S} = \tilde{q}'$  gives the update formula in equation B.3.

---

## MARKOV CHAIN STATE OCCUPANCY STATISTICS

---

Consider a Markov chain [48] defined by the transition matrix  $\mathbf{T} = \{p_{ij}\}$ . The multi-step transition probabilities  $\phi_{ij}(n) = p(s(n) = j | s(0) = i)$ , where  $s(k)$  is the state occupied after  $k$  transitions, are given by  $\Phi(n) = \{\phi_{ij}(n)\} = \mathbf{T}^n$ . Taking z-transforms,  $\Phi^g(z) = \{\phi_{ij}(z)\} = [\mathbf{I} - \mathbf{T}z]^{-1}$ . A partial fraction expansion will result in a closed-form expression for  $\mathbf{P}^n$  when this transform is inverted.

Let  $v_{ij}(n)$  be the number of times state  $j$  is entered in  $n$  transitions, given that the state at time zero was  $i$ , and  $\overline{(\cdot)}$  denote mean values or expectations.

### D.1 Mean state occupancy

Define

$$x_{ij}(n) = \begin{cases} 1 & \text{if } s(n) = j \\ 0 & \text{otherwise} \end{cases} \quad \text{given } s(0) = i$$

then

$$v_{ij}(n) = \sum_{m=0}^n x_{ij}(m)$$

$$\overline{v}_{ij}(n) = \sum_{m=0}^n \overline{x}_{ij}(m) = \sum_{m=0}^n \phi_{ij}(m)$$

Taking z-transforms

$$\overline{v}_{ij}^g(z) = \frac{1}{1-z} \phi_{ij}(z)$$

and a partial fraction expansion will result in a closed-form expression for  $\overline{v}_{ij}(n)$ . Define  $\overline{\mathbf{N}}(n) = \{\overline{v}_{ij}(n)\}$ .

### D.2 Second moments of state occupancy

$$v_{ij}^2 = \sum_{m=0}^n x_{ij}(m) \sum_{r=0}^n x_{ij}(r) = \sum_{m=0}^n \sum_{r=0}^n x_{ij}(m) x_{ij}(r)$$

$$\overline{v_{ij}^2}(n) = \sum_{m=0}^n \sum_{r=0}^n \overline{x_{ij}(m)x_{ij}(r)} = \sum_{m=0}^n \sum_{r=0}^n \psi_{ijj}(m, r)$$

where  $\psi_{ijk}(m, r)$  is the joint transition probability,  $p(s(m) = j, s(r) = k | s(0) = i)$ , hence

$$\begin{aligned} \overline{v_{ij}^2}(n) &= 2 \sum_{m=0}^n \sum_{r=m}^n \phi_{ij}(m)\phi_{jj}(r-m) - \sum_{m=0}^n \phi_{ij}(m) \\ &= 2 \sum_{m=0}^n \phi_{ij}(m)\overline{v_{jj}}(n-m) - \overline{v_{ij}}(n) \end{aligned}$$

Taking z-transforms results in

$$\overline{v_{ij}^2}^g(z) = \frac{1}{1-z} \phi_{ij}^g(z) [2\phi_{jj}^g(z) - 1]$$

Again a closed-form expression for  $\overline{v_{ij}^2}(n)$  can be found by using partial fractions and inverting this transform. Define  $\overline{\mathbf{N}^2}(n) = \{\overline{v_{ij}^2}(n)\}$ .

### D.3 Occupancies under random starting

If the chain is started by choosing a state with probability

$$\boldsymbol{\pi}(0) = [\pi_0(0) \cdots \pi_k(0)]$$

then the vector of means is

$$\overline{\mathbf{v}} = \boldsymbol{\pi}(0)\overline{\mathbf{N}}(n)$$

the second moments are given by

$$\overline{\mathbf{v}^2}(n) = \boldsymbol{\pi}(0)\overline{\mathbf{N}^2}(n)$$

and the variances by

$$\mathbf{v}^v(n) = \overline{\mathbf{v}^2}(n) - \overline{\mathbf{v}}(n)\square\overline{\mathbf{v}}(n)$$

where  $\square$  denotes element-wise multiplication.

### D.4 General 2-state process

The results required in the main text are the mean and variance of the occupancy of state one of a two-state process, started with  $\boldsymbol{\pi}(0) = [0.5 \ 0.5]$ . The general two-state process is

defined by

$$\mathbf{T} = \begin{bmatrix} 1-a & a \\ b & 1-b \end{bmatrix}$$

and some algebraic manipulation results in

$$\overline{v_1}(n) \simeq (n+1) \frac{a}{a+b} + \frac{b-a}{2(a+b)^2} \quad (\text{D.1})$$

$$\begin{aligned} \overline{v_1^2}(n) &\simeq n^2 \frac{a^2}{(a+b)^2} + n \frac{a}{a+b} \left( \frac{3a}{a+b} + \frac{3b-a}{(a+b)^2} - 1 \right) \\ &+ \frac{2a^2}{(a+b)^2} + \frac{a}{a+b} \left( \frac{3b-a}{(a+b)^2} - 1 \right) - \frac{a(3b-a)(1-a-b)}{(a+b)^4} \\ &+ \frac{b-a}{2(a+b)^2} \left( \frac{2b}{(a+b)^2} - 1 \right) \end{aligned}$$

$$\begin{aligned} v_1^v(n) &\simeq \frac{na}{a+b} \left( \frac{2b}{(a+b)^2} + \frac{a}{a+b} - 1 \right) \\ &+ \frac{1}{4(a+b)^4} (-4a^3 + 3a^2 + 8a^2b - 14ab + 12ab^2 + 3b^2) \\ &+ \frac{2ab}{(a+b)^3} + \frac{a(2a-1)+b}{2(a+b)^2} - \frac{a}{a+b} \end{aligned} \quad (\text{D.2})$$

Where the approximations are due to neglecting terms in  $n(1-a-b)^n$  and  $(1-a-b)^n$ , making the above expressions valid only for large  $n$ .

---

## DERIVATION OF THE ACCEPTANCE PROBABILITIES FOR THE REVERSIBLE JUMP MCMC SAMPLER

---

This appendix contains details from [40] of the derivation of the acceptance probabilities for the Reversible Jump sampler.

From the proposal distributions discussed in chapter 5, the transition kernel is

$$T(\mathbf{x}, B) = \sum_m \int_B q_m(\mathbf{x}, d\mathbf{x}') A_m(\mathbf{x}, \mathbf{x}') + s(\mathbf{x}) I[\mathbf{x} \in B]$$

The probability of moving from state  $\mathbf{x}$  to a region of parameter space  $B$  is the probability of moving from  $\mathbf{x}$  to  $B$  via each of the possible move types;  $s(\mathbf{x})$  is the probability of not moving from  $\mathbf{x}$ , either by not proposing a move, or by rejecting the proposed move.  $I[\cdot]$  is the indicator function, which takes a value of one when the condition within the bracket is true. The final term is thus the probability of  $\mathbf{x}$  being within  $B$ , and no move being made. Detailed balance requires that the probability of a transition from a region of parameter space  $A$  to a region  $B$  is equal to the probability of a transition from  $B$  to  $A$ , giving

$$\begin{aligned} & \sum_m \int_A \pi(d\mathbf{x}) \int_B q_m(\mathbf{x}, d\mathbf{x}') A_m(\mathbf{x}, \mathbf{x}') + \int_{A \cap B} \pi(d\mathbf{x}) s(\mathbf{x}) \\ &= \sum_m \int_B \pi(d\mathbf{x}') \int_A q_m(\mathbf{x}', d\mathbf{x}) A_m(\mathbf{x}', \mathbf{x}) + \int_{B \cap A} \pi(d\mathbf{x}') s(\mathbf{x}') \end{aligned}$$

and clearly a sufficient condition is that

$$\int_A \pi(d\mathbf{x}) \int_B q_m(\mathbf{x}, d\mathbf{x}') A_m(\mathbf{x}, \mathbf{x}') = \int_B \pi(d\mathbf{x}') \int_A q_m(\mathbf{x}', d\mathbf{x}) A_m(\mathbf{x}', \mathbf{x})$$

for each  $m$ ,  $A$  and  $B$ . The problem is to determine  $A_m(\mathbf{x}, \mathbf{x}')$  to achieve this.

The following technical condition is sufficient to allow  $A(\mathbf{x}, \mathbf{x}')$  to be specified. If

$\pi(d\mathbf{x}) q_m(\mathbf{x}, d\mathbf{x}')$  has finite density  $f_m(\mathbf{x}, \mathbf{x}')$  with respect to a symmetric measure  $\zeta_m$ , and  $f_m(\mathbf{x}, \mathbf{x}')$  and  $f_m(\mathbf{x}', \mathbf{x})$  are either both positive or both zero for all values of  $\mathbf{x}, \mathbf{x}'$

then

$$\begin{aligned}
\int_A \pi(d\mathbf{x}) \int_B q_m(\mathbf{x}, d\mathbf{x}') A_m(\mathbf{x}, \mathbf{x}') &= \int_A \int_B \zeta_m(d\mathbf{x}, d\mathbf{x}') f_m(\mathbf{x}, \mathbf{x}') A_m(\mathbf{x}, \mathbf{x}') \\
&= \int_B \int_A \zeta_m(d\mathbf{x}', d\mathbf{x}) f_m(\mathbf{x}', \mathbf{x}) A_m(\mathbf{x}', \mathbf{x}) \\
&= \int_B \pi(d\mathbf{x}') \int_A q_m(\mathbf{x}', d\mathbf{x}) A_m(\mathbf{x}', \mathbf{x})
\end{aligned}$$

where the middle equality holds by the assumed symmetry of  $\zeta_m$ , provided that

$$A_m(\mathbf{x}, \mathbf{x}') f_m(\mathbf{x}, \mathbf{x}') = A_m(\mathbf{x}', \mathbf{x}) f_m(\mathbf{x}, \mathbf{x}')$$

The following acceptance probability clearly satisfies this condition.

$$\begin{aligned}
A_m(\mathbf{x}, \mathbf{x}') &= \min \left( 1, \frac{f_m(\mathbf{x}', \mathbf{x})}{f_m(\mathbf{x}, \mathbf{x}')} \right) \\
&= \min \left( 1, \frac{\pi(d\mathbf{x}') q_m(\mathbf{x}', d\mathbf{x})}{\pi(d\mathbf{x}) q_m(\mathbf{x}, d\mathbf{x}')} \right)
\end{aligned}$$