

May 1993

LIDS-TH-2178

**Research Supported By:**

Air Force Office of Scientific Research  
Grant AFOSR-92-J-0002

Draper Laboratory IR&D Program  
DL-H-441678

Office of Naval Research  
Grant ONR N00014-91-J-1004

Army Research Office  
Grant ARO DAAL03-92-G-0115

Lincoln Laboratory Fellowship

**Image Processing with Multiscale  
Stochastic Models**

**Mark R. Luetgen**

May 1993

LIDS-TH-2178

Sponsor Acknowledgments

Air Force Office of Scientific Research  
Grant AFOSR-92-J-0002

Draper Laboratory IR&D Program  
DL-H-441678

Office of Naval Research  
Grant ONR N00014-91-J-1004

Army Research Office  
Grant ARO DAAL03-92-G-0115

Lincoln Laboratory Fellowship

## Image Processing with Multiscale Stochastic Models

Mark R. Luetgen

This report is based on the unaltered thesis of Mark R. Luetgen submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of Doctor of Philosophy at the Massachusetts Institute of Technology in May 1993.

This research was conducted at the M.I.T. Laboratory for Information and Decision Systems with research support gratefully acknowledged by the above mentioned sponsors.

Laboratory for Information and Decision Systems  
Massachusetts Institute of Technology  
Cambridge, MA 02139, USA

# Image Processing with Multiscale Stochastic Models

by

Mark R. Luetttgen

S.B.. Massachusetts Institute of Technology (1988)

S.M.. Massachusetts Institute of Technology (1990)

Submitted to the Department of Electrical Engineering and Computer Science  
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1993

© Mark R. Luetttgen, MCMXCIII. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute copies  
of this thesis document in whole or in part, and to grant others the right to do so.

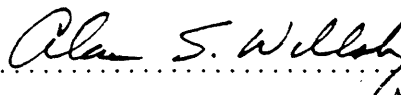
Author .....



Department of Electrical Engineering and Computer Science

May 12, 1993

Certified by .....



Alan S. Willsky

Professor of Electrical Engineering

Thesis Supervisor

Accepted by .....

Campbell L. Searle

Chairman, Departmental Committee on Graduate Students

# Image Processing with Multiscale Stochastic Models

by

Mark R. Luetngen

Submitted to the Department of Electrical Engineering and Computer Science  
on May 12, 1993, in partial fulfillment of the  
requirements for the degree of  
Doctor of Philosophy

## Abstract

In this thesis, we develop image processing algorithms and applications for a particular class of multiscale stochastic models. First, we provide background on the model class, including a discussion of its relationship to wavelet transforms and the details of a two-sweep algorithm for estimation. A multiscale model for the error process associated with this algorithm is derived. Next, we illustrate how the multiscale models can be used in the context of regularizing ill-posed inverse problems and demonstrate the substantial computational savings that such an approach offers. Several novel features of the approach are developed including a technique for choosing the optimal resolution at which to recover the object of interest. Next, we show that this class of models contains other widely used classes of statistical models including 1-D Markov processes and 2-D Markov random fields, and we propose a class of multiscale models for approximately representing Gaussian Markov random fields. These results, coupled with those illustrating the computational efficiencies that the multiscale models lead to, suggest that the multiscale framework is a powerful paradigm for image processing both because of the efficient algorithms it admits and because of the rich class of phenomena it can be used to describe. This motivates us in the final section of this thesis to pursue further algorithmic development for the multiscale models. In particular, we develop an efficient likelihood calculation algorithm for multiscale models and demonstrate an application of the algorithm in the area of texture discrimination. The thesis concludes with a review of our main results and with a discussion of a few of the many open problems and promising directions for further research and application.

Thesis Supervisor: Alan S. Willsky  
Title: Professor of Electrical Engineering



## Acknowledgments

I want to thank my thesis advisor, Alan Willsky, for the input and guidance he gave to me throughout my doctoral program at MIT. His perspective and insight has had a profound influence both on this thesis and on my professional development. I also wish to thank the members of my thesis committee, Clem Karl, Bob Tenney, John Tsitsiklis and Greg Wornell. They all provided important criticism and invaluable comments. Clem Karl, in particular, provided substantial input throughout the period during which this research was being done. Thanks are also in order for Albert Benveniste, Claude Labit, Fabrice Heitz and Patrick Bouthemy for their hospitality and help during my visit to France in the summer of 1991.

I want to thank my office-mates and friends at MIT, past and present, who in many ways made getting through this place much easier, especially Peyman Milanfar, Eric Miller, Mike Daniel, Rachel Learned, Bill Irving and Mitch Livstone.

Above all, the love and support of my family and Wanda made this effort possible.

I dedicate this thesis to my cat, Heddy, who chewed and slept on many earlier versions.

---

# Contents

<b>1</b>	<b>Introduction</b>	<b>7</b>
1.1	Multiscale Signal Processing and Wavelet Transforms . . . . .	7
1.2	Multiscale Stochastic Models . . . . .	10
1.3	Multiscale Optimal Estimation . . . . .	18
1.4	Other Related Work . . . . .	21
1.5	Thesis Organization . . . . .	22
<b>2</b>	<b>Applications of Multiscale Regularization to the Computation of Optical Flow</b>	<b>26</b>
2.1	Introduction . . . . .	26
2.2	Multiscale Regularization . . . . .	34
2.3	Experimental Results . . . . .	42
2.3.1	Rotation Sequence . . . . .	43
2.3.2	Yosemite Sequence . . . . .	52
2.3.3	Moving Vehicle Sequence . . . . .	58
2.3.4	Chopper Sequence . . . . .	65
2.4	Summary . . . . .	71
<b>3</b>	<b>Multiscale Representations of Markov Random Fields</b>	<b>72</b>
3.1	Introduction . . . . .	72
3.2	Generalized Multiscale Stochastic Models . . . . .	76
3.3	Representation of 1-D Reciprocal Processes . . . . .	78
3.3.1	1-D Reciprocal Processes . . . . .	78

3.3.2	Exact Representations of 1-D Reciprocal Processes . . . . .	79
3.3.3	Examples . . . . .	89
3.4	Representation of 2-D Markov Random Fields . . . . .	110
3.4.1	2-D Markov Random Fields . . . . .	110
3.4.2	Exact Representations of 2-D Markov Random Fields . . . . .	112
3.4.3	Approximate Representation of 2-D Gaussian MRF's . . . . .	118
3.5	Examples of 2-D GMRF Approximate Representations . . . . .	130
3.5.1	Separable Gaussian MRF Examples . . . . .	130
3.5.2	Non-Separable Gaussian MRF Examples . . . . .	133
3.6	Summary . . . . .	139
<b>4</b>	<b>Likelihood Calculations and their Applications</b>	<b>141</b>
4.1	Introduction . . . . .	141
4.2	Likelihood Function Calculation . . . . .	143
4.2.1	From Partial to Total Ordering . . . . .	145
4.2.2	Algorithm Description . . . . .	148
4.3	A Texture Discrimination Application . . . . .	155
4.3.1	The Texture Discrimination Problem . . . . .	157
4.3.2	Performance on Rectangular Domains . . . . .	159
4.3.3	Performance on Non-rectangular Regions . . . . .	166
4.4	Summary . . . . .	171
<b>5</b>	<b>Thesis Contributions and Recommendations for Future Research</b>	<b>172</b>
5.1	Thesis Contributions . . . . .	172
5.2	Recommendations for Future Research . . . . .	174
<b>A</b>	<b>Smoothing Error Models</b>	<b>182</b>
<b>B</b>	<b>Appendices to Chapter 2</b>	<b>185</b>
B.1	Non-homogeneous Tree Structures . . . . .	185
B.2	Iterative Approaches to Computation of the Smoothness Constraint Solution . . . . .	186

B.3 MR Algorithm Computational Complexity Analysis . . . . .	189
<b>C Correlation Computations for Multiscale GMRF Models</b>	<b>192</b>
<b>D Appendices to Chapter 4</b>	<b>195</b>
D.1 Likelihood Calculations for Singular $P$ , and $P(s Y)$ . . . . .	195
D.2 A Minimum Distance Classifier . . . . .	200

# Chapter 1

## Introduction

### 1.1 Multiscale Signal Processing and Wavelet Transforms

There has been, and continues to be, strong interest in the signal processing community in multiresolution methods in image processing. One often expressed reason for this is that multiresolution methods lead naturally to processing which is *scale-invariant*, that is, the interpretation or processing of the image does not depend on the scale of the objects therein [85, 38]. For instance, several researchers have developed stereo matching techniques which operate at a range of scales, allowing objects to be matched despite changes in size [94, 115]. Others have developed multiresolution methods for edge detection [147, 114, 93], filtering [32, 110] and compression [21, 44]. Such multiresolution approaches often also lead to substantial computational advantages, especially when the processing is organized in a coarse-to-fine progression. For instance, in [18] a multiresolution segmentation algorithm is proposed in which images are first segmented at coarse resolutions, the results of which are then used to guide segmentation at successively finer levels. The operation at any given level is iterative, and because there is a good initial guess from the previous resolution level, the number of iterations can be substantially reduced over an approach which begins and ends at the finest level of resolution.

The development recently of multiscale representations based on wavelet transforms has initiated a substantial amount of new research in multiscale signal processing [39, 40, 85, 56]. This theory provides very elegant and general tools for decomposing signals into their components at different scales, and a rich framework for the development of applications in a variety of areas [37, 44, 87, 153, 152].

The orthonormal wavelet representation of a signal is based on translations and dilations of a single *scaling* function  $\phi(x)$ . The approximation of a function  $f(x)$ ,  $x \in \mathcal{R}$  at the  $m^{\text{th}}$  level of resolution is given by:

$$f_m(x) = \sum_n f(m, n) \phi_{m,n}(x) \quad (1.1)$$

where  $\phi_{m,n}(x) \equiv 2^{m/2} \phi(2^m x - n)$  and the set of translates of the scaling function  $\{\phi_{m,n}(x), n \in \mathcal{Z}\}$  provide an orthonormal basis for a space of functions  $V_m$  at resolution  $m$  [86]. In particular, the scaling function  $\phi(x)$  provides a multiresolution analysis of the space of square integrable functions  $L_2(\mathcal{R})$ , which consists of a sequence of nested approximation spaces:

$$\cdots V_{m-1} \subset V_m \subset V_{m+1} \cdots \quad (1.2)$$

with the properties:

1.  $g(x) \in V_m \leftrightarrow g(2x) \in V_{m+1}$
2.  $\bigcap V_m = \{0\}, \overline{\bigcup V_m} = L_2(\mathcal{R})$

and  $V_m = \overline{\text{linear span}\{\phi_{m,n}(x), n \in \mathcal{Z}\}}$ . The multiresolution approximation in (1.1) is then just the partial expansion of  $f(x)$  in  $V_m$ , with respect to the basis  $\{\phi_{m,n}(x), n \in \mathcal{Z}\}$ . The coefficients of the expansion,  $f(m, n)$ , are given by a projection equation:

$$f(m, n) = \int \phi_{m,n}(x) f(x) dx \quad (1.3)$$

Associated with the scaling function  $\phi(x)$  is a *wavelet* function  $\psi(x)$  which provides an orthonormal expansion of certain detail spaces associated with the multiresolution

analysis. In particular, define the detail space  $W_m$  by:

$$V_m = W_m \oplus V_{m-1} \quad (1.4)$$

That is,  $W_m$  is the space of functions contained in the resolution space  $V_m$ , but not contained in  $V_{m-1}$ . Thus,  $W_m \perp V_{m-1}$ , i.e. the space of incremental detail associated with an increase in one resolution step, from  $V_{m-1}$  to  $V_m$ , is orthogonal to  $V_{m-1}$ . The wavelet  $\psi(x)$ , and in particular, the set of translates for a given  $m$ ,  $\{\psi_{m,n}(x), n \in \mathcal{Z}\}$ , where  $\psi_{m,n}(x) \equiv 2^{m/2}\psi(2^m x - n)$ , provide an orthonormal basis of  $W_m$ . From (1.4), we see that  $V_{m-1} = W_{m-1} \oplus V_{m-2}$ , and by substituting this relation into (1.4), and proceeding in a similar way with respect to  $V_{m-2}, V_{m-3}, \dots$ , we see that the space of functions at any given resolution can be written as a direct sum of the detail spaces:  $V_m = \oplus_{n \leq m} W_n$ . Thus, we can rewrite the expansion (1.1) as:

$$f_m(x) = \sum_{k=-\infty}^m \sum_n d(k, n) \psi_{k,n}(x) \quad (1.5)$$

where the coefficients  $d(k, n)$  of this expansion are given by a projection equation similar to (1.3).

One simple example of a multiresolution analysis is associated with the Haar wavelet transform. In this case, the scaling function  $\phi(x)$  and corresponding wavelet  $\psi(x)$  are given by:

$$\phi(x) = \begin{cases} 1 & \text{if } x \in [0, 1] \\ 0 & \text{else} \end{cases} \quad (1.6)$$

$$\psi(x) = \begin{cases} 1 & \text{if } x \in [0, 1/2] \\ -1 & \text{if } x \in (1/2, 1] \\ 0 & \text{else} \end{cases} \quad (1.7)$$

The  $f(m, n)$  in this case just correspond to averages of  $f(x)$  over intervals:

$$f(m, n) = 2^{m/2} \int_{2^{-m}n}^{2^{-m}(n+1)} f(x) dx \quad (1.8)$$

As  $m$  increases, the length of the intervals decreases, and we see that the  $f(m, n)$  provide information about  $f(x)$  at progressively finer levels of resolution.

Much of the usefulness of the wavelet transform in practical signal processing problems stems from the fact that the expansion coefficients  $f(m, n)$  and  $d(m, n)$  can be efficiently computed recursively in scale. Specifically, associated with each  $(\phi, \psi)$  pair is a *quadrature mirror filter* pair  $(h, g)$  that allows computation of  $f(m, n)$  and  $d(m, n)$  from the scaling function coefficients  $f(m + 1, n)$  at the next finest scale. In particular:

$$f(m, n) = \sum_k h(2n - k)f(m + 1, k) \quad (1.9)$$

$$d(m, n) = \sum_k g(2n - k)f(m + 1, k) \quad (1.10)$$

This fact was an important link leading to the collaboration between researchers in the physics and mathematical communities, where much of the original wavelet theory was developed, and the signal processing community, where quadrature mirror filters and multirate filterbank structures had been studied for some time in communication and coding contexts [125, 138, 140, 141].

## 1.2 Multiscale Stochastic Models

As pointed out in [7], the development of statistically optimal multiscale signal processing algorithms requires a theory of multiscale stochastic processes. The wavelet-based multiresolution framework discussed above motivates the study of stochastic processes indexed by nodes of a dyadic tree in [8, 30, 27, 29] (see Figure 1-1). In this thesis, our interest is in image processing applications, and hence we will focus primarily on processes defined on the quadtree structure shown in Figure 1-2.

Pyramidal data structures such as the quadtree arise naturally in multiresolution approaches to image processing problems. For instance, successive filtering and decimation operations lead to images defined on such a hierarchy of grids in the Laplacian pyramid coding algorithm of Burt and Adelson [21] and of course in the wavelet trans-



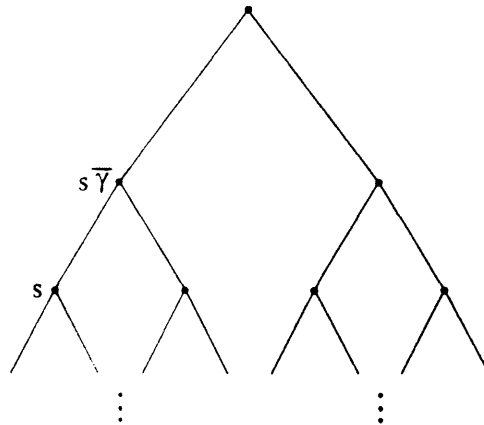


Figure 1-1: The dyadic tree. The abstract index  $s$  refers to a node in the tree;  $s\bar{\gamma}$  refers to the parent node.

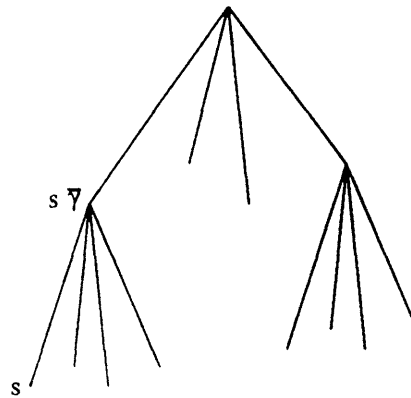


Figure 1-2: Quadtree structures such as that above arise naturally in multiresolution approaches to image processing problems.

form decomposition of images [86]. Also, the multigrid approaches to low level vision problems discussed by Terzopoulos [134] involve relaxation on a similar sequence of grids. It is important to emphasize here, however, that in contrast to approaches such as these, in our case we will use the quadtree structure to *model* a spatially-distributed random field rather than to analyze or decompose a given field. As we will see, this model does, in fact, lead to processing algorithms operating on the quadtree, but these algorithms are optimal estimation and likelihood calculation procedures (if the

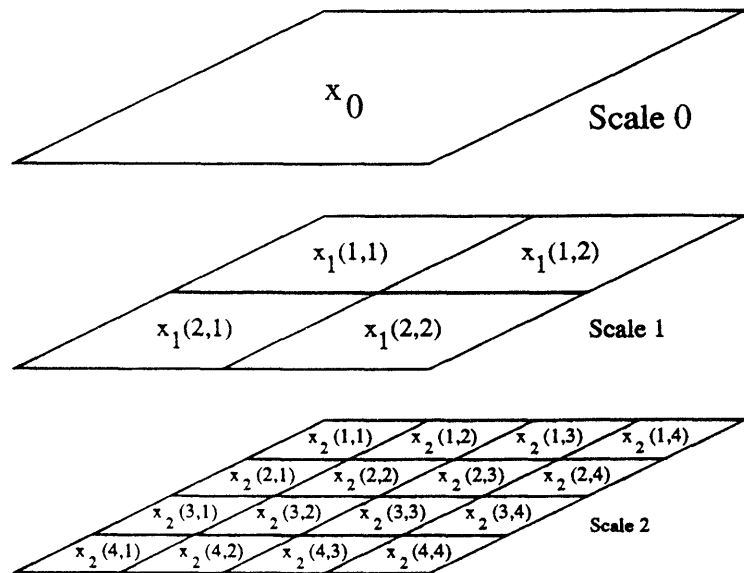


Figure 1-3: The structure of a multiscale random field is depicted. The components of the field are denoted  $x_m(i, j)$  where  $m$  refers to the scale and the pair  $(i, j)$  denotes a particular grid location at a given scale. At the coarsest scale, the field is represented by a single value or state vector and, more generally, at the  $m^{\text{th}}$  scale there are  $4^m$  state vectors.

assumptions of the model are satisfied), and thus are completely different in form, nature, and intent from standard pyramidal decomposition procedures.

A stochastic process defined on the quadtree is naturally indexed by the 3-tuple  $(m, i, j)$ , where  $m$  is a scale index and the pair  $(i, j)$  specifies a spatial location. Such a scheme is shown in Figure 1-3 where the value of the multiscale process at a particular node in the quadtree is given by  $x_m(i, j)$ . Higher levels of the tree correspond to coarser scale representations of the random field at the finest level. In particular, at the  $m^{\text{th}}$  level, the process is characterized by  $4^m$  values (or, more generally, state vectors). The coarsest version of the random field, represented by the single value (or vector)  $x_0$  at the root node of the quadtree, corresponds to an aggregate description of the process at the finest level of the quadtree.

The model introduced in [30, 27, 29] and adapted here to quadtrees for the state  $x_m(i, j)$  is motivated directly by the wavelet transform synthesis equation:

$$f(m+1, n) = \sum_k h(2k-n)f(m, k) + \sum_k g(2k-n)d(m, k) \quad (1.11)$$

This equation is just the counterpart to the analysis equations, (1.9), (1.10), and describes how the scaling coefficients evolve in scale, from coarse to fine. In particular, note that (1.11) defines a dynamical relationship between the coefficients  $f(m, n)$  at one scale and those at the next. To make this statement more precise, let us consider again the Haar transform. In this case, the QMF filters  $h$  and  $g$  are given by:

$$h(n) = (1/\sqrt{2})\delta(n) + (1/\sqrt{2})\delta(n - 1) \quad (1.12)$$

$$g(n) = (1/\sqrt{2})\delta(n) - (1/\sqrt{2})\delta(n - 1) \quad (1.13)$$

Using (1.11), we see that, in this case, for  $n$  a multiple of 2:

$$f(m + 1, n) = (1/\sqrt{2})f(m, n/2) + (1/\sqrt{2})d(m, n/2) \quad (1.14)$$

$$f(m + 1, n + 1) = (1/\sqrt{2})f(m, n/2) - (1/\sqrt{2})d(m, n/2) \quad (1.15)$$

and the scaling coefficients in the expansion are naturally associated with the dyadic tree in Figure 1-1. In particular, a scaling coefficient  $f(m, n/2)$  at any given level is associated thru (1.14), (1.15) with two coefficients at the next level, this association being represented in Figure 1-1 by the branches connecting each node to two offspring. What this fact suggests is a statistical model which relates the components of a process at different scales in terms of the relationship of components at successive resolutions, and in fact in terms of relationships between scaling coefficients at nodes which are connected on the tree. This philosophy is similar to that in standard time series analysis where, e.g. Markov models are defined in terms of transition probabilities for the state at time  $t$  given the state at time  $t - 1$  [16], or Markov random field modeling, where the random field statistical structure is determined by local energy functions [13, 53].

Also, note that the wavelet coefficients in (1.14), (1.15) can be interpreted as the driving term in the dynamical equation describing the scale to scale evolution of the scaling coefficients of the process. Analysis in several papers suggests that for many classes of stochastic processes, these wavelet coefficients are nearly white

[46, 50, 51, 55, 109, 118, 135]. Thus, we are led naturally to models for processes defined on tree structures in which the process is built up recursively in scale, as in (1.14), (1.15), and in which the details added at successive levels, corresponding to the wavelet coefficients in (1.14), (1.15), are *independent* of previous scales.

In fact, we can define a class of processes which, while still motivated by (1.14), (1.15), allows for a substantially more general modeling framework. In particular, as shown in Chapter 3, these models allow us to represent not only processes which have an intrinsic multiscale nature, e.g.  $1/f$  and other fractal processes, but also the *entire* class of Markov random fields. To describe these models, let us define in place of the 3-tuple  $(m, i, j)$  an abstract index  $s$  to specify nodes on the quadtree. Also, let  $s\bar{\gamma}$  denote the parent of node  $s$ , i.e.  $\bar{\gamma}$  is an upward shift operator on the set of nodes on the tree, and let  $m(s)$  denote the scale of node  $s$  (we number the scales sequentially from coarse to fine, as in Figure 1-3). The stochastic tree process  $x(s) \in \mathcal{R}^n$  is then described via the following scale-recursive model:

$$x(s) = A(s)x(s\bar{\gamma}) + B(s)w(s) \quad (1.16)$$

under the following assumptions:

$$x_0 \sim \mathcal{N}(0, P_0) \quad (1.17)$$

$$w(s) \sim \mathcal{N}(0, I) \quad (1.18)$$

where  $w(s) \in \mathcal{R}^m$  and  $A(s)$  and  $B(s)$  are matrices of appropriate size<sup>1</sup>. The state variable  $x_0$  at the root node of the tree provides an initial condition for the recursion. The driving noise  $w(s)$  is white, i.e.  $w(s)$  and  $w(\sigma)$  are uncorrelated if  $s \neq \sigma$ , and is uncorrelated with the initial condition.

The class of models introduced in [30, 27, 29] for processes defined on dyadic trees is of precisely the same form as (1.16). The only difference is that the shift operator  $\bar{\gamma}$  is defined in an appropriate way — each node on the dyadic tree has two offspring,

---

<sup>1</sup>The expression  $y \sim \mathcal{N}(\mu, \lambda)$  means that random variable  $y$  is normally distributed, with mean  $\mu$  and variance  $\lambda$ .

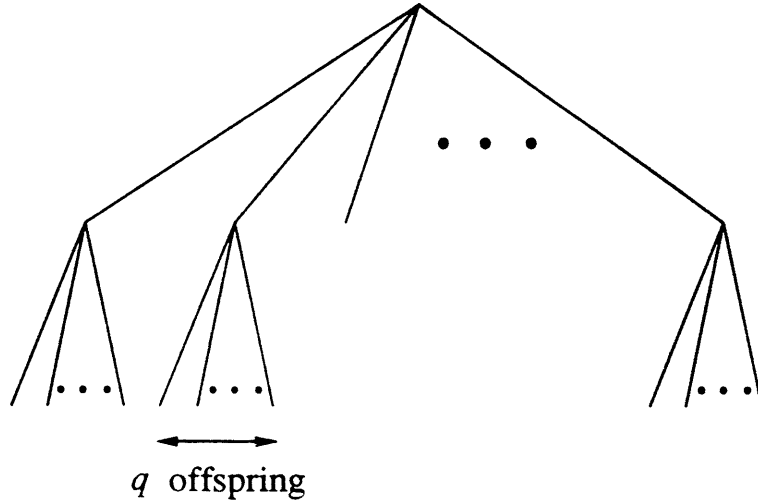


Figure 1-4: The  $q^{\text{th}}$ -order tree is illustrated. This is a set of nodes, each of which has  $q$  offspring.

as opposed to four on the quadtree. Indeed, there is an obvious generalization of the model (1.16) to general  $q^{\text{th}}$ -order trees, that is, to pyramidal structures of nodes connected such that any given node has  $q$  offspring, as shown in Figure 1-4. In the rest of this thesis, we refer often to (1.16) when we describe particular multiscale models, and the particular tree structure, e.g. quadtree or dyadic tree or  $q^{\text{th}}$ -order tree, will be clear from the context.

Interpreting the states at a given level of the tree as a representation of one scale of the process, we see that (1.16) describes the evolution of the process from coarse to fine scales. The term  $A(s)x(s\bar{\gamma})$  represents interpolation or prediction down to the next level, and  $B(s)w(s)$  represents new information added as the process evolves from one scale to the next. On the dyadic tree, we can recover a model directly associated with the Haar synthesis equations (1.14), (1.15) by setting  $A(s) = 1, B(s) = 1$  and assuming that if two nodes  $s$  and  $\sigma$  have the same parent, i.e. if  $s\bar{\gamma} = \sigma\bar{\gamma}$ , that  $w(s) = -w(\sigma)$ . In this form of the model, the driving noise is not white, but it can easily be converted to a white noise driven model by state augmentation (see [27]). In general, the choice of the parameters  $A(s)$  and  $B(s)$  and their dependence (if any) on the node  $s$ , depends upon the particular application and process being modeled.

Moreover, there is no reason to require that the model state  $x(s)$  be interpreted

as a scaling coefficient, i.e. as a local average. In this thesis, we interpret the states at a given scale more generally as *information* about the process at that scale. This information may correspond to local averages, as it has to this point, but it may also correspond to local detail, that is, to wavelet coefficients of some sequence or function. Alternatively, the values of the process at a given scale may correspond to a decimated version of the process at the finest scale. In Chapter 2 of this thesis we will use the scaling coefficient interpretation of the state and make choices of the model parameters  $A(s), B(s)$  that lead to fractal-like models. In Chapter 3 we construct parameters  $A(s), B(s)$  such that (1.16) provides a multiscale representation of 1-D Markov processes or 2-D Markov random fields, and in this case the coarse scale values are interpreted as decimated versions of a fine scale process. In this thesis, we will not use models in which the states are interpreted as wavelet coefficients, but preliminary investigations of such an approach can be found in [46].

We have motivated the model (1.16) by emphasizing its potential for modeling processes at multiple scales. However, the model also has interesting connections to standard time-series models. In particular, the set of integers with  $t$  connected to  $t-1$  as shown in Figure 1-5 can be viewed as a *first-order tree*, that is, as the simplest counterpart to the dyadic and quadrees. Gauss-Markov models of the following form on the first-order tree have found use in many applications:

$$z(t+1) = Fz(t) + G\mu(t) \tag{1.19}$$

where  $z(t)$  is the state of the process,  $F$  is a one-step transition matrix, and  $\mu(t)$  is a white Gaussian driving noise term ( $\mu(t)$  and  $\mu(\tau)$  are uncorrelated if  $t \neq \tau$ ).

The model (1.19) is widely used as the basis for the design and analysis of systems for two important reasons. First, Gauss-Markov processes (and their non-Gaussian generalizations) are excellent models for a wide class of interesting problems and phenomena, including many arising in the design and control of dynamic systems [3, 16, 69], biomedical, seismic and geophysical signal processing [6, 57], and speech and image processing [108, 111, 88]. Second, their simple structure leads naturally



Figure 1-5: A first-order tree, corresponding to the set of integers, is shown. Time recursive models defined on the first-order tree, such as (1.19), provide a rich modeling framework and lead to efficient time-recursive algorithms for 1-D signal processing.

to efficient recursive algorithms for problems such as state and parameter estimation [41, 66, 70, 84]. It is natural to expect then that models of this same form, adapted to higher order trees, can be used to model a wide range of processes and will also lead to efficient algorithms for statistical signal processing. To show this result in an image processing context is one of the main goals of this thesis.

There is, in fact, already substantial evidence that this is true for 1-D signal processing applications. In particular, note that, in the same way that any given node on the first-order tree can be viewed as a boundary between two subsets of nodes on the first-order tree (i.e. the node  $t$  separates the two subsets which consist of the nodes before and after  $t$ ), any given node on the  $q^{\text{th}}$ -order tree can be viewed as a boundary between  $q + 1$  subsets of nodes ( $q$  corresponding to paths leading towards offspring and one corresponding to a path leading towards the parent)<sup>2</sup>. An extremely important property of the scale-recursive model (1.16) is that not only is it Markov from scale-to-scale but, conditioned on the value of the state at any node, the values of the state in the corresponding subsets of nodes are *independent*. This fact is the basis for the development in [7, 30, 27, 29, 31, 32] of an extremely efficient and highly parallelizable algorithm for optimal estimation of processes of the form (1.16) based on noisy measurements  $y(s) \in \mathcal{R}^p$  of the form:

$$y(s) = C(s)x(s) + v(s) \quad (1.20)$$

---

<sup>2</sup>The only exception is the root node, which has no parent and hence separates the tree into just  $q$  subsets of nodes.

where  $v(s) \sim \mathcal{N}(0, R(s))$  and the matrix  $C(s)$  can specify, in a very general way, measurements taken at different times or spatial locations *and* at different scales. The structure of the algorithm consists of two scale-recursive sweeps (a fine-to-coarse sweep followed by a coarse-to-fine recursion), each of which follows the structure of the dyadic tree so that there is substantial parallelism and efficiency. The *upward* or *fine-to-coarse* sweep on the tree propagates the measurement information in parallel, level by level, from the fine scale nodes up to the root node. This sweep is followed by the *downward* or *coarse-to-fine* sweep, which propagates the measurement information back down, and throughout the tree. The result of the algorithm is the least squares estimate of the state  $x(s)$  at each node based on all of the data. In Chapter 2, we use a form of this algorithm adapted to quadrees in the context of image sequence optical flow estimation. The details of the algorithm are given below for models defined on  $q^{\text{th}}$ -order trees, and are discussed (for the case of dyadic trees) in much greater detail in [29, 31].

### 1.3 Multiscale Optimal Estimation

The model given by (1.16) is a downward model in the sense that the recursion starts from the root node and propagates down the tree from coarse-to-fine scales. In order to describe the upward sweep of the smoothing algorithm, we need a corresponding *upward model*. This upward model is equivalent to the downward model in the sense that the joint second order statistics of the states  $x(s)$  and measurements  $y(s)$  are the same. The upward model is given by<sup>3</sup> [27, 29]:

$$x(s\bar{\gamma}) = F(s)x(s) + \bar{w}(s) \tag{1.21}$$

$$y(s) = C(s)x(s) + v(s) \tag{1.22}$$

---

<sup>3</sup>We use  $\mathbf{E}[x]$  to denote the expected value of the random variable  $x$  and  $\mathbf{E}[x|y]$  to denote the expected value of  $x$  given  $y$ .



where:

$$F(s) = P_{s\bar{\gamma}} A^T(s) P_s^{-1} \quad (1.23)$$

$$\mathbf{E}[\bar{w}(s)\bar{w}^T(s)] = P_{s\bar{\gamma}} - P_{s\bar{\gamma}} A^T(s) P_s^{-1} A(s) P_{s\bar{\gamma}} \quad (1.24)$$

$$\equiv Q(s) \quad (1.25)$$

and where  $P_s = \mathbf{E}[x(s)x^T(s)]$  is the variance of the state at node  $s$  and evolves according to the Lyapunov equation:

$$P_s = A(s)P_{s\bar{\gamma}}A^T(s) + B(s)B^T(s) \quad (1.26)$$

To proceed further we need to define some new notation.

$$Y_s = \{y(\sigma) | \sigma = s \text{ or } \sigma \text{ is a descendant of } s\} \quad (1.27)$$

$$Y_s^+ = Y_s \setminus \{y(s)\} \quad (1.28)$$

$$\hat{x}(\sigma|s) = \mathbf{E}[x(\sigma)|Y_s] \quad (1.29)$$

$$\hat{x}(\sigma|s+) = \mathbf{E}[x(\sigma)|Y_s^+] \quad (1.30)$$

$$\hat{P}(\sigma|s) = \mathbf{E}[(x(\sigma) - \hat{x}(\sigma|s))(x(\sigma) - \hat{x}(\sigma|s))^T] \quad (1.31)$$

$$\hat{P}(\sigma|s+) = \mathbf{E}[(x(\sigma) - \hat{x}(\sigma|s+))(x(\sigma) - \hat{x}(\sigma|s+))^T] \quad (1.32)$$

where the notation  $Y_s \setminus \{y(s)\}$  means the measurement  $y(s)$  is not included in the set  $Y_s^+$ . The upward sweep of the smoothing algorithm begins with the initialization of  $\hat{x}(s|s+)$  and the corresponding error covariance  $P(s|s+)$  at the finest level. The initial conditions at this scale reflect the prior statistics of  $x(s)$  at the finest scale, as we have not yet incorporated data. Thus, for every  $s$  at this finest scale we set  $\hat{x}(s|s+)$  to zero (which is the prior mean of  $x(s)$ ) and similarly set  $P(s|s+)$  to the corresponding covariance, namely the solution of the Lyapunov equation (1.26) at the finest level. The upward sweep of the smoothing algorithm then proceeds recursively. Specifically, suppose that we have  $\hat{x}(s|s+)$  and  $P(s|s+)$  at a given node  $s$ . Then this estimate is *updated* to incorporate the measurement  $y(s)$  (if there is a measurement

at this node) according to the following:

$$\hat{x}(s|s) = \hat{x}(s|s+) + K(s)[y(s) - C(s)\hat{x}(s|s+)] \quad (1.33)$$

$$P(s|s) = [I - K(s)C(s)]P(s|s+) \quad (1.34)$$

$$K(s) = P(s|s+)C^T(s)V^{-1}(s) \quad (1.35)$$

$$V(s) = C(s)P(s|s+)C^T(s) + R(s) \quad (1.36)$$

Denote the offspring of  $x(s)$  as  $x(s\alpha_i)$ ,  $i = 1, \dots, q$ . The updated estimates at the offspring nodes are then *predicted* back up to the next level:

$$\hat{x}(s|s\alpha_i) = F(s\alpha_i)\hat{x}(s\alpha_i|s\alpha_i) \quad (1.37)$$

$$P(s|s\alpha_i) = F(s\alpha_i)P(s\alpha_i|s\alpha_i)F^T(s\alpha_i) + Q(s\alpha_i) \quad (1.38)$$

The predicted estimates from the  $q$  offspring are then *merged*:

$$\hat{x}(s|s+) = P(s|s+) \sum_{i=1}^q P^{-1}(s|s\alpha_i)\hat{x}(s|s\alpha_i) \quad (1.39)$$

$$P(s|s+) = [(1 - q)P_s^{-1} + \sum_{i=1}^q P^{-1}(s|s\alpha_i)]^{-1} \quad (1.40)$$

The upward sweep given by the update, predict and merge equations proceeds recursively up the quadtree. At the top of the tree (corresponding to the root node  $s = 0$ ), one obtains the *smoothed* estimate of the root node, that is, an estimate based on all of the data. The estimate and its error covariance are given by:

$$\hat{x}^s(0) = \hat{x}(0|0) \quad (1.41)$$

$$P^s(0) = P(0|0) \quad (1.42)$$

where the superscript  $s$  denotes the fact that these are smoothed estimates. The smoothed estimate and associated error covariance at the root node provide initializa-

tion for the *downward sweep*, which is given by the following coarse-to-fine recursion:

$$\hat{x}^s(s) = \hat{x}(s|s) + J(s)[\hat{x}^s(s\bar{\gamma}) - \hat{x}(s\bar{\gamma}|s)] \quad (1.43)$$

$$P^s(s) = P(s|s) + J(s)[P^s(s\bar{\gamma}) - P(s\bar{\gamma}|s)]J^T(s) \quad (1.44)$$

$$J(s) = P(s|s)F^T(s)P^{-1}(s\bar{\gamma}|s) \quad (1.45)$$

The form of the algorithm we have specified here, which generalizes standard Kalman filtering and smoothing algorithms to the multiscale context, obviously assumes that the state covariance  $P_s$  is well defined and finite, and it is not difficult to see from (1.26) that this will be the case if  $P_0$  is finite. There is, however, an alternate form of this algorithm presented in [29, 31] which generalizes so-called *information form* algorithms for standard state space models and which propagates *inverses* of covariances. In this alternate form it is straightforward to accommodate the setting of  $P_0$  to infinity (which corresponds to  $P_0^{-1} = 0$ ), and we refer the reader to [29, 31] for details.

Finally, note that a recursion for the covariance of the error process  $\tilde{x}^s(s) \equiv x(s) - \hat{x}^s(s)$  associated with the two-sweep smoothing algorithm is provided by (1.44). In fact, information about the *correlation structure* of the error process can also be obtained. That is, whereas (1.44) allows us to obtain  $\mathbf{E}\{\tilde{x}^s(s)(\tilde{x}^s(s))^T\}$ , we show in Appendix A how to obtain cross-correlations of the form  $\mathbf{E}\{\tilde{x}^s(s)(\tilde{x}^s(\sigma))^T\}$ . In particular, we show that  $\tilde{x}^s(s)$  satisfies a white noise driven difference equation of the form (1.16), and we obtain a simple form for the model parameters. The Lyapunov equation associated with this model can then be used to obtain correlations between errors at different nodes on the tree. Applications of the model for the smoothing error are discussed in Chapter 5.

## 1.4 Other Related Work

In addition to the previous work on multiresolution methods in signal processing, including that on wavelet transforms, which we have discussed in Section 1.1, and

the research in [7, 30, 27, 29] on which much of our work is built, a number of other researchers have developed image processing algorithms in similar multiscale stochastic modeling frameworks. In particular, in [32], a scalar version of the model (1.16) with  $A(s) = 1$  and  $B(s)$  assumed to vary only as a function of scale is proposed for image modeling. A smoothing algorithm consistent with that in the previous section is derived along with an approach to adaptively estimating  $B(s)$  across scales.

In addition, in [17, 19, 20] a model is proposed for a class of discrete-state processes defined on trees. The model has the same recursive structure as (1.16), and leads to an efficient two-sweep algorithm for computing maximum a-posteriori estimates of the process based on noisy observations. The modeling framework was applied to a problem of image segmentation. While our work here does not directly build on either this research or that in [32], there is one interesting point of contact to be made. In particular, it was noted in the above works that processing based on the quadtree models led to blocky results which could differ significantly for different positionings of the tree with respect to the finest scale lattice, and this fact led to the need for more complex processing structures. Indeed, in [32] it was necessary to perform processing several times with different tree positions and to average the results, while in [17, 19, 20] the tree was replaced by a more connected lattice (so that any two nodes have both common ancestors *and* common descendants). The latter modification, however, destroys the partially-ordered Markovian structure which the tree processes possess and which leads to highly parallelizable and scale-recursive algorithms such as that in the previous section and in Chapter 4 of this thesis where we develop a likelihood calculation algorithm for the multiscale processes (1.16). In Chapter 3, we show that one can avoid the apparent problems in using quadtree models by demonstrating that one can model *any* MRF *exactly* using such a model.

## 1.5 Thesis Organization

As mentioned previously, in this thesis we will demonstrate that, especially for 2-D applications, models of the form (1.16) provide a rich framework for practical ap-

plications and lead to very efficient signal processing algorithms. To begin then, in Chapter 2 we illustrate how this class of models can be used as a means of regularizing ill-posed inverse problems, using as a vehicle for this development the problem of computing dense optical flow fields in an image sequence. Standard formulations of this problem require the computationally intensive solution of an elliptic partial differential equation which arises from the often used “smoothness constraint” type regularization. We utilize the interpretation of the smoothness constraint as a “fractal prior” to motivate regularization based on multiscale models of the form (1.16). The solution of the new problem formulation is computed with the multiscale smoothing algorithm discussed in Section 1.3 and experiments on several image sequences demonstrate the substantial computational savings that can be achieved. Such savings result from the fact that the algorithm is non-iterative and in fact has a per pixel computational complexity which is independent of image size. The new approach also has a number of other important advantages. Specifically, multiresolution flow field estimates are available at no extra cost, allowing great flexibility in dealing with the tradeoff between resolution and accuracy. Multiscale error covariance information is also available, which is of considerable use in assessing the accuracy of the estimates. In particular, these error statistics can be used as the basis for a rational procedure for determining the spatially-varying optimal reconstruction resolution, in addition to providing the information necessary for the statistically optimal fusion of these estimates with other data or prior information. Furthermore, if there are compelling reasons to insist upon a standard smoothness constraint, the new approach provides an excellent initialization for the iterative algorithms associated with the smoothness constraint problem formulation.

In Chapter 3 we describe how 1-D Markov processes and 2-D Markov random fields (MRF’s) can be represented within the multiscale modeling framework of (1.16). Markov processes in 1-D and 2-D Markov random fields are widely used classes of models for analysis, design and statistical inference. As we have stated, the recursive structure of 1-D Markov processes makes them simple to analyze, and generally leads to computationally efficient algorithms for statistical inference. On the other hand,

2-D MRF's are well known to be very difficult to analyze due to their non-causal structure, and thus their use typically leads to computationally intensive algorithms for smoothing and parameter identification. Our multiscale representations of these processes are based on the scale-recursive model (1.16), thus providing a framework for the development of new and efficient algorithms for Markov processes and MRF's. In 1-D, the representation generalizes the mid-point deflection construction of Brownian motion. In 2-D, we use a further generalization to a "mid-line" deflection construction. Our exact representations of 2-D MRF's are of potentially high dimension, and this fact motivates our development of a class of approximate models based on *one-dimensional* wavelet transforms. We demonstrate the use of these models in the context of texture representation and in particular, we show how they can be used as approximations for, or alternatives to, well-known MRF texture models. We illustrate how the quality of the representations varies as a function of the underlying MRF and the complexity of the wavelet-based approximate representation.

Our development in Chapter 2 of an algorithm based on (1.16) for computing optical flow in an image sequence, and the construction in Chapter 3 of multiscale models for representing Markov random fields, provides substantial evidence that the multiscale model class can be used to model a broad range of phenomena and allows for the development of efficient image processing algorithms. Further algorithmic development is the focus of Chapter 4 in which we introduce an algorithm for computing likelihoods for Gaussian multiscale models on  $q^{\text{th}}$  order trees. That is, we consider the problem of computing the conditional probability of a set of noisy observations, given that the data corresponds to a particular multiscale model. Our development exploits the scale-recursive structure of the multiscale models thereby leading to a computationally efficient and highly parallelizable algorithm. The algorithm is non-iterative and in fact has a constant per node computational complexity. We illustrate one possible application of the algorithm to texture discrimination and demonstrate that likelihood-based methods using our algorithm and the results in Chapter 3 for modeling GMRF's have substantially better probability of error characteristics than well-known least-squares methods, and achieve performance comparable to that of

GMRF-based techniques, which in general are prohibitively complex computationally.

Finally, in Chapter 5 we summarize the contributions of this thesis and discuss directions for further research.

## Chapter 2

# Applications of Multiscale Regularization to the Computation of Optical Flow

### 2.1 Introduction

In this chapter we introduce and develop a new multiscale approach to regularization problems in image processing, using the computation of dense optical flow fields as the vehicle for our development. Regularization is, of course, a widely-known and used concept in image analysis. In some cases the introduction of a regularizing term is necessitated by ill-posedness (also referred to as the “aperture problem” in computer vision), i.e. by the insufficient information provided solely by the available data, or by a desire to reduce noise. In other problems the so-called regularizing term represents substantive prior information arising, for example, from physical constraints or laws or from information extracted from previous image frames. The family of optical flow reconstruction algorithms stemming from the work of Horn and Schunck [62], which forms the specific context for our development and which has found success in a number of applications such as [197], is one example of a formulation typically introduced to deal with ill-posedness. However, very similar formulations arise in other contexts ranging from the problem of the temporal tracking of optical flow [26]



to large scale oceanographic data assimilation problems [120]. Thus, while we use the problem of estimating optical flow at a single point in time as the focus for our development, it is our strong belief that the ideas developed here have a far broader range of applicability.

Optical flow, the apparent velocity vector field corresponding to the observed motion of intensity patterns in successive image frames, is an important quantity in a variety of problems. For example, in MRI imaging of the heart [107, 104] this vector field provides diagnostic information concerning cardiac muscle motion and differential strain. In oceanographic data processing such information can be of use, for example, in tracking the meandering motion of the Gulf Stream [92]. Also, in computational vision, optical flow is an important input into higher level vision algorithms performing tasks such as segmentation, tracking, object detection, robot guidance and recovery of shape information [2, 98, 112, 122, 130]. In addition, methods for computing optical flow are an essential part of motion compensated coding schemes [4, 143].

As we have indicated, our approach to optical flow estimation is motivated by, and represents an alternative to, regularization methods such as that of Horn and Schunck [62] which employs the often used “smoothness constraint” regularization term. In particular, the starting point for this and many other approaches to optical flow estimation is the use of a brightness constraint, i.e. the assumption that changes in image brightness are due only to motion in the image frame. This leads to the so-called *brightness constraint equation*<sup>1</sup> [62]:

$$0 = \frac{d}{dt}E(z_1, z_2, t) = \frac{\partial}{\partial t}E(z_1, z_2, t) + \nabla E(z_1, z_2, t) \cdot \mathbf{x}(z_1, z_2, t) \quad (2.1)$$

where  $E(z_1, z_2, t)$  is the image intensity as a function of time  $t$  and space  $(z_1, z_2)$ ,

---

<sup>1</sup>More generally, it is straightforward to adapt (2.1) to cases in which  $E$  has a known temporal variation. See [107] for an example in the context of MRI imaging.

$x(z_1, z_2, t)$  is the optical flow vector field, and:

$$x = \begin{bmatrix} \frac{\partial z_1}{\partial t} & \frac{\partial z_2}{\partial t} \end{bmatrix}^T \quad (2.2)$$

$$\nabla E = \begin{bmatrix} \frac{\partial E}{\partial z_1} & \frac{\partial E}{\partial z_2} \end{bmatrix} \quad (2.3)$$

The brightness constraint equation (2.1), however, does not completely specify the flow field  $x(z_1, z_2, t)$  since it provides only one linear constraint for the two unknowns at each point. Thus, (2.1) by itself represents an under-determined or ill-posed set of constraints on optical flow. In addition, in practice, only noisy measurements of the temporal and spatial intensity derivatives will be available, meaning that we in fact have available only noisy constraints. For both of these reasons one must regularize the problem of reconstructing  $x(z_1, z_2, t)$ , and one commonly used way to do this is to assume some type of spatial coherence in the optical flow field, for instance by assuming that  $x(z_1, z_2, t)$  is constant over spatial patches or by other methods for imposing coherence and achieving spatial noise averaging.

In particular, Horn and Schunck's approach [62], often referred to as imposing a *smoothness constraint*, consists of constructing the optical flow field estimate as the solution of the following optimization problem:

$$\hat{x}_{SC} = \arg \min_x \iint R^{-1} \left( \frac{d}{dt} E \right)^2 + \|\nabla x\|^2 dz_1 dz_2 \quad (2.4)$$

The smoothness constraint is captured by the second term which penalizes large gradients in the optical flow. The constant  $R$  allows one to tradeoff between the relative importance in the cost function of the brightness and smoothness constraint terms. For example, in some situations  $R^{-1}$  is taken to be quite large to force the solution to match the constraints (2.1), and in such a case the smoothness constraint serves merely to regularize the problem, i.e. to ensure that (2.4) has a unique solution. In other cases, however, one might use a more moderate value of  $R^{-1}$  either to account for the fact that the constraint (2.1) is noisy or to reflect the fact that the smoothness constraint penalty represents a useful source of information itself. For example,

in [26] the smoothness constraint is replaced by an analogous term reflecting both smoothness and prior information gleaned from preceding image frames. We refer to the optical flow estimate obtained from (2.4) as the smoothness constraint (SC) solution to the problem of computing optical flow.

One of the major problems associated with the formulation in (2.4) and with analogous formulations for other regularized image processing problems is that they lead to computationally intensive algorithms. Specifically, one can show that the solution of (2.4) satisfies an elliptic partial differential equation (PDE) [62]. Discretization of this PDE leads to a sparse but extremely large set of linear equations which are typically solved using iterative approaches. Terzopoulos [134] proposed the use of *multigrid* approaches and reported a factor of 7 reduction in computation over the Gauss-Seidel approach. Successive over-relaxation (SOR) algorithms [75] also provide significant computational improvement over GS approaches and have been successfully used in [107, 116, 117]. However, whatever numerical method is employed, computational complexity per pixel typically grows with image size, a fact that can make real-time or in some cases even off-line implementation prohibitively complex. For example, while computational complexity for such a problem may be severe for  $512 \times 512$  images, especially if real-time processing of image sequences is required, the computational demands in other contexts, such as oceanographic data processing where one may consider problems as large as 100,000,000 voxels (3-D pixels), are more than a serious problem: they are, in fact, the *major* problem.

One of the principal motivations for the method proposed in this chapter is the computational challenge discussed above. To do this, we need to analyze the smoothness constraint in more detail. Note in particular that the penalty associated with the smoothness constraint term in (2.4) is equal to the integral of the squared norm of the field gradient over the image plane. In a one-dimensional context, such a constraint would penalize each of the (one-dimensional) fields in Figure 2-1 equally. Intuitively, the smoothness constraint has a fractal nature, and in fact is often referred to as a “fractal prior” [131].

Moreover, as discussed in [116, 117] and as described in more detail in the next

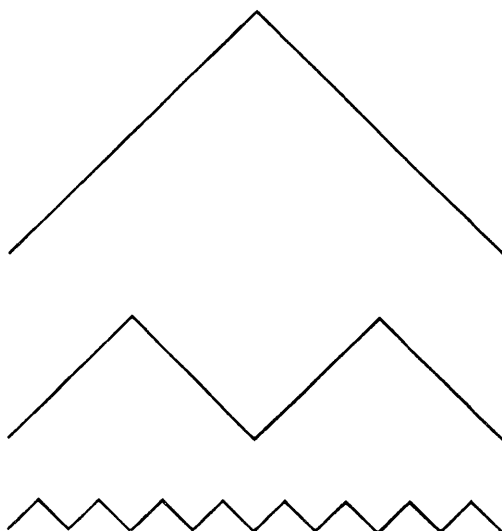


Figure 2-1: Depiction of three fields which are equally favored by the smoothness constraint, illustrating how this penalty provides a fractal prior model for the optical flow.

section, the optical flow problem formulation in (2.4) has an equivalent formulation and precise interpretation in an estimation-theoretic context. Roughly speaking, the optimization problem (2.4) corresponds to a statistical model in which the noise or error in the brightness constraint is assumed to be spatially white and in which the two components of the optical flow are modeled as independent random fields, each of which has a zero mean, spatially white gradient. That is, as discussed in [26, 116, 117], the smoothness constraint essentially corresponds to modeling each component of optical flow as a spatial Brownian motion, i.e. as a statistically self-similar, fractal process with a  $1/|f|^2$  generalized spectrum [131].

Given that the smoothness constraint corresponds precisely to a prior model with fractal characteristics, a natural idea is that of using alternate prior statistical models — corresponding to alternate penalty terms to that in (2.4) — that possess the same type of fractal characteristics but that lead to computationally more attractive problem formulations. In this chapter, we do just that as we introduce an approach based on substituting the class of prior models introduced in Chapter 1 for the smoothness constraint prior. The key idea behind this approach is that instead of the Brownian motion fractal prior that describes the optical flow field as one that has independent

increments in space, we use a statistical model for optical flow that has independent increments in *scale*. The model can be interpreted as a smoothness constraint that provides individual penalties on each scale of detail or as providing a multiscale probabilistic model in which the variances of the detail components vary from scale to scale in a fractal, self-similar fashion. For this reason, we say that our formulation corresponds to a multiscale regularization (MR) of the optical flow problem, and we refer below to the MR algorithm and solution.

One of the most important consequences of this alternate smoothness constraint is that it allows us to make use of the extremely efficient scale-recursive optimal estimation algorithm discussed in Chapter 1. In particular, the resulting algorithm is not iterative and in fact requires a fixed number of floating point operations per pixel *independent of image size*. Thus, since iterative methods for solving the smoothness constraint problem formulation have per pixel computational complexities that typically grow with image size, the computational savings associated with the new approach increase as the image size grows and, as we will see, can be considerable even for modest-sized problems.

Moreover, while computational efficiency did serve as the original motivation for this new formulation and in many problems may be its most important asset, there are several other potential advantages that the new approach has. First, the scale-recursive nature of the algorithm directly yields estimates of the optical flow field at multiple resolutions, providing us with considerable flexibility in dealing with the tradeoff between accuracy and resolution. Specifically, one can expect to obtain higher accuracy at coarser resolutions, and thus one can imagine trading off resolution versus accuracy in a data-adaptive way. For example, in regions with substantial local intensity variations one would expect to be able to estimate optical flow at a finer spatial resolution than in regions in which intensity varies more smoothly and contrast is low. The question, of course, is how such an intuitive concept can be realized in an algorithm. As we will demonstrate, our multiscale algorithm provides us with all of the information required to do this with essentially no additional computation, leading to a simple approach to designating the preferred resolution for estimating

optical flow at every point in the image frame.

Secondly, an important consequence of employing an estimation-theoretic interpretation is that it offers the possibility of evaluating a quantitative measure of the quality of our optical flow estimate, namely the estimation error covariance. This idea, of course, also applies to the original smoothness constraint formulation (2.4). However, in that case, the computation of the error covariance must be done in addition to solving the partial differential equations for the optimal flow estimates, and in fact, the computation of these error statistics has complexity at least as great as that for calculating the estimates. In contrast, for our formulation, error covariances can be calculated with essentially no increase in computational complexity. Furthermore, our algorithm provides error covariance statistics at multiple resolutions, providing information that is essential to addressing the tradeoff between resolution and accuracy as discussed in the previous paragraph, and that may also be useful to higher level vision algorithms which need to combine information in a rational way from a variety of sources.

As we have indicated, the new algorithm we develop is based on a problem formulation that is similar but not identical to that given by (2.4), and there are several implications of this fact. The first is that while the estimates produced by our algorithm are not identical to those based on (2.4), they are similar and have comparable root-mean-square (rms) error characteristics, as the experimental evidence in Section 2.3 illustrates. Moreover, these results also show that the difference between the SC and MR flow estimates consists of mostly high spatial frequency components, which are precisely the components which can be quickly removed by the iterative algorithms computing a smoothness constraint solution. Thus, even in situations in which a solution to the original smoothness constraint formulation is required (for instance, if the smoothness constraint corresponds to physically-based prior information) there may be considerable computational advantages in using the MR solution as an initial estimate of the optical flow, i.e. as an initial estimate for an iterative algorithm which computes the solution of the partial differential equation characterizing (2.4). Indeed, given the promise suggested by results presented here, we conjecture

that another potential application of the approach we introduce is in providing easily computed, accurate initial estimates for the solution of partial differential equations arising in contexts other than image processing.

There is another implication of the relationship of our approach to the formulation in (2.4). Specifically, there are of course, problems of practical importance in which the basic assumptions underlying the Horn and Schunck formalism are violated, for instance if there is substantial temporal aliasing (so that the data implied by (2.1) are not available), if there are discontinuities in the motion field due to object boundaries and occlusion or if there are multiple motions. In such cases, the Horn and Schunck formulation may fail to give adequate results, and, due to the similarity of the approaches, our method would likely fail as well. In such contexts algorithms developed to deal explicitly with such issues, such as those in [52, 60], may be more appropriate. On the other hand, for the not insignificant class of problems for which the Horn and Schunck formulation is well-suited, such as [107] and the many ill-posed and variational problems arising in fields ranging from image processing and tomography to meteorology, seismology and oceanography [12, 106, 137, 78, 136, 97], our method will also work well and also provides the advantages described previously: computational efficiency, multiresolution estimates and multiscale error covariance information. Moreover, even in cases in which Horn and Schunck-type global smoothness constraints are inappropriate, there are reasons to believe that algorithms based on our formulation may provide the basis for promising new solutions. While it is beyond the scope of this thesis to develop such methods in detail, we provide an example suggesting this promise and also indicate how the statistical interpretation and flexible structure of our formalism might be used to advantage.

This chapter is organized as follows. In Section 2.2 we discuss in more detail an estimation-theoretic interpretation of the optical flow formulation in (2.4) and develop our new approach to the computation of optical flow. Section 2.3 presents experimental results on several real and synthetic image sequences. Section 2.4 provides further discussion and a summary of the results.

## 2.2 Multiscale Regularization

In the first part of this section we develop a discrete formulation of the optical flow problem, and discuss in more detail the estimation-theoretic interpretation of it. We then illustrate precisely how the smoothness constraint can be interpreted as a prior model for the flow field, and how it can be replaced by a multiscale model of the form (1.16), hence leading to a more computationally attractive problem formulation. The quadtree models we use, i.e. the specific choices for the parameters in (1.16), are then discussed in detail.

Estimation-theoretic formulations and interpretations of optical flow problems have been introduced and studied by a number of authors. For instance, in [73, 142] Markov random field (MRF) models are proposed along with a *maximum a-posteriori* criterion for estimating optical flow. MRF models are also used in [60] to address problems of occlusion and flow field discontinuity. Kalman filtering approaches which allow for temporal as well as spatial smoothness constraints have been discussed in [26, 124, 59, 129]. In addition, in [123] a Bayesian formulation which provides optical flow estimates and confidence measures based on a local window of data is proposed. In addition there is the interpretation by Rougée et al. [116, 117] of the Horn and Schunck smoothness constraint formulation (2.4) as an equivalent estimation problem with a Brownian motion, fractal prior for the flow field. The distinguishing feature of the Brownian motion model implied by (2.4), the Markov random field models, and the spatio-temporal models used in the Kalman filtering approaches, is that they all provide models in terms of local relationships (typically nearest neighbor) of the flow field components at a single, finest level of resolution. This leads naturally to spatially local, iterative algorithms for computing the optimal optical flow estimates (such as those needed to solve the partial differential equation resulting from (2.4) or simulated annealing algorithms for MRF models). In contrast, the probabilistic model for optical flow proposed in this chapter describes the flow field in terms of probabilistic variations from scale to scale and leads naturally to the efficient scale recursive algorithms described in Chapters 1 and 4.



As we have indicated, our approach is motivated by the probabilistic interpretations of Horn and Schunck's formulation, which we now discuss briefly. The reader is referred to [25, 26, 116, 117] for a more extensive discussion of this and related probabilistic models. We start by introducing the following notation. Define:

$$y(z_1, z_2) \equiv -\frac{\partial}{\partial t} E(z_1, z_2, t) \quad (2.5)$$

$$C(z_1, z_2) \equiv \nabla E(z_1, z_2, t) \quad (2.6)$$

The brightness constraint equation (2.1) can then be written:

$$y(z_1, z_2) = C(z_1, z_2) \cdot x(z_1, z_2) \quad (2.7)$$

where the time dependence of the equations has been suppressed.

In practice, brightness measurements are only available over a discrete set of points in space and time. Thus, the temporal and spatial derivative terms in the brightness constraint equation (2.7) must be approximated by a finite difference scheme in time and space, and the optical flow is only estimated on a discrete space-time grid. There are a number of important issues which arise due to the discretization, such as the use of spatial and/or temporal smoothing prior to discretization, the use of more than two image frames in the computation of temporal derivatives, etc., and we refer the reader to [25, 5, 52] for further discussion. We assume here that the optical flow is to be estimated on the set  $\{(z_1, z_2) | z_1 = ih, z_2 = jh; i, j \in \{1, \dots, 2^M\}\}$  where  $h$  is the grid spacing and  $M$  is an integer. The assumption that the lattice is square and that the number of rows is equal to a power of two simplifies the notation in the subsequent development, but is not essential as we discuss in Appendix B.1. In order to simplify the notation further, we let  $y(i, j)$ ,  $x(i, j)$ , and  $C(i, j)$  denote the measured temporal brightness derivative, the optical flow, and the spatial gradient of the image brightness, respectively, at grid point  $(ih, jh)$ . The brightness constraints at all grid points can then be grouped into one large set of linear equations to capture the optical flow information contained in the image sequence. Defining  $\mathbf{x}$  as the vector

of optical flow vectors  $x(i, j)$  at all grid points (using, say, a lexicographic ordering),  $\mathbf{C}$  as the matrix containing the corresponding spatial gradient terms  $C(i, j)$ , and  $\mathbf{y}$  as the vector of temporal gradients  $y(i, j)$ , we can write:

$$\mathbf{y} = \mathbf{C}\mathbf{x} \quad (2.8)$$

Then, the discrete counterpart of (2.4) is:

$$\begin{aligned} \hat{\mathbf{x}}_{SC} &\equiv \arg \min_{\mathbf{x}} \|\mathbf{y} - \mathbf{C}\mathbf{x}\|_{\mathbf{R}^{-1}}^2 + \|\mathbf{L}\mathbf{x}\|_{\mathbf{I}}^2 \\ &= \arg \min_{\mathbf{x}} (\mathbf{y} - \mathbf{C}\mathbf{x})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{C}\mathbf{x}) + \mathbf{x}^T \mathbf{L}^T \mathbf{L} \mathbf{x} \end{aligned} \quad (2.9)$$

where the matrix  $\mathbf{L}$  is a discrete approximation of the gradient operator in (2.4) and  $\mathbf{R} = R\mathbf{I}$ , where  $\mathbf{I}$  is the identity matrix. The regularization term  $\mathbf{x}^T \mathbf{L}^T \mathbf{L} \mathbf{x}$  makes the optimization problem (2.9) well-posed. In particular, the solution of (2.13) satisfies the so-called *normal equations* [128]:

$$(\mathbf{C}^T \mathbf{R}^{-1} \mathbf{C} + \mathbf{L}^T \mathbf{L}) \hat{\mathbf{x}}_{SC} = \mathbf{C}^T \mathbf{R}^{-1} \mathbf{y} \quad (2.10)$$

and the invertibility of  $(\mathbf{C}^T \mathbf{R}^{-1} \mathbf{C} + \mathbf{L}^T \mathbf{L})$  guarantees that  $\hat{\mathbf{x}}_{SC}$  is unique. The normal equations (2.10) are the discrete counterpart of the partial differential equation that arises from (2.4).

An estimation-theoretic formulation of the optimization problem in (2.9) can now be developed. Specifically, suppose that we wish to estimate  $\mathbf{x}$  based on the measurements

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{v} \quad (2.11)$$

$$\mathbf{0} = \mathbf{L}\mathbf{x} + \mathbf{w} \quad (2.12)$$

where  $\mathbf{v}$  and  $\mathbf{w}$  are uncorrelated random vectors with<sup>2</sup>  $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$  and  $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ .

---

<sup>2</sup>The notation  $\mathbf{z} \sim \mathcal{N}(\mathbf{m}, \mathbf{\Lambda})$  means that  $\mathbf{z}$  has a Gaussian distribution, with mean  $\mathbf{m}$  and covariance  $\mathbf{\Lambda}$ .

Then the measurement vector  $\bar{\mathbf{y}} \equiv [\mathbf{y}^T | \mathbf{0}]^T$  is conditionally Gaussian, and the maximum likelihood estimate [139] of  $\mathbf{x}$  is:

$$\begin{aligned}
\hat{\mathbf{x}}_{ML} &\equiv \arg \max_{\mathbf{x}} p(\bar{\mathbf{y}}|\mathbf{x}) \\
&= \arg \min_{\mathbf{x}} -\log p(\bar{\mathbf{y}}|\mathbf{x}) \\
&= \arg \min_{\mathbf{x}} (\mathbf{y} - \mathbf{C}\mathbf{x})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{C}\mathbf{x}) + \mathbf{x}^T \mathbf{L}^T \mathbf{L} \mathbf{x} \quad (2.13) \\
&= \hat{\mathbf{x}}_{SC}
\end{aligned}$$

Thus, the maximum likelihood problem formulation results in the same solution as the smoothness constraint formulation when  $\mathbf{L}$  is used to define an additional set of noisy measurements.

The main point here is that by formulating the problem in this estimation-theoretic framework, we can use (2.12) to interpret the smoothness constraint as a prior probabilistic model for the flow field. Specifically, we can rewrite (2.12) as:

$$\mathbf{L}\mathbf{x} = -\mathbf{w} \quad (2.14)$$

Recalling that  $\mathbf{L}$  is an approximation to the gradient operator, we see that (2.14) is nothing more than a spatial difference equation model for  $\mathbf{x}$  driven by the spatial white noise field  $\mathbf{w}$ .

To some extent the precise form of this prior model is arbitrary, and thus we are led to the idea of introducing a new prior model which is similar in nature, but which leads to a computationally more attractive problem formulation. That is, we want to change the smoothness constraint term  $\mathbf{x}^T \mathbf{L}^T \mathbf{L} \mathbf{x}$  in (2.13) to something similar, say,  $\mathbf{x}^T \mathbf{S} \mathbf{x} \approx \mathbf{x}^T \mathbf{L}^T \mathbf{L} \mathbf{x}$  (where  $\mathbf{S}$  is a symmetric positive semi-definite matrix) such that the resulting optimization problem is easy to solve. If we factor  $\mathbf{S}$  as  $\mathbf{S} = \bar{\mathbf{L}}^T \bar{\mathbf{L}}$  then we can interpret the new constraint as a prior probabilistic model just as we did with the smoothness constraint. In addition, there is a precise interpretation of what we have done as a Bayesian estimation problem. Specifically, if  $\mathbf{S}$  is invertible, then the use of this new constraint in place of the smoothness constraint is equivalent to modeling

the flow field probabilistically as  $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{S}^{-1})$ , since in this case the Bayes' least squares estimate of the flow field  $\mathbf{x}$ , given this prior model and the measurements in (2.11) is given by:

$$\hat{\mathbf{x}}_{BLSE} = \arg \min_{\mathbf{x}} (\mathbf{y} - \mathbf{C}\mathbf{x})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{C}\mathbf{x}) + \mathbf{x}^T \mathbf{S} \mathbf{x} \quad (2.15)$$

which corresponds to (2.13) with a different prior model term. The normal equations corresponding to (2.15) are given by:

$$(\mathbf{C}^T \mathbf{R}^{-1} \mathbf{C} + \mathbf{S}) \hat{\mathbf{x}}_{BLSE} = \mathbf{C}^T \mathbf{R}^{-1} \mathbf{y} \quad (2.16)$$

Comparison of the problem formulations (2.9) and (2.15), or of the normal equations (2.10) and (2.16), makes it apparent how the two problem formulations are related. Note that an analogous Bayesian interpretation can apparently be given to the smoothness constraint formulation (2.9), (2.10), with the corresponding prior model for optical flow given by  $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, (\mathbf{L}^T \mathbf{L})^{-1})$ . Recall, however, that  $\mathbf{L}$  is an approximation to the spatial gradient operator and thus is *not* invertible since operating on constants with this operator yields zero. The probabilistic interpretation of this is that the model (2.14) places probabilistic constraints on the spatial *differences* of the optical flow, but not on its DC value. Indeed, it is not difficult to check that if we model optical flow instead as  $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, (\mathbf{L}^T \mathbf{L} + \epsilon \mathbf{I})^{-1})$ , where  $\epsilon$  is any arbitrarily small positive number, then  $\mathbf{L}^T \mathbf{L} + \epsilon \mathbf{I}$  is indeed invertible and the DC value of  $\mathbf{x}$  has a prior covariance  $P_0$  on the order of  $1/\epsilon$ , so that  $P_0 \rightarrow \infty$  as  $\epsilon \rightarrow 0$ . Thus, the original smoothness constraint formulation in essence assumes an infinite prior covariance on the DC value of optical flow.

We propose now the replacement of the smoothness constraint model by a multiscale model of the form (1.16) defined on a quadtree. The state of this model represents the optical flow at a range of scales,  $m = 0, 1, \dots, M$ . As illustrated in Figure 1-3, at the  $m^{\text{th}}$  scale the field consists of  $4^m$  flow vectors. At the finest level, the flow field is represented on a grid with the same resolution as the image brightness data. In particular,  $x_M(i, j)$  corresponds to the optical flow vector  $x(i, j)$ .

Measurements of the finest level optical flow field are available from the brightness constraint. In particular, at a particular node  $s$  at the finest level  $M$ , we have the measurement equation exactly of the form (1.20, with  $v(i, j) \sim \mathcal{N}(0, R)$ , assumed to be independent of the process driving noise  $w(s)$  and the initial condition  $x_0$ , and where  $C(s) \in \mathbb{R}^{1 \times 2}$ . Of course, we can group the state variables  $x(s)$  at the finest level into a vector  $\mathbf{x}_M$  as well as the corresponding measurements  $y(s)$  and spatial gradient terms  $C(s)$  in the same way as we did to get (2.8):

$$\mathbf{y} = \mathbf{C}\mathbf{x}_M + \mathbf{v} \quad (2.17)$$

$$\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}) \quad (2.18)$$

We now have exactly the framework which led to the statement of (2.15) as a generalization of the smoothness constraint formulation (2.13). In particular, (1.16) indicates that at the finest level of the quadtree, the flow field vectors will be a set of jointly Gaussian random variables  $\mathbf{x}_M \sim \mathcal{N}(\mathbf{0}, \Lambda)$ , where  $\Lambda$  is implicitly given by the parameters in (1.16), and a set of noisy measurements given by (2.17). The Bayes' least squares estimate of  $\mathbf{x}_M$  given the measurements in (2.17) and the prior model (1.16) is:

$$\hat{\mathbf{x}}_M = \arg \min_{\mathbf{x}_M} (\mathbf{y} - \mathbf{C}\mathbf{x}_M)^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{C}\mathbf{x}_M) + \mathbf{x}_M^T \Lambda^{-1} \mathbf{x}_M \quad (2.19)$$

The multiscale modeling framework thus provides an alternative to the smoothness constraint formulation of (2.9) or (2.13). Furthermore, if we drop the assumption of Gaussianity for  $x_0, w(s)$ , and  $v(i, j)$ , the optimal estimate  $\hat{\mathbf{x}}_M$  has the interpretation as the linear least squares estimate of  $\mathbf{x}$ .

What remains to be done are (1) to specify a model within this class that has characteristics similar to those of the smoothness constraint prior model, and (2) to demonstrate why the use of this alternate multiresolution formulation is of interest. We defer the latter of these to the next section and focus here on the former. In particular, for our multiscale model (1.16) to approximate the smoothness constraint

prior we would like to choose our model parameters so that we have  $\Lambda^{-1} \approx \mathbf{L}^T \mathbf{L}$ . The observation that the prior model implied by the operator  $\mathbf{L}$  in (2.13) corresponds to a Brownian motion “fractal prior” suggests one approach to choosing the model parameters. In particular, the one-dimensional Brownian motion has a  $1/f^2$  generalized spectrum [89]. It has been demonstrated that such processes are well approximated by multiscale models such as ours in one dimension if geometrically decreasing powers of noise are added at each level  $m$  of the process [29, 150, 151, 152]. This motivates the choice of  $B(s) = b4^{-\frac{\mu m(s)}{2}} I$  in (1.16), where  $I$  is the  $2 \times 2$  identity matrix, and where  $b$  and  $\mu$  are scalar constants. The constant  $b$  directly controls the overall noise power in the process. Also, as discussed in [150, 151, 152], the choice of  $\mu$  controls the power law dependence of the generalized spectrum of the process at the finest resolution as well as the fractal dimension of its sample paths. Specifically, this spectrum has a  $1/f^\mu$  dependence and the choice of  $\mu = 2$  would correspond to a Brownian-like fractal process.

Thus, we use the following parameterization of the optical flow field model (1.16) and measurement (1.20):

$$x(s) = x(s\bar{\gamma}) + (b4^{-\frac{\mu m(s)}{2}})w(s) \quad (2.20)$$

$$y(s) = C(s)x(s) + v(s) \quad (2.21)$$

with  $v(s) \sim \mathcal{N}(0, R(s))$ ,  $x_0 \sim \mathcal{N}(0, pI)$ , and where  $I$  is a  $2 \times 2$  identity matrix. In the context of the optical flow estimation problem, measurements are taken only at the finest scale, corresponding to  $C(s) = 0$  unless  $s$  is a node on the finest scale. From (2.20) we see that the two components of the optical flow field are modeled as independent sets of random variables, and that each has a fractal-like characteristic due to the form of the driving noise gain  $B(s)$ . The independence of the flow field components is motivated by the fact that the smoothness constraint formulation implicitly makes this assumption as well [116, 117]. We view  $\mu$  and  $b$  as free model parameters which can be varied to control the degree and type of regularization in much the same way that the parameter  $R$  in the smoothness constraint formulation

(2.4) is used to tradeoff between the data dependent and regularization terms in the optimization functional. However, we have found in our experiments that the choice  $b = \mu = 1$  typically works well, and we have used these values in all of the experiments below.

As discussed previously, the measurements  $y(s)$  and measurement matrix  $C(s)$  come directly from the image temporal and spatial gradients, which are available at the finest level of the quadtree. In the experiments described below, we smoothed the images with the  $7 \times 7$  filter given by:

$$\begin{bmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{bmatrix} * \begin{bmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{bmatrix} * \dots * \begin{bmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{bmatrix} \quad (2.22)$$

(where  $*$  denotes the 2-D convolution) and then calculated spatial gradients with a central difference approximation. The temporal gradient was computed as the difference of two smoothed images. Temporal smoothing (in addition to the spatial smoothing) has been shown to reduce estimation errors in several methods, including the smoothness constraint approach [5] and thus would be of value for the multiscale regularization method as well. For our purposes here, however, namely to demonstrate comparative computational efficiency relative to the smoothness constraint formulation and to illustrate the use and value of both multiresolution estimates and covariance information, the simple two frame difference is sufficient.

The additive noise variance is given by  $R(s)$ . We have found empirically that the choice  $R(s) = \max(\|C(s)\|^2, 10)$  worked well in all cases. This choice effectively penalizes large spatial gradients, which are points at which the brightness constraint equation is likely to have large errors [123] (due, for example, to noise, aliasing or occlusion).

Finally, recall that in the smoothness constraint formulation,  $L^T L$  was not invertible because of the implicit assumption of infinite prior variance on the DC value of the optical flow field. In our multiscale regularization context, this would correspond to setting  $P_0$  equal to infinity in (1.17). This can be done without difficulty in the estimation algorithms described next, but we have found that it is generally sufficient

simply to choose  $P_0$  to be a large multiple of the identity. In particular, the parameter  $p$  in the prior covariance was set to  $p = 100$ .

## 2.3 Experimental Results

We compare our approach computationally and visually to the Gauss-Seidel (GS) and successive over-relaxation (SOR) algorithms, which can be used to compute the solution of the smoothness constraint formulation given by (2.9) or (2.13) (see, for example, [62, 75, 107, 116, 117, 127] and Appendix B.2). In our experiments, we have found that SOR typically provides a factor of 10 to 100 performance improvement of Gauss-Seidel, and hence is computationally equal to or better than multigrid approaches [134, 49]. The parameter  $R$  in the Horn and Schunck formulation (2.4) was chosen in to yield good visual and quantitative results. In particular,  $R$  was set to 100 in the first example below, and 2500 in the subsequent examples. Several possibilities for choosing this parameter based on the image data have been proposed in the literature [12, 99], although there is no universally agreed upon method; our choice is comparable to those in [25, 5, 58].

Straightforward analysis shows (see Appendix B.2) that the GS and SOR algorithms require 14 and 18 floating point operations (flops) per pixel per iteration respectively. The number of iterations required for convergence of the iterative algorithms grows with image size [75]. For reasonable size images (say,  $512 \times 512$ ), SOR may require on the order of hundreds of iterations to converge, so that the total computation per pixel can be on the order of  $10^3$  to  $10^4$  flops. On the other hand, the MR algorithm requires 76 flops per pixel (see Appendix B.3). Recall further that *the MR algorithm is not iterative*. Thus, as we will now see, the computational gain associated with the MR algorithm can be on the order of one to two orders of magnitude for problems of this size and substantially greater for problems defined over much larger spatial regions.



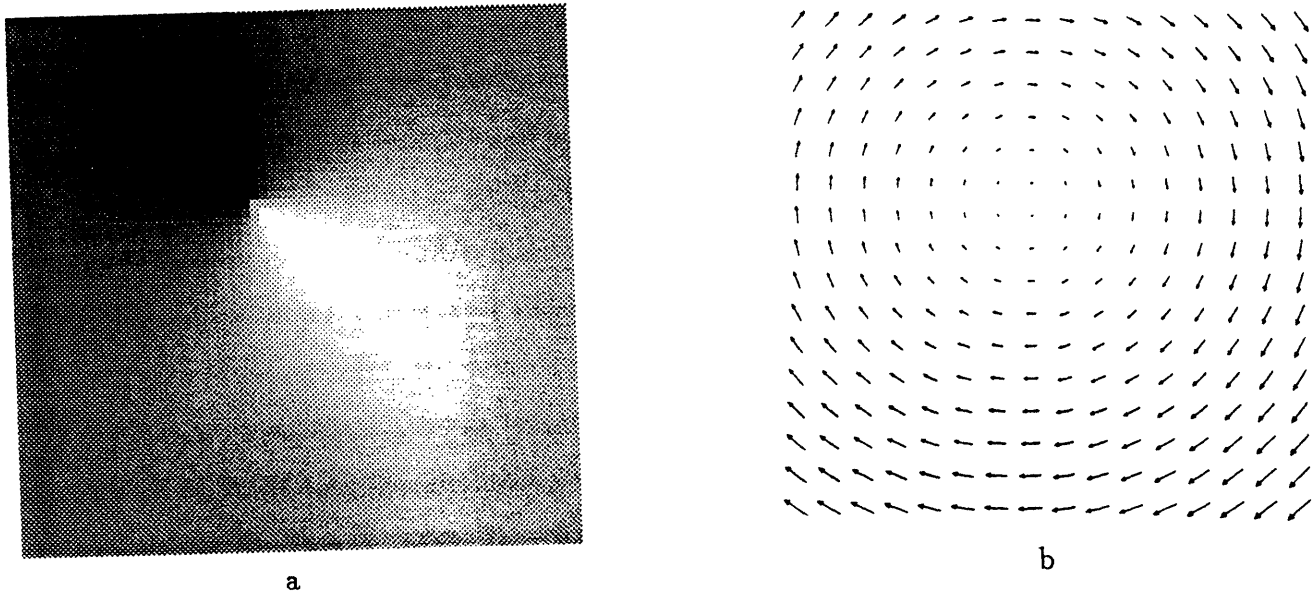


Figure 2-2: (a) First frame of the "rotation" sequence and (b) Rotation sequence true optical flow.

### 2.3.1 Rotation Sequence

We begin with a comparatively small synthetic example of rotational motion in order to illustrate the basic features of our approach. Specifically, this first example is a synthetic sequence of Gaussian images modulated by a spatial sinewave with the first frame brightness pattern given by:

$$E(z_1, z_2, t_1) = \sin(\text{atan}(z_1 - 23, z_2 - 28)) \exp\left(-\frac{1}{2} z' Z^{-1} z\right) \quad (2.23)$$

$$z = \begin{bmatrix} z_1 - 23 \\ z_2 - 28 \end{bmatrix} \quad (2.24)$$

$$Z = \begin{bmatrix} 1000 & 0 \\ 0 & 500 \end{bmatrix} \quad (2.25)$$

where  $\text{atan}(z_1, z_2)$  is a  $2\pi$  arctangent ( $\text{atan}(0,1) = 0$ ,  $\text{atan}(1,0) = -\pi$ ),  $h = 1$  and  $M = 6$  (i.e. the image lattice is  $64 \times 64$ , cf. the discussion about discretization at the beginning of Section 2.2). The second frame is equal to the first, rotated by 1 degree about pixel (23,28). The first frame and actual optical flow are illustrated in Figure 2-2. The rms value of this flow field is 0.49.

The first point we wish to examine is the visual appearance of the estimates produced. Figure 2-3 shows four different estimates of the optical flow. The first of these (a) is the SC estimate produced using the original smoothness constraint formulation and performing 50 iterations of the SOR algorithm<sup>3</sup>; (b) is the finest scale of the MR estimates produced by the MR algorithm with the parameters set as  $b = \mu = 1$ ; (c), which we refer to as MR-PF, is a post-filtered version of the MR estimates in (b) to be described; and (d), which we refer to as MR-SOR, is the estimate produced by performing 5 iterations of the SOR algorithm used in (a) but using the MR estimates in (b) as an initial condition. All four estimates clearly display the rotational nature of the true flow with quality that is roughly comparable. In particular, while rms error is not necessarily an appropriate measure of absolute estimate quality, it is of value in assessing the *relative* quality of these four methods, and for this example the rms errors for the estimates in Figure 2-3 are:

(SC)	0.24
(MR)	0.22
(MR-PF)	0.22
(MR-SOR)	0.20

which indicates that the MR method and its variations in (c) and (d) yield estimates of quantitative accuracy comparable to the SC-based method.

Despite this fact, the MR estimate in (b) has visual characteristics that may be somewhat distracting to the viewer: namely, the apparent blockiness of the estimates. As the rms errors indicate, and as we argue further in a moment, this visual artifact is not quantitatively significant. However, its nature and the reason for its presence motivate the computationally simple post-processing procedures illustrated in parts (c) and (d) of Figure 2-3. The first of these is motivated by the interpretation of our MR algorithm in terms of wavelet transforms and multiresolution analysis [21, 86]. Specifically, as discussed in Chapter 1, the values of a multiscale process at a given scale can be thought of as the so-called “scaling coefficients” [86] of particular basis

---

<sup>3</sup>In this and subsequent examples, the iterative algorithms computing the solution of (2.4) were initialized with zero.

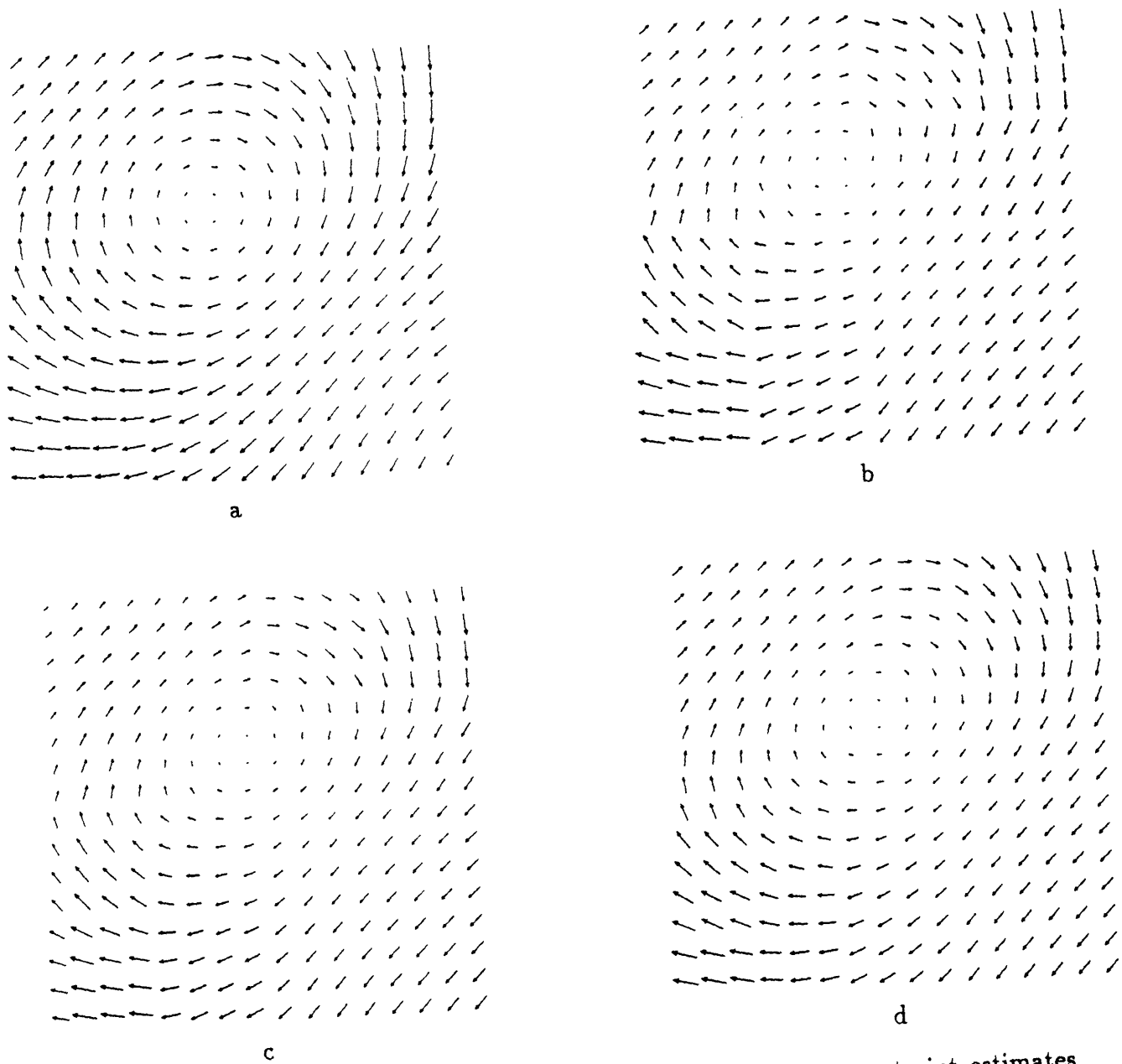


Figure 2-3: Rotation sequence flow estimates. (a) Smoothness constraint estimates computed using 50 iterations of SOR, (b) Multiscale Regularization (MR) estimates, (c) Post-filtered MR estimates and (d) Estimates produced by using MR estimates as initial condition for SOR algorithm.

functions used in the approximation at that scale. In that sense, the flow field estimate in (b) corresponds to the Haar approximation in which the basis functions are piecewise constant over squares of size corresponding to the scale being represented. The blockiness in (b) is thus due to the “staircase” nature of the Haar approximation. On the other hand, there are far smoother choices for basis functions and multiresolution approximations, each of which corresponds in essence to convolving the 2-D array of quadtree estimates at the finest scale with particular FIR filters. The MR-PF estimates in Figure 2-3(c) corresponds to using the FIR filter given by (2.22) together with the MR estimate in (b).

The estimate in (d) is motivated by the observation that the visual artifacts in the estimate (b) are local and high-frequency in nature. Indeed, it is *precisely* these high frequency artifacts that are quickly and easily removed by SOR or GS algorithms computing the smoothness constraint solution. This is clearly demonstrated in the MR-SOR estimates in (d) in which only 5 SOR iterations have been used to post-process (b).

Let us now turn to the question of computational complexity. Figure 2-4 illustrates the rms error in the flow estimates as a function of iteration for the SOR and GS algorithms. The rms error in the MR flow estimate of Figure 2-3(b) as well as those of MR-PF and MR-SOR in (c) and (d) are also indicated in the figure. The procedures used to generate the MR, MR-PF and MR-SOR estimates are *not* iterative and thus the associated rms errors are shown simply as straight lines. Note first that, as expected, the SOR algorithm is significantly faster than the GS algorithm (they will converge to the same result since they are solving the same partial differential equation). However, the SOR algorithm itself has a substantial computational burden. For example, while the SOR algorithm has not converged after 50 iterations, the estimates in Figure 2-3(a) are not bad, but even at this point and even for this small example, SOR requires far more computation than the MR based estimate. In particular, as we indicated previously, the computational load of the MR algorithm equals 4.2 SOR iterations, while producing the MR-PF and MR-SOR esti-

mates requires computation equivalent to 5.6 and 9.2 SOR iterations, respectively<sup>4</sup>. Thus, for this small example, the algorithms corresponding to Figures 2-3(b) – (d) offer computational savings over SOR of factors of  $50/4.2 = 11.9$ ,  $50/5.6 = 8.9$  and  $50/9.2 = 5.4$  respectively. As an aside, note that these results also suggest that if one insists upon solving the partial differential equation corresponding to the SC formulation, then using the MR estimate as an initial condition is a computationally attractive way in which to do this. Specifically, Figure 2-5 illustrates the rms difference between the smoothness constraint solution<sup>5</sup> and the intermediate values of the GS, SOR and MR-initialized SOR estimates as a function of iteration. The error plot for the MR-initialized SOR algorithm begins at 4.2 iterations to take into account the initial computation associated with the MR algorithm. The figure demonstrates that the MR-initialized SOR approach provides a substantial reduction in computational burden even for this small size problem. This in fact suggests that MR algorithms may be of more general use in the efficient solution of partial differential equations in other applications as well.

As we have emphasized, the MR algorithm has other attractive features beyond its computational efficiency, including the fact that it directly provides estimates at multiple resolutions. Figure 2-6 depicts these estimates at several scales (where the finest scale estimates are in Figure 2-3(b)). These coarser estimates also obviously capture the rotational motion and may, in some cases, be preferable representations of perceived motion because of their comparative parsimony compared to Figure 2-3(b). Indeed in many applications one is interested in fairly aggregate measures of motion which these estimates provide directly. Furthermore, as we describe next, the MR algorithm in fact directly provides a precise way in which to determine the optimal resolution for characterizing optical flow in different regions of the image, the basis of which is the multiscale covariance information computed by the MR algorithm.

---

<sup>4</sup>With respect to the MR-PF estimates (c), straightforward convolution of the two components of the optical flow in (b) with a separable  $7 \times 7$  filter requires 26 flops per pixel (equivalent to 1.4 SOR iterations) and could, of course, be reduced further with FFT algorithms.

<sup>5</sup>The smoothness constraint solution is approximated as the SOR algorithm optical flow estimates after 500 iterations.

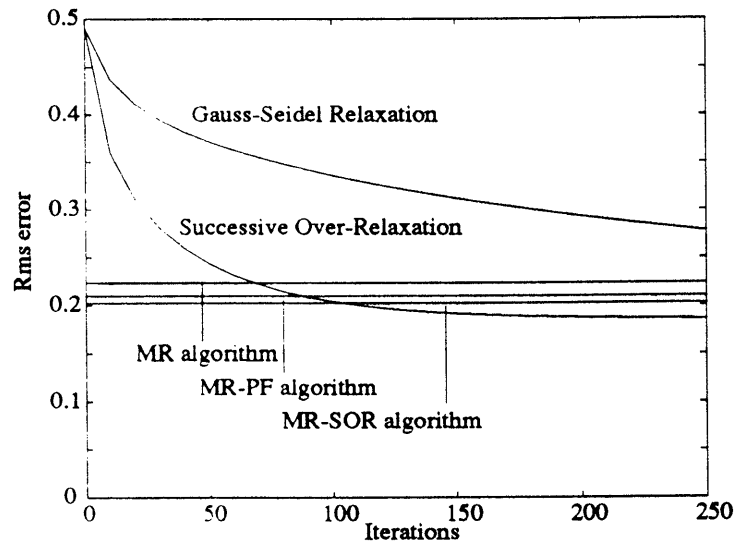


Figure 2-4: Rms Error Comparison of MR, MR-PF, MR-SOR, SOR and Gauss-Seidel (GS) algorithm flow estimates for the rotation sequence.

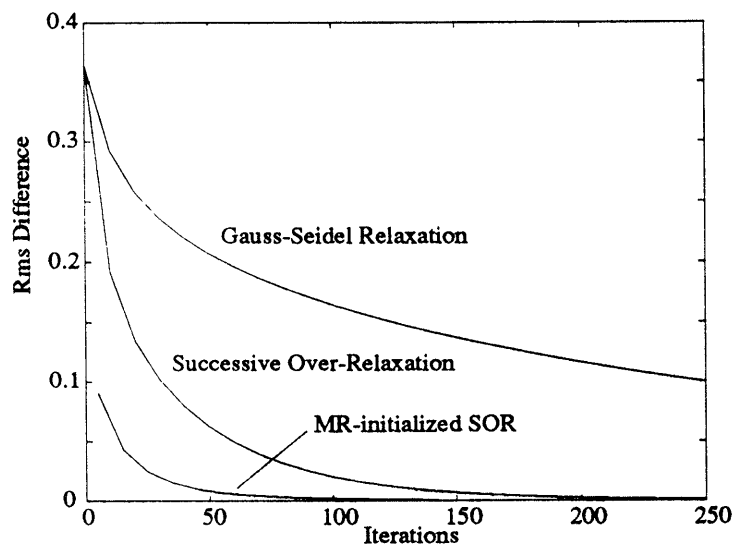


Figure 2-5: Rms difference comparison illustrates how the MR-initialized SOR, SOR and GS algorithms converge to the smoothness constraint solution for the Rotation sequence. The plots show the rms difference between the smoothness constraint solution and the estimates as a function of iteration. All will eventually converge, but the MR-initialized SOR algorithm converges much faster than SOR or GS.

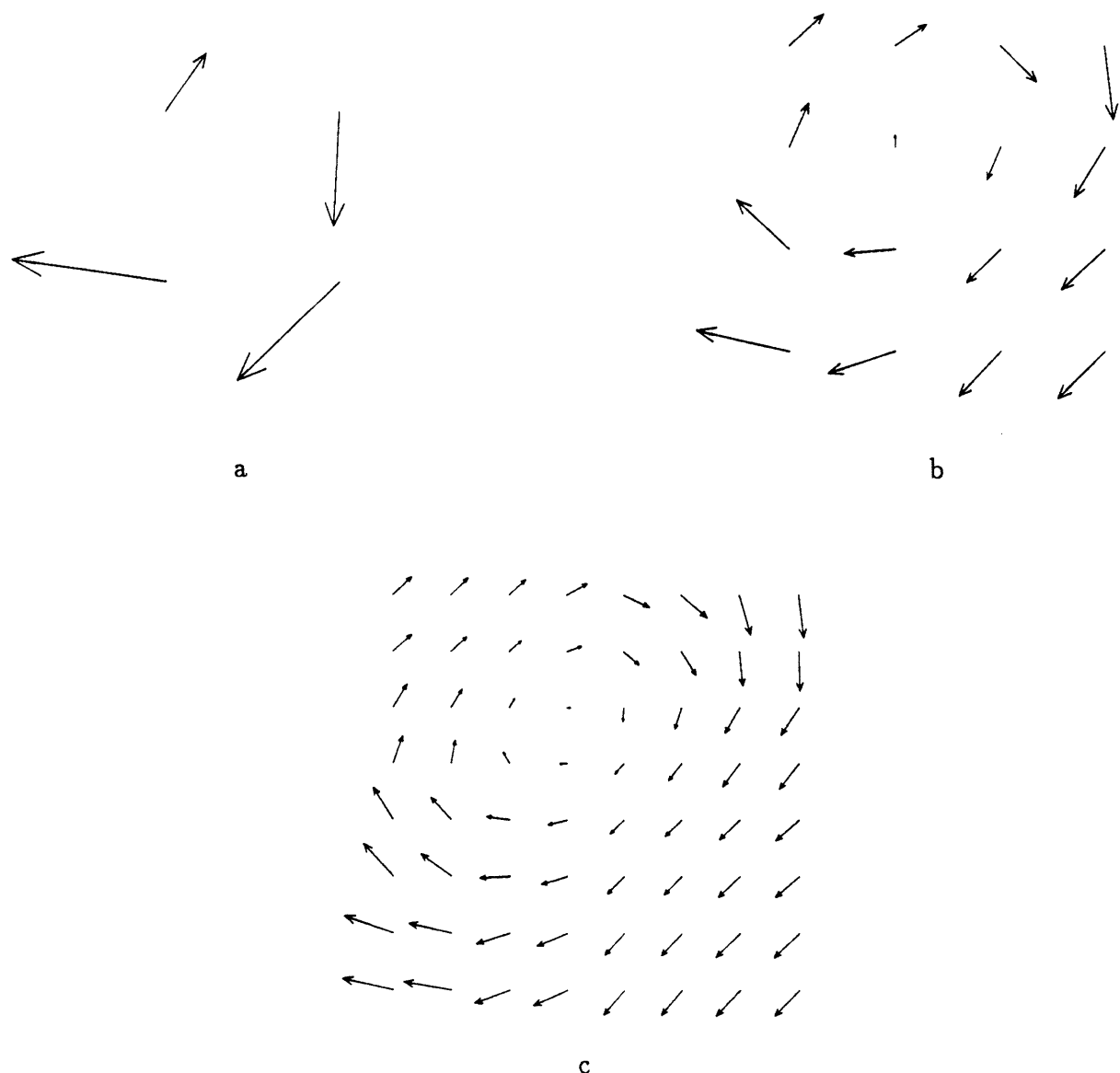


Figure 2-6: Multiscale Regularization flow estimates at the (a) first, (b) second and (c) third scales.

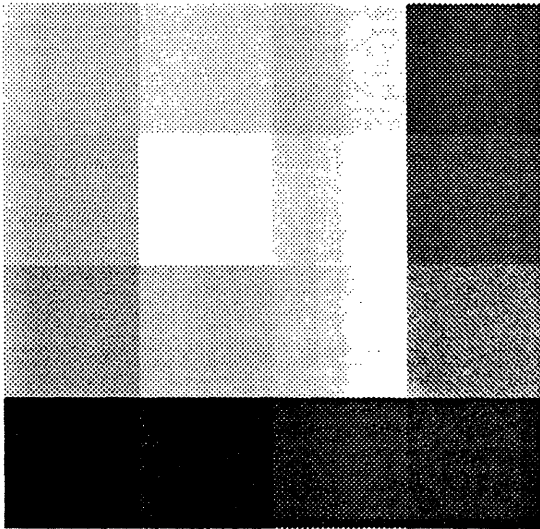
Figure 2-7 illustrates the trace of the  $2 \times 2$  estimation error covariance in (1.44) at each point in the quadtree at different scales. Bright areas correspond to regions of lower covariance (higher confidence). Note that around the border of the image, where the Gaussian has tapered off and the gradients are relatively small, the error covariance is relatively large, as compared to the region around the point of rotation. One use of this covariance information is to provide information that may be useful

to higher level vision algorithms which use the optical flow field in conjunction with information from other sources, and need to combine this information in a rational way. Moreover, as we have suggested, this information can also be used in the context of addressing the problem of resolution vs. accuracy in the estimates. The idea is that we would expect to estimate rather well the coarse resolution features in the optical flow field and that finer resolution features could be estimated with decreasing fidelity depending on the quality and characteristics of the available data (e.g. on the presence or absence of fine scale image intensity fluctuations). Thus, what we would like is a rational procedure for determining the estimate resolution supported by the data.

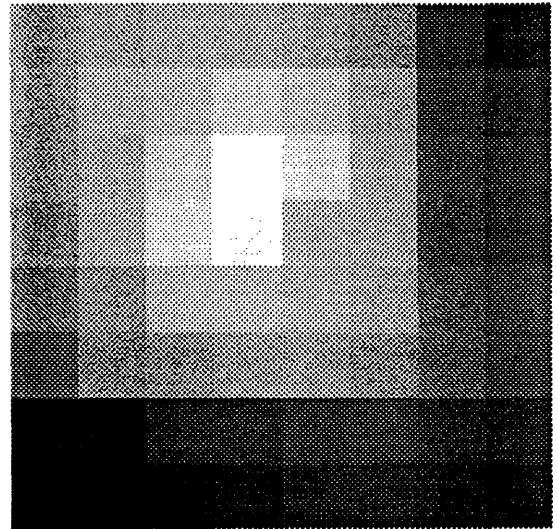
There are several ways in which the flow estimate covariance information can be used to approach this problem. One possibility, which has a precise statistical interpretation, is as follows. To each node at the finest scale, we can trace a path up to the root node, where nodes in the path correspond to the parent, grandparent, great-grandparent, etc. of the node at the finest level. The optical flow estimates at each of these resolutions can be thought of as successively coarser representations of the optical flow estimate at the finest scale. Associated with that same path is a sequence of smoothing error covariance matrices computed via (1.44). At each pixel location we can choose the optimal resolution at which to represent the field by choosing the scale at which this error covariance is minimum. In Figure 2-8 the scale of the minimum of the trace of the smoothed error covariance along this path is plotted for each lattice site. Note that in regions near the border, where the Gaussian has tapered off and not much gradient information is available, a lower resolution representation for the flow field is given. On the other hand, near the point of rotation, where there is gradient information, the resolution is at a higher (i.e. finer) level. It is interesting to note that the areas in which the finest level MR estimate of Figure 2-3(b) has the most visually obvious blocky behavior are also areas in which one has no business estimating optical flow at such a fine scale to begin with. Said another way, one interpretation of Figure 2-8 is that *any* estimate of optical flow at such a fine scale in such regions is a visual artifact!

Finally, let us briefly comment on the choice of the parameters  $b$  and  $\mu$  in the MR

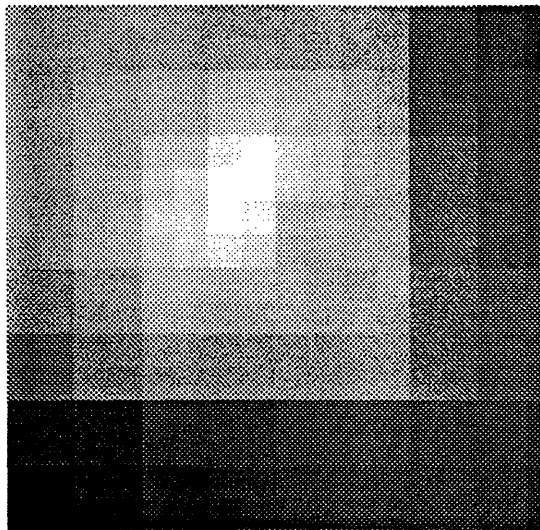




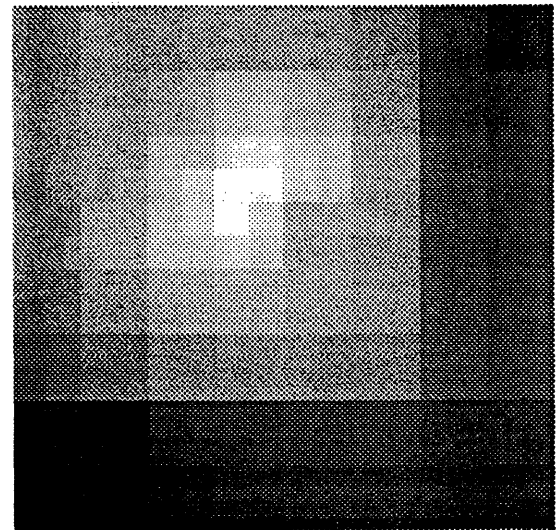
a



b



c



d

Figure 2-7: Multiscale Regularization error covariance at the (a) third, (b) fourth, (c) fifth and (d) finest scales.

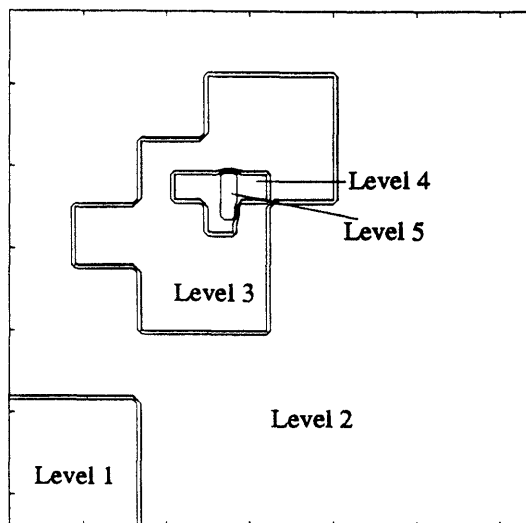


Figure 2-8: Map showing the optimal resolution for optical flow reconstruction for the rotation image sequence optical flow. At points near the focus of rotation the flow is represented at fine scales, while at points near the edge of the image (where little gradient information is available) the optical flow is represented at a coarser level of the quadtree.

algorithm. In particular, we have found through experimentation that the rms error in the estimates and their qualitative appearance is relatively insensitive to  $b$  and  $\mu$ . Figure 2-9 depicts the rms errors in the MR flow estimates for the rotation example as a function of  $b$  and  $\mu$ , displaying characteristically flat behavior over a very large range of values.

### 2.3.2 Yosemite Sequence

The second example is a synthetic image sequence which simulates the view from a plane flying through the Yosemite Valley<sup>6</sup>. The first image in the sequence and the

<sup>6</sup>This sequence was synthesized by Lyn Quam of SRI International. The original sequence is  $252 \times 312$ . As discussed in Appendix B.1, it is straightforward to adapt our approach to trees other than regular quadtrees, i.e. to trees with varying numbers of branches. However, for simplicity, in

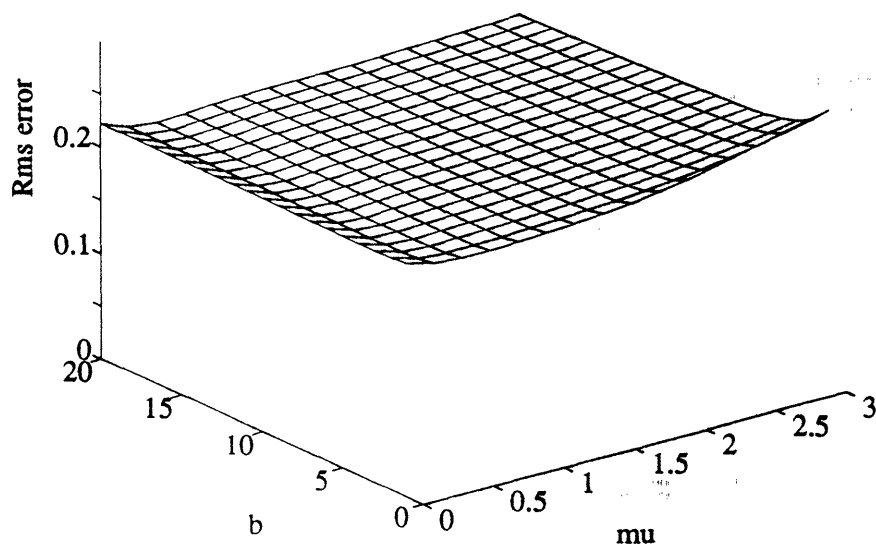


Figure 2-9: Multiscale Regularization rms error sensitivity to the parameters  $b$  and  $\mu$  (rotation sequence).

corresponding optical flow are shown in Figure 2-10. The rms value of the flow field is 1.86.

Figure 2-11 illustrates four estimates of the optical flow corresponding to (a) the SC formulation after 100 iterations of the SOR algorithm, (b) the finest scale of estimates produced by the MR algorithm with parameters  $b = \mu = 1$ , (c) the MR-PF estimates derived as described previously and (d) the MR-SOR estimates produced by post-processing the MR estimates with 10 iterations of SOR. The estimates are qualitatively similar, each indicating the fly-through nature of the sequence. The estimates are also quantitatively similar as indicated by the rms errors for the four estimates:

(SC)	0.76
(MR)	0.79
(MR-PF)	0.79
(MR-SOR)	0.78

---

these experiments we have coded our algorithms for quadtrees. For this example, then, we extracted a  $252 \times 256$  portion of the sequence (the left side) so that processing could be done on a quadtree with  $256 \times 256$  lattice sites at the finest level. The measurement matrix  $C(s)$  defined at the unneeded four rows of the quadtree structure was set to zero, reflecting the fact that we have no information about the (non-existent) optical flow field in that region.

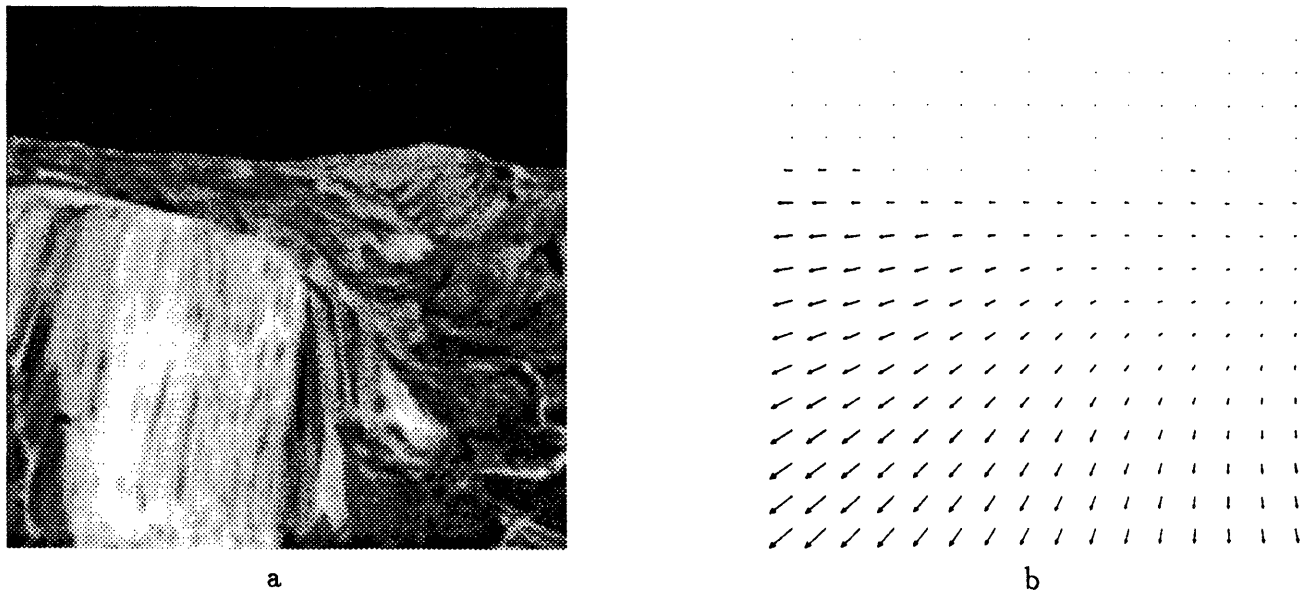


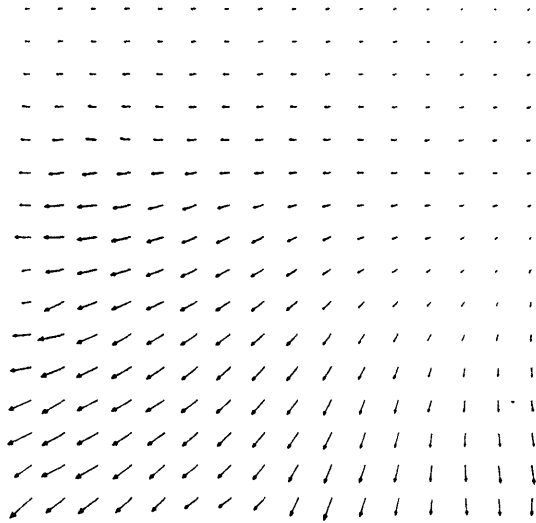
Figure 2-10: (a) First frame of Yosemite sequence and (b) Yosemite sequence true optical flow.

The rms errors as a function of iteration are shown in Figure 2-12. Note that the SC estimates (a) have actually not yet converged after 100 iterations and that when they do, the rms error of the SC estimate is slightly higher than those for the various approaches based on the MR algorithm.

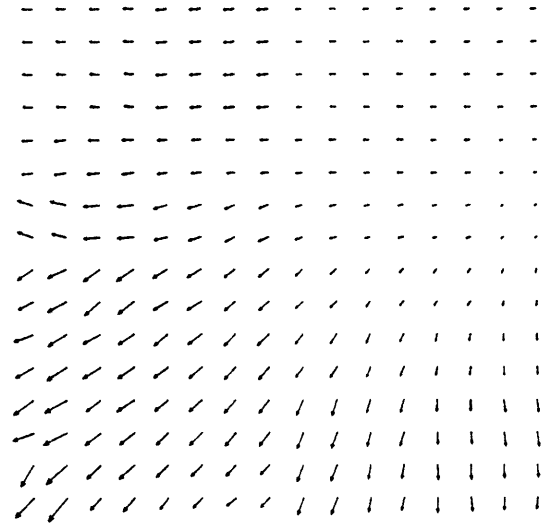
Again, there is some blockiness in the MR optical flow estimates, and, as seen in Figures 2-11(c) and (d), some of this effect can be eliminated by post-processing the estimates with an FIR filter as in the previous example. There is still some blockiness apparent, but comparison with (a) shows that this is also apparent in the SC solution. Hence, the residual blockiness in the smoothed estimates is not due to the quadtree structure, but rather to the nature of the image sequence data itself.

An examination of computational complexity again shows the gains achievable using MR-based methods. The SC flow estimates shown in Figure 2-11(a) required 100 SOR iterations in this example, representing a factor of  $100/4.2 = 23.8$  more computation than the MR estimates. Likewise, the MR-PF and MR-SOR (c) and (d) represent factors of  $100/7.7 = 13$  and  $100/14.2 = 7.0$  computational improvement.

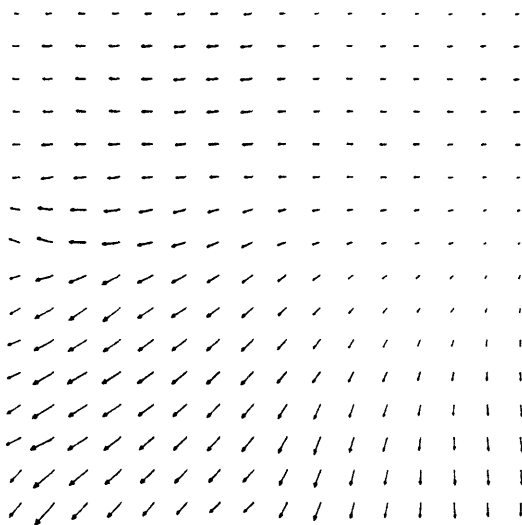
Furthermore, as before one would expect to be able to quickly obtain the SC solution by using the MR solution as an initial condition. Figure 2-13 illustrates



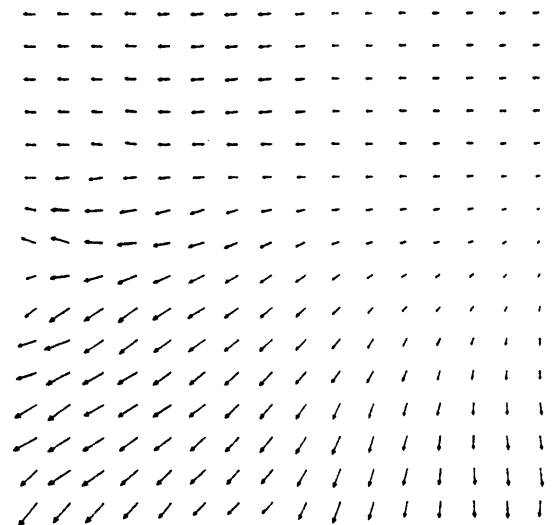
a



b



c



d

Figure 2-11: Yosemite sequence flow estimates. (a) Smoothness constraint estimates computed using 100 iterations of SOR, (b) Multiscale Regularization (MR) estimates, (c) Post-filtered MR estimates and (d) Estimates produced by using MR estimates as initial condition for SOR algorithm.

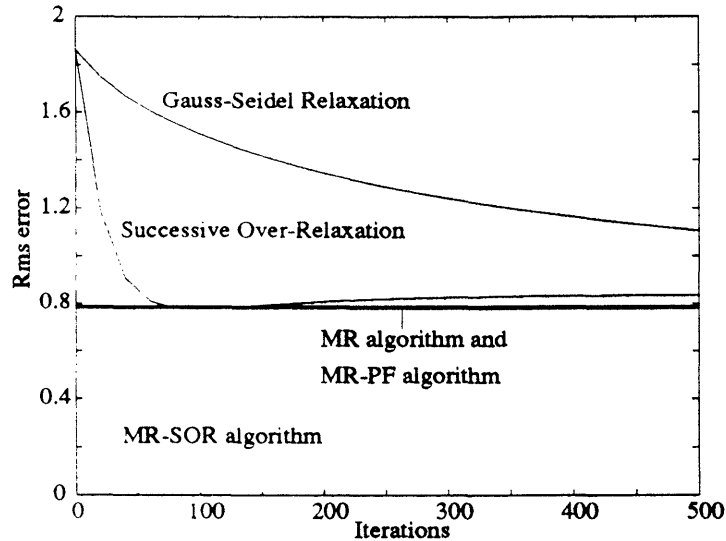


Figure 2-12: Rms Error Comparison of MR, MR-PF, MR-SOR, SOR and Gauss-Seidel (GS) algorithm flow estimates for the yosemite sequence.

how the GS, SOR and MR-initialized SOR algorithms converge to the smoothness constraint solution. Note that visually, there is almost no difference between the MR-initialized SOR estimates Figure 2-11(d) and the SC estimates shown in Figure 2-11(a). Indeed, the rms difference between the MR estimates and the smoothness constraint solution is 0.178, while the rms difference between the estimates in Figure 2-11(a) and the smoothness constraint solution is 0.181. More generally, Figure 2-13 shows that for any given number of iterations, the MR-initialized SOR estimates are substantially closer to the final solution than the GS or SOR estimates.

Estimates of the optical flow at several scales computed via the MR algorithm are shown in Figure 2-14 and multiscale error covariance images, again, corresponding to the traces of the smoothing error covariance matrices at individual lattice sites, are shown in Figure 2-15. The coarser versions of the flow are intuitively reasonable given the estimates at the finest level and, as expected, the covariance images are relatively dark (high covariance) in the top portion of the image where there is no gradient information available.

Figure 2-16 depicts a map of the optimum resolution for flow estimation at each pixel location computed as the minimum of the trace of the smoothed error covariance matrix along paths from nodes at the finest level to the root node. We see,

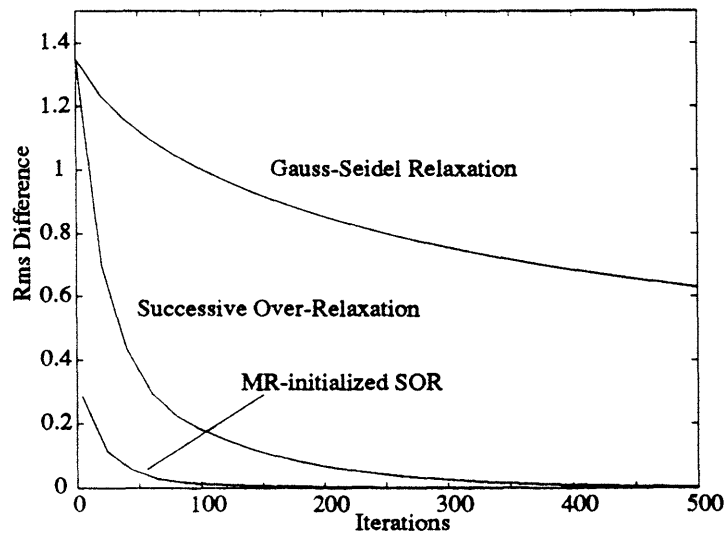


Figure 2-13: Rms Difference Comparison illustrates how the MR-initialized SOR, SOR and GS algorithms converge to the smoothness constraint solution (Yosemite sequence).

not surprisingly, that the level of resolution chosen for the region with no intensity information is quite low. In addition, the resolution along the face of the mountain in the foreground is slightly reduced due to the relative lack of gradient information in the direction of the striations.

Finally, Figure 2-17 illustrates the variations in the rms error in the optical flow estimates to variations in the parameters  $b$  and  $\mu$ . The figure shows that the estimates are relatively insensitive to the parameter  $b$ , and are also insensitive to  $\mu$  for values ranging from slightly less than 1 upward. The degradation in performance as  $\mu$  decreases toward zero is not uncommon or unexpected. In particular, as discussed in [30, 27, 29, 31, 150, 151, 152] decreasing  $\mu$  leads to significant decreases in spatial correlation in the model and to far noisier sample paths. Thus, the estimates for small values of  $\mu$  correspond to imposing virtually *no* smoothness constraint, resulting in estimated fields with noise-like characteristics. On the other hand, choosing any value of  $\mu \geq 1$  yields results of comparable quality to each other and to the SC solution.

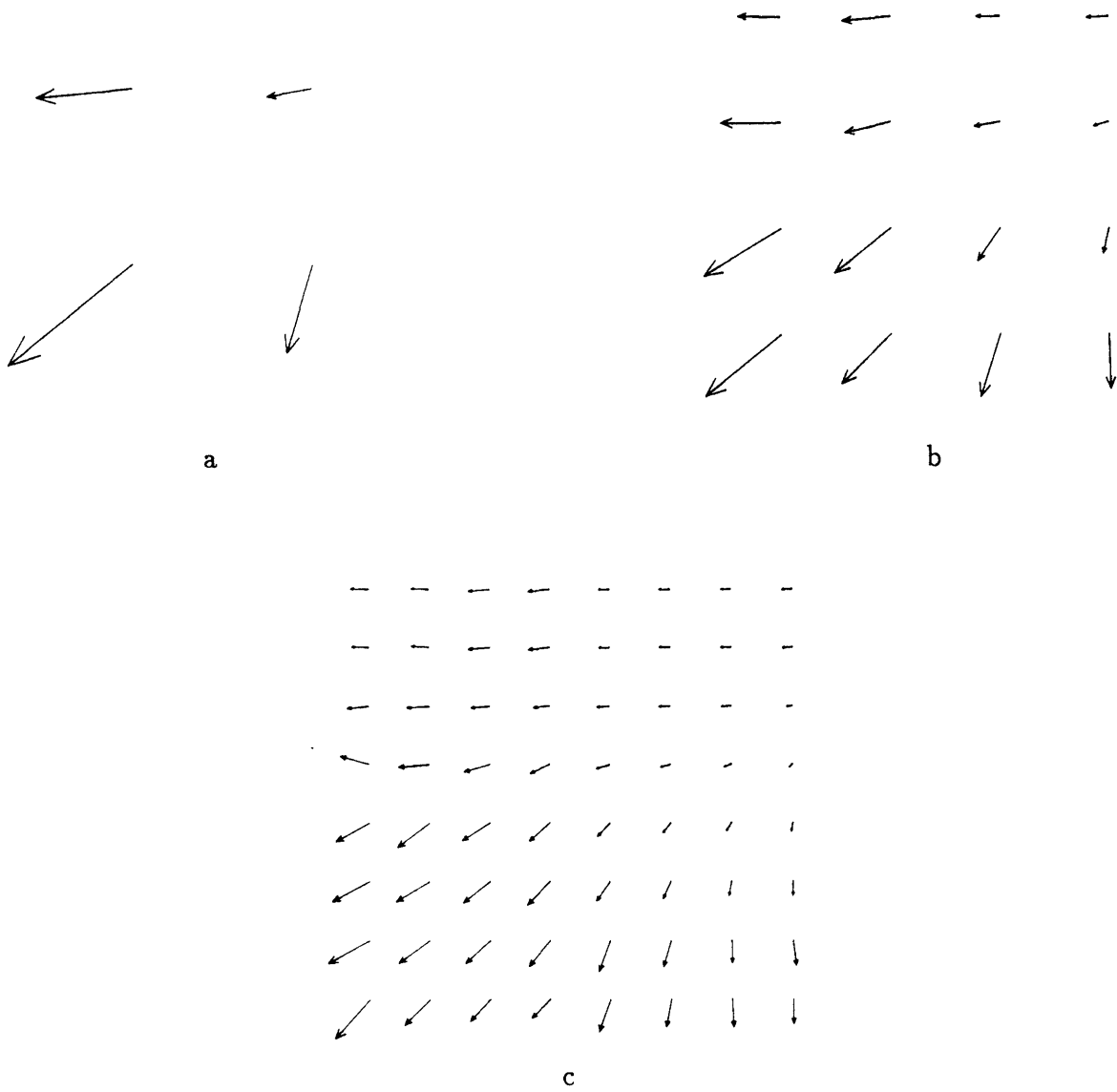


Figure 2-14: Multiscale Regularization flow estimates at the (a) first, (b) second and (c) third scales.

### 2.3.3 Moving Vehicle Sequence

The third example is based on a real<sup>7</sup> image sequence which depicts the view from a car driving down a road. The first image in the sequence is illustrated in Figure 2-18 and Figure 2-19 illustrates four estimates of the optical flow corresponding to

<sup>7</sup>The sequence is courtesy of Saab-Scania.





a



b



c



d

Figure 2-15: Multiscale Regularization error covariance at the (a) second, (b) fourth, (c) sixth and (d) finest scales.

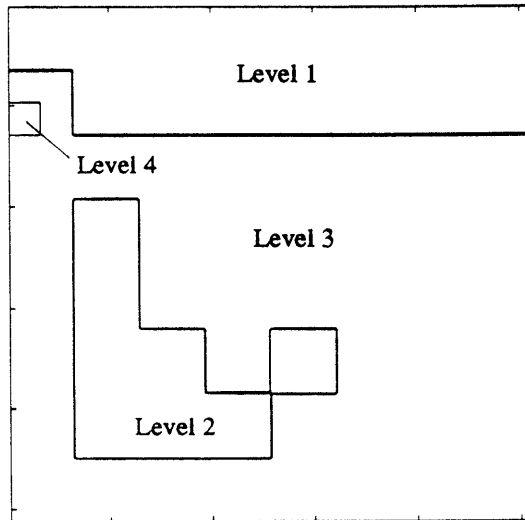


Figure 2-16: Map depicting the optimal resolution for representing the optical flow field as a function of lattice site. Note that the optical flow field is represented at a coarser level in the quadtree in regions where there is no gradient information (at the top). It is also represented at a coarser level along the face of the mountain, where there is little gradient information parallel to the striations.

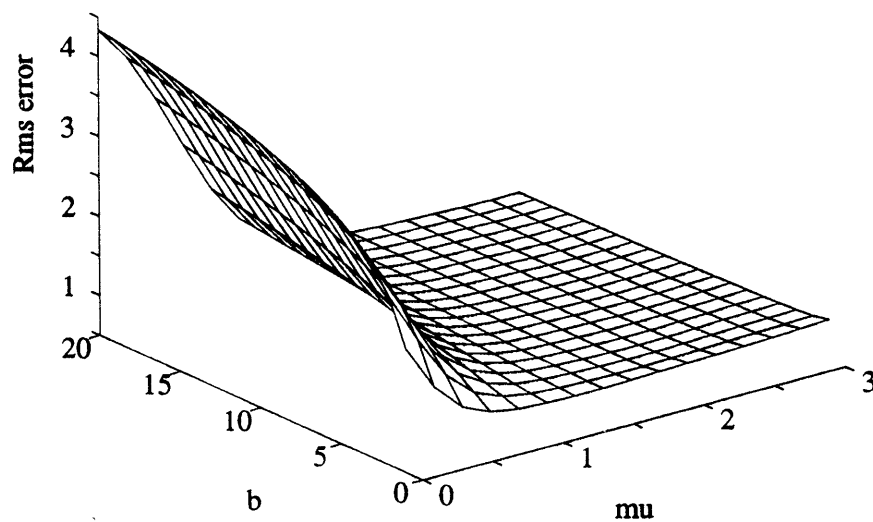


Figure 2-17: Multiscale Regularization rms error sensitivity to the parameters  $b$  and  $\mu$  (Yosemite sequence).

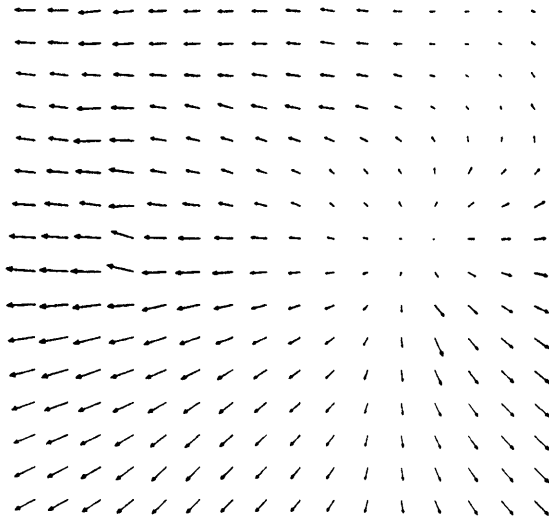


Figure 2-18: First frame of Moving vehicle sequence.

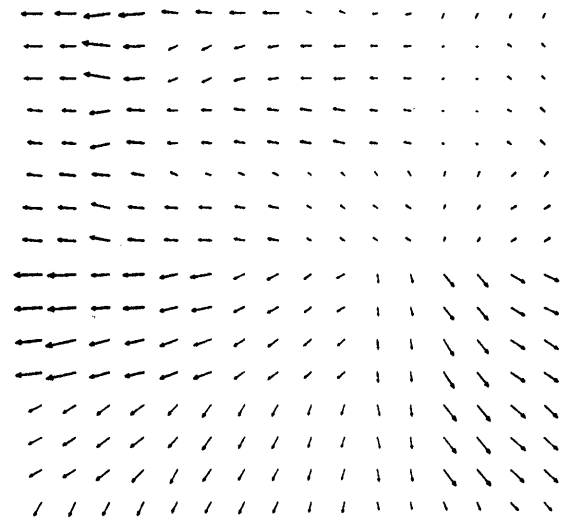
(a) the SC formulation and 200 iterations of the SOR algorithm, (b) the finest scale of estimates produced by the MR algorithm with parameters  $b = \mu = 1$ , (c) the MR-PF estimate and (d) the MR-SOR estimate produced by post-processing the MR estimates (b) with 30 iterations of SOR.

Since the true optical flow is not available (as it was in the previous simulated examples), an alternate performance metric is needed. In particular, we will use a reconstruction error metric, which is often used in contexts in which one is interested in using optical flow for motion-compensated coding. This metric measures the mean square difference between the current image in a sequence and an estimate of it based on the computed optical flow, the previous image, and a bilinear interpolation scheme [100]. The optical flow used is that associated with the current image. Essentially, one estimates the brightness at any given point by using the optical flow to project that point back to the previous image. In general, that point will not be on the image plane, and the bilinear interpolation is required.

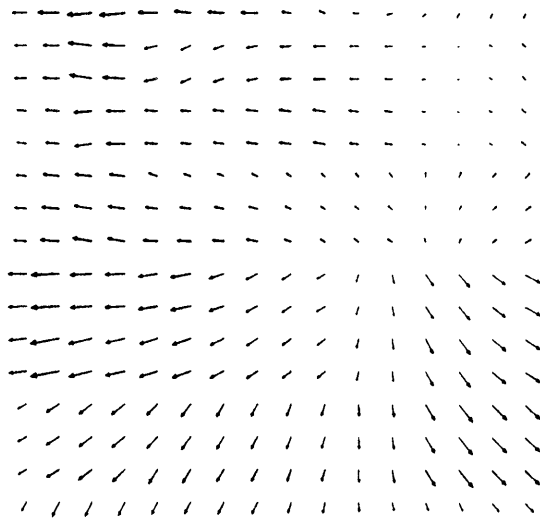
Figure 2-20 provides a comparison of reconstruction error performance for the approaches as a function of iteration (where once again the results for the non-iterative MR, MR-PF and MR-SOR approaches are depicted as horizontal lines). In this example, the SC solution was slightly better than the MR and MR-PF methods,



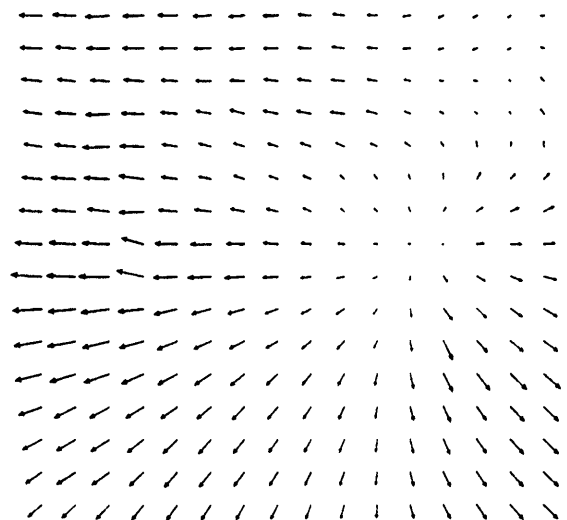
a



b



c



d

Figure 2-19: Moving vehicle sequence flow estimates. (a) Smoothness constraint estimates computed using 300 iterations of SOR, (b) Multiscale Regularization (MR) estimates, (c) Post-filtered MR estimates and (d) Estimates produced by using MR estimates as initial condition for SOR algorithm.

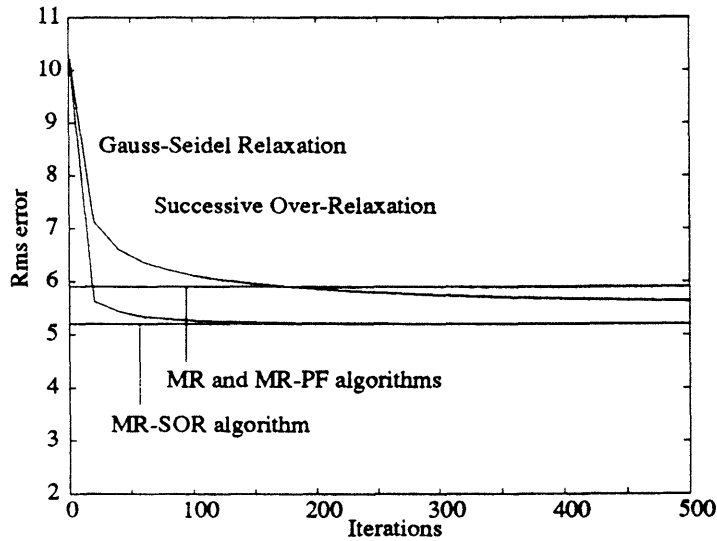


Figure 2-20: Rms Error Comparison of MR, SOR and Gauss-Seidel (GS) algorithm flow estimates for the Moving vehicle sequence.

achieving a slightly greater rms error reduction from the value obtained without motion compensation (i.e. straightforward frame difference given by the zero-iteration starting point for SOR). However, this slight increase in performance is achieved at the cost of significantly greater computation. In particular, the computational gains are  $200/4.2 = 47.6$ ,  $200/6.98 = 28.7$  for the MR-PF and MR-SOR approaches, respectively. Furthermore, as is also illustrated in Figure 2-20, the modest performance gain of SC over MR can be recouped with far less computation using the MR-SOR procedure which has a factor of  $200/34.2 = 5.8$  computational speedup. Indeed, as Figure 2-21 shows, the MR-SOR solution of Figure 2-19(d) is *closer* to the SC solution than the result in Figure 2-19(a), which required 200 iterations of SOR to obtain.

As in the previous examples, multiresolution flow estimates and error covariance information is available at all levels of the quadtree, and an image of the error covariance information at the finest level lattice points is shown in Figure 2-22(a). Note in this case that the error covariance is relatively high (dark regions in the image) along the road where the image gradient is relatively low. Also, Figure 2-22(b) depicts the optimal resolution at which to recover the optical flow field computed using this error covariance information.

Finally, the sensitivity of the optical flow estimates in this example to parameter

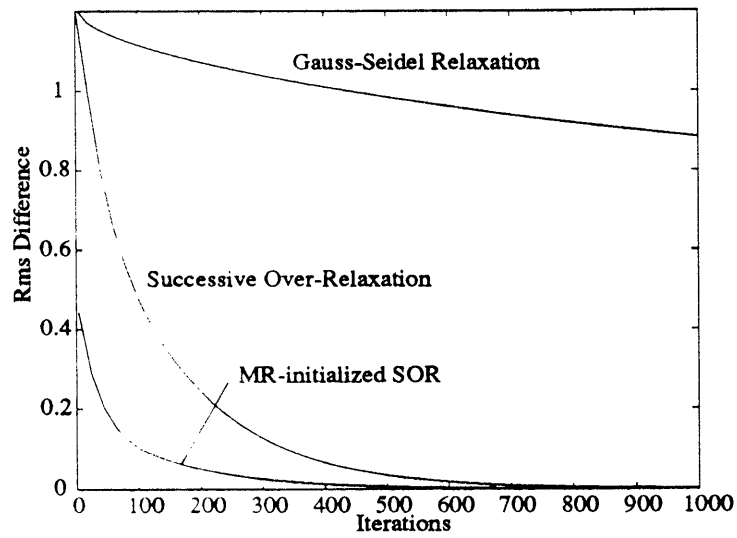
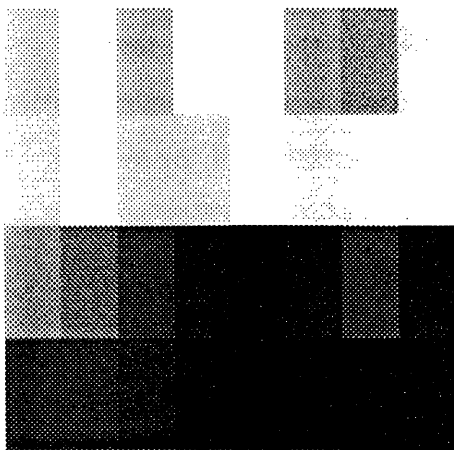
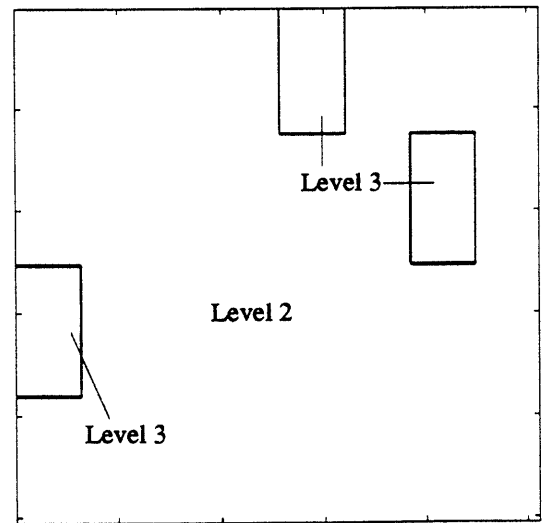


Figure 2-21: Rms Difference Comparison illustrates how the MR initialized SOR, SOR and GS algorithms converge to the smoothness constraint solution (Moving vehicle sequence).



a



b

Figure 2-22: (a) Multiscale Regularization error covariance at the finest scale and (b) Map illustrating the optimal representation resolution for the Moving vehicle sequence optical flow estimates.

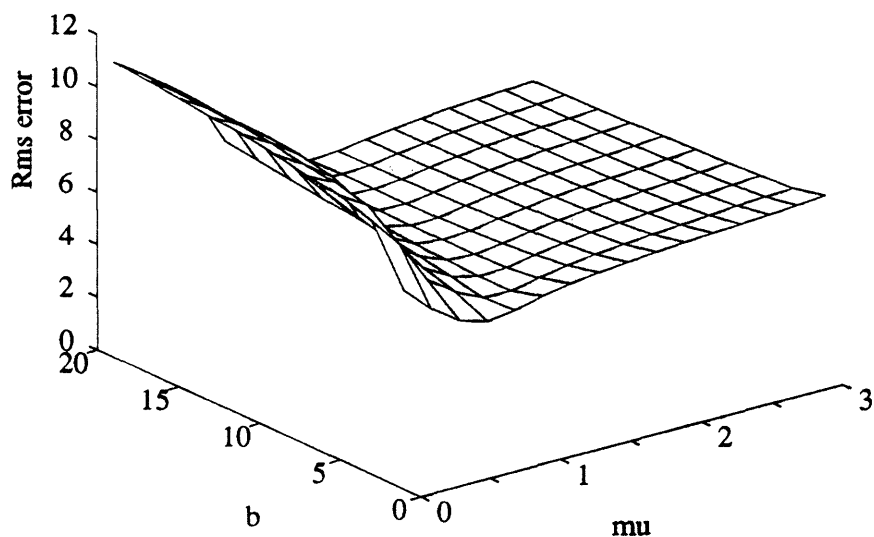


Figure 2-23: Multiscale Regularization rms error sensitivity to the parameters  $b$  and  $\mu$  (Moving vehicle sequence).

choice is shown in Figure 2-23. The figure shows that the reconstruction error is stable for  $\mu \geq 1$  as in the Yosemite example, and is insensitive to variations in  $b$  over a significant range of values.

### 2.3.4 Chopper Sequence

The first frame of the real “chopper” sequence<sup>8</sup> is shown in Figure 2-24. Figure 2-25 illustrates four estimates of the optical flow corresponding to (a) the SC formulation and 200 iterations of the SOR algorithm, (b) the finest scale of estimates produced by the MR algorithm with parameters  $b = \mu = 1$ , (c) the MR-PF estimate and (d) the MR-SOR estimate produced by post-processing the MR estimates (b) with 80 iterations of SOR.

As in the previous example, rms reconstruction error is the metric we use for comparison since the true flow is not known. Figure 2-26 provides a comparison of the reconstruction error performance of the approaches as a function of iteration. Note that in this example all four methods yield essentially identical rms performance,

<sup>8</sup>The  $480 \times 480$  image lattice was centered on the finest level of a 10 level ( $512 \times 512$  at the finest scale) quadtree. Again, as discussed in Appendix B.1, adapting our approach to deal directly with arbitrary size lattices is straightforward.



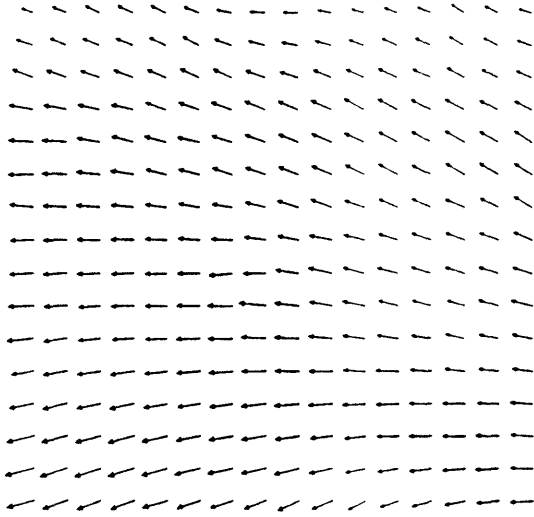
Figure 2-24: First frame of Chopper sequence.

but once again the MR-based algorithms have significant computational advantage. Computational gains for the MR, MR-PF, and MR-SOR approaches are  $200/4.2 = 47.6$ ,  $200/6.53 = 30.6$  and  $200/84.2 = 2.38$ .

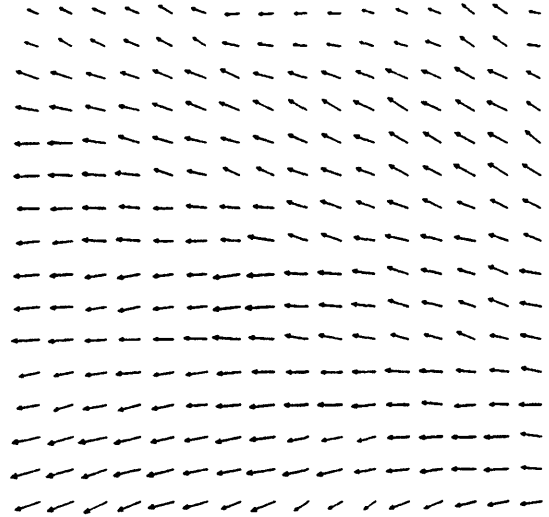
Also, as in the previous examples, the performance of the MR algorithm is stable over a wide range of values of the parameters  $b$  and  $\mu$ , as is illustrated in Figure 2-27. In addition, multiresolution estimates and error covariance information are, of course, available. For the sake of brevity, we illustrate only map of the optimum resolution information constructed from the multiscale error covariance information in Figure 2-28. Note in this case that the resolution level is relatively uniform over the image and is at a scale far coarser than the finest scale scale. That is, the image spatial intensity variations in this image sequence are not particularly strong so that fine resolution flow estimation can only be achieved with high levels of uncertainty.

On the other hand, there is an important fine-level velocity feature of some significance in this image sequence, namely a helicopter, located near the center of the image frame, which is moving relative to the background. While the local image contrast in the image is not sufficiently strong to allow very accurate estimation of what is in essence a discontinuity in the optical flow field, it is reasonable to expect that there would be *some* useful, quantitative information in the image sequence that

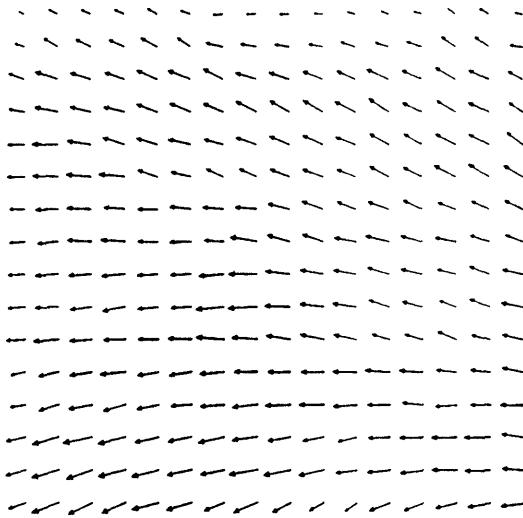




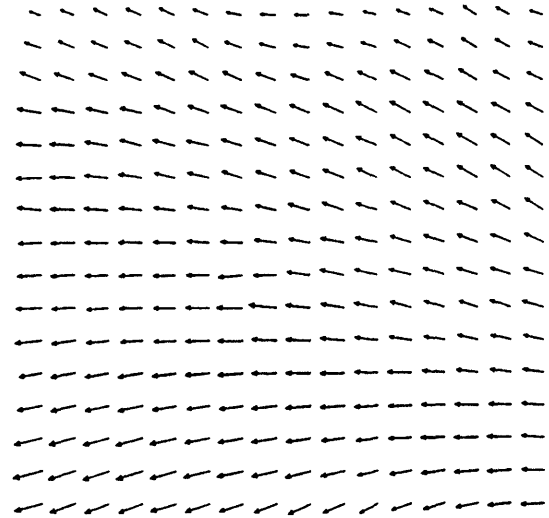
a



b



c



d

Figure 2-25: Chopper sequence flow estimates. (a) Smoothness constraint estimates computed using 200 iterations of SOR, (b) Multiscale Regularization (MR) estimates, (c) Post-filtered MR estimates and (d) Estimates produced by using MR estimates as initial condition for SOR algorithm.

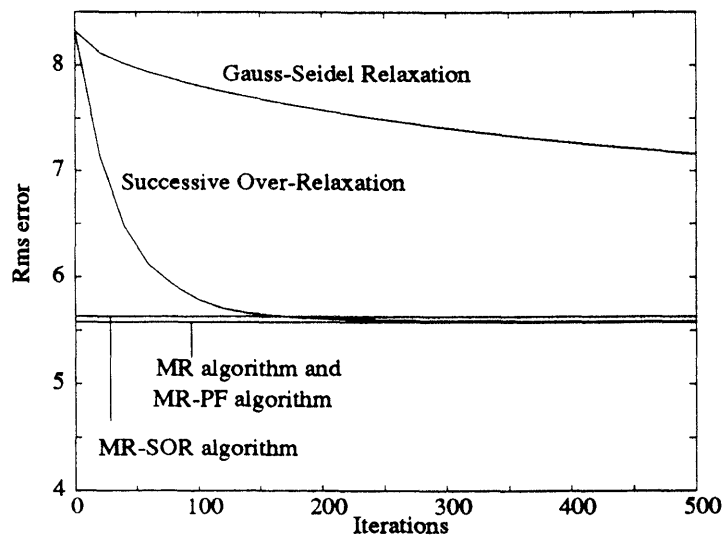


Figure 2-26: Rms Error Comparison of MR, SOR and Gauss-Seidel (GS) algorithm flow estimates for the Chopper sequence.

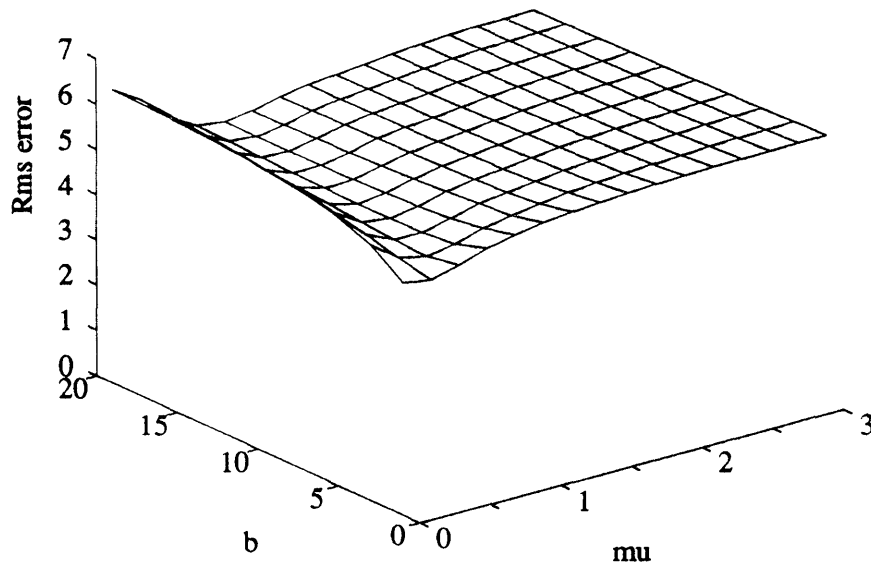


Figure 2-27: Multiscale Regularization rms error sensitivity to the parameters  $b$  and  $\mu$  (Chopper sequence).

could be used to detect this motion discontinuity and obtain rough (i.e. coarse level) motion estimates. While it is beyond the scope of this chapter to develop such a scheme in detail, we can provide an indication of how the MR method provides the essential elements for an effective solution.

The starting point for this is the well-known criterion of global smoothness con-

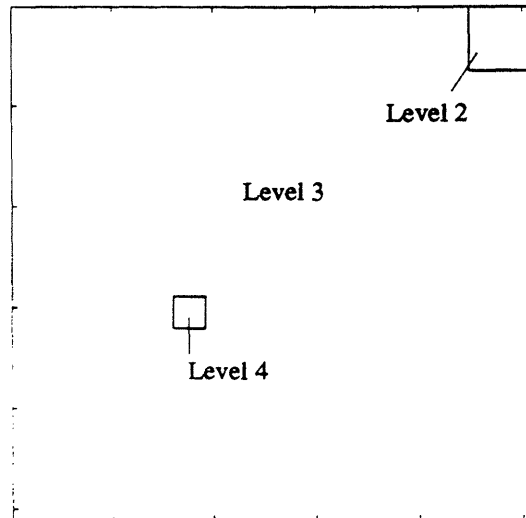


Figure 2-28: Map illustrating the optimal resolution for the Chopper sequence optical flow estimates.

straint type formulations such as ours, namely that they tend to obscure localized motions such as that due to the helicopter in Figure 2-24. This is not surprising since SC-type formulations yield what are in essence low-pass spatial filters. However, there is an extremely critical point that is well-known in Kalman filtering theory and in that relating to the use of such filters for the detection of abrupt changes in time series or dynamic systems. Specifically, such filters can *also* be used to implement *high-pass* filters which produce outputs that not only *enhance* the discontinuities to be detected but also make optimal detection possible. Specifically, the residuals or innovations in a Kalman filter, that is, the difference between the observations and predicted observations based on model and data, represent a statistically whitened version of the observations resulting from what is in essence a high-pass filter. As discussed in many papers and books ([6, 145], for example), discontinuities in the data being processed then lead to distinctive *signatures* which can be looked for using optimal detection methods.

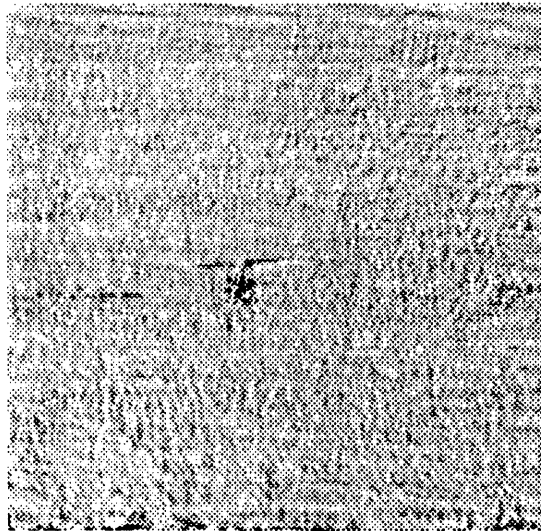


Figure 2-29: The smoothing filter residuals shown above can be used to develop adaptive algorithms for the motion-based object detection.

In a similar fashion we can compute the residuals of the MR estimates:

$$\nu(s) = y(s) - C(s)\hat{x}^s(s) \quad (2.26)$$

for the chopper sequence, an image of which is illustrated in Figure 2-29. Note that in contrast to the original image in Figure 2-24, this residual image does not display *any* coherent structure other than the helicopter, making detection of the helicopter a far easier task in this domain. Furthermore, high pass filtering has in fact enhanced the chopper signature, as the helicopter rotors, nearly imperceptible in Figure 2-24 are clearly in evidence in Figure 2-29 because of the motion discontinuity. As we have indicated, statistically optimal methods for using residuals analogous to these have been developed for time series, and, as discussed in [6, 145], such methods require error covariance information from the estimator in order to specify the optimal detection procedure. Since the MR algorithm also produces such error covariance information it is possible to develop optimal detection methods in this imaging context as well.

## 2.4 Summary

We have presented a new approach to the regularization of ill-posed inverse problems, and have demonstrated its potential through its application to the problem of computing optical flow. This approach starts from the “fractal prior” interpretation of the smoothness constraint introduced by Horn and Schunck to motivate regularization based on multiscale stochastic models. This new formulation leads to an extremely efficient, non-iterative, scale-recursive solution, yielding substantial savings over the iterative algorithms required for the smoothness constraint solution. In particular for  $256 \times 256$  or  $512 \times 512$  images, the algorithm leads to computational savings on the order of a factor of 10 to 100. Indeed, since the iterative approaches associated with the smoothness constraint solution typically require progressively more iterations as the image grows, whereas the per pixel computation associated with the MR algorithm is independent of image size, even larger savings can be realized for larger image domains.

# Chapter 3

## Multiscale Representations of Markov Random Fields

### 3.1 Introduction

In this chapter, we describe how the class of multiscale stochastic models introduced in Chapter 1 can be used to represent 1-D Markov and reciprocal processes and 2-D Markov random fields (MRF's). Markov models in one dimension provide a rich framework for modeling a wide variety of biological, chemical, electrical, mechanical and economic phenomena [16]. Moreover, the Markov structure makes the models very simple to analyze, so that they often can be easily applied to statistical inference problems (such as detection, parameter identification and state estimation) as well as problems in system design (e.g. control and queuing systems).

In two dimensions, MRF's also have been widely used as models for physical systems [9, 13, 102, 48], and more recently for images. For example, Gaussian fields [148] have been used as image texture models [35, 34, 71, 23, 91, 90], and the more general Gibb's fields have been used as prior models in image segmentation, edge detection and smoothing problems [14, 53, 103, 95]. Causal sub-classes of MRF's, such as Markov Mesh Random Fields [1, 42] and Non-Symmetric Half-Plane Markov chains [67] lead to two-dimensional versions of Kalman filtering algorithms when the fields are Gaussian [149]. In addition, efficient fast Fourier transform algorithms are avail-

able for stationary Gaussian fields defined on toroidal lattices [35, 71, 24]. In general, however, Markov random field models lead to computationally intensive algorithms (e.g. stochastic relaxation [53]) for estimation problems. In addition, parameter identification is difficult for MRF models due to the problem of computing the partition function [13, 76, 105]. Thus, while Markov random fields provide a rich structure for multidimensional modeling, they do not generally lead to the simple analysis and computationally efficient algorithms that 1-D Markov processes do.

These computational issues are the most important obstacle to the application of MRF models to a broader range of problems, and are the principal motivations for the investigation in this chapter of the richness of the class of multiscale stochastic processes, and in particular of how such multiscale processes can be used to exactly and approximately represent Markov random fields. We demonstrate how a simple generalization of the model (1.16) leads to classes of models which can be used to represent *all* 1-D Markov processes and 2-D Markov random fields. The significance of this result is not only that it opens the door to the possibility of new and efficient algorithms for MRF models, but also that it suggests that this multiscale modeling framework may be a decidedly superior basis for image and random field modeling and analysis than MRF's both because of the efficient algorithms it admits *and* because of the rich class of phenomena it can be used to describe.

The multiscale representations developed here rely on a generalization of the mid-point deflection technique for constructing a Brownian motion in one dimension [41, 51, 82]. To construct a Brownian motion sample path over an interval by mid-point deflection, we start by randomly choosing values for the process at the mid-point and the two boundary points of the interval according to the joint probability distribution implied by the Brownian motion model. We then use these three values to compute the expected values of the Brownian motion at the one-fourth and three-fourths points of the interval. The expected value at the one-fourth (three-fourths) point corresponds to the average of the initial and mid-point values (mid-point and final values) as shown in the upper left of Figure 3-1. Random values, with appropriate error variances, are then added to the predictions at each of these new points. The critical observation

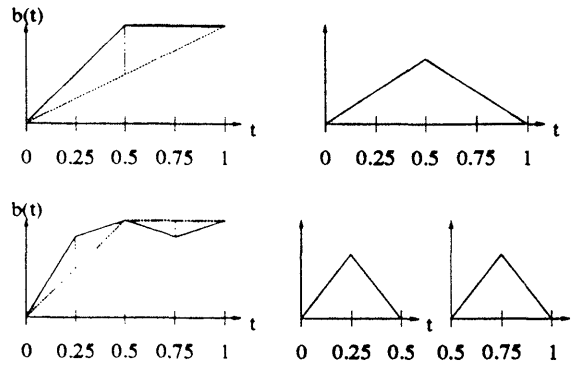


Figure 3-1: The first two levels of a “mid-point deflection” construction of a Brownian motion sample path are shown on the left. The construction generates a sequence of approximations based on linear interpolations of samples of the Brownian motion at the dyadic points. On the right, the basis functions, integrals of the Haar wavelet, in this construction are shown.

to be made here is that, since the Brownian motion process is a Markov process, its value at the one-fourth point, given the values at the initial point and mid-point is *independent* of the process values beyond the mid-point, in particular the values at the three-fourths and end-points of the interval. Obviously, it is also the case that the value at the three-fourths point is independent of the values at the initial and one-fourth points, given the values at the mid-point and final point. Consequently, the random deflection terms used to generate the values of the Brownian motion at the one-fourth and three-fourths points can be chosen *independently*. In addition, we see that the Markov property of Brownian motion allows us to iterate this process, generating values at increasingly dense sets of dyadic points in the interval.

There are several important observations to be made about the preceding development. The first is that, by linearly interpolating at each level in this procedure, as illustrated in Figure 3-1, a sequence of continuous approximations of a Brownian motion is constructed, and the statistics of these approximations converge to those of a Brownian motion [41]. Indeed, this sequence of linear spline approximations can be interpreted exactly as a non-orthogonal multiscale approximation using as the scaling function the triangular “hat” function [127] which is the integral of the Haar wavelet [51]. Second, as we will see, the structure of this mid-point deflection construction fits precisely into our multiscale modeling framework, and corresponds simply to a



particular choice of the parameters in a dyadic tree multiscale model of the form (1.16). Moreover, this concept generalizes, allowing us to show that *all* reciprocal and Markov processes in one dimension<sup>1</sup> can be represented by multiscale stochastic models in a similar way. Thus, in one dimension we will show that the class of processes realizable via multiscale, scale-recursive models is at least as rich as the class of all Markov and reciprocal processes. In fact, as we will illustrate, it is significantly richer than this.

Furthermore, these same ideas can be extended to multidimensional processes. In particular, we show how a generalization of the mid-point deflection concept to a “mid-line” deflection construction can be used to represent all 2-D Markov random fields with multiscale models defined on quadrees. In particular, the key to our multiscale representations in one or two dimensions is a partitioning of the domain over which the process is defined so that the coarse-to-fine construction of the process can proceed independently in each subdomain. Markovianity plus knowledge of the process on the boundaries of the subdomain partition make this possible. The fundamental difference, however, between the 1-D and 2-D cases is due to the fact that boundaries in  $\mathcal{R}^2$  correspond to curves or in  $\mathcal{Z}^2$  to sets of connected lattice sites, as opposed to pairs of points in one dimension. Because of this difference, exact multiscale representations of MRF’s defined over a subset of  $\mathcal{Z}^2$  have a dimension which varies from scale to scale, and which depends on the size of the domain over which the MRF is defined.

As a consequence, in addition to the exact representations, we will introduce a family of approximate representations for Gaussian MRF’s (GMRF’s) based on wavelet transforms. As we have indicated, maintaining complete knowledge of a process on 2-D boundaries leads to models of scale-varying dimension, which can become prohibitively large for domains of substantial size. On the other hand, at coarser scales, it would seem reasonable to keep only coarse approximations to these boundary values,

---

<sup>1</sup>Note that this also includes all so-called higher order Markov and reciprocal processes. For example, a second order Markov process, i.e. one for which the value of the process  $z(t)$  at time  $t$  depends on both  $z(t-1)$  and  $z(t-2)$ , can be represented as a *vector* first order Markov process  $[z(t), z(t-1)]^T$ , which can then be represented in the manner developed in Section 3.3.

and this leads naturally to the use of a multiscale change of basis for the representation of the values of a 2-D process along each 1-D boundary. That is, through our mid-line deflection based models, we are led to the idea of using *one-dimensional wavelet transforms* in the representation of the values of a *two-dimensional* GMRF. The result is a family of models, ranging from those which keep only the coarsest wavelet coefficients along each 1-D boundary to the exact model which keeps them all. This family of approximate representations allows one to tradeoff the complexity and accuracy of the representations.

We demonstrate our framework for wavelet-based approximate representation of Gaussian MRF's in the context of natural texture representation. In particular, classes of GMRF's have been widely used to represent natural textures in the context of segmentation and anomaly detection applications [23, 35, 34, 33, 71, 91, 90], and we illustrate how these models can be approximated in our multiscale framework. In addition, we illustrate how the fidelity of the approximation varies with the characteristics of the GMRF being approximated and with the complexity of the approximate representation.

This chapter is organized as follows. In Section 3.2 we describe a non-Gaussian generalization of the multiscale stochastic models (1.16). In Section 3.3 we develop the details of the representation of Brownian motion discussed above, and generalize this idea to allow the representation of all 1-D Markov and reciprocal processes. These ideas are then further generalized in Section 3.4 to provide exact and approximate representations of MRF's. In Section 3.5 we illustrate how the approximate models can be used to represent GMRF texture models. In Section 3.6, we summarize the results of this chapter.

## 3.2 Generalized Multiscale Stochastic Models

In this section we describe a simple generalization of the multiscale model (1.16) which allows for more general (non-Gaussian) processes. As we indicated in Chapter 1, a basic property of the model (1.16) is the Markovianity of the state with respect

to the ordering structure defined by the tree.

More precisely, on a  $q^{\text{th}}$ -order tree, let  $\Upsilon_i^s, i = 1, \dots, q + 1$  denote the  $q + 1$  subsets of states which correspond to viewing node  $s$  as a boundary (c.f. the discussion immediately preceding Section 1.3). Then, (1.16), along with the assumption that the driving noise  $w(s)$  is white, implies that:

$$p_{v_1^s, \dots, v_{q+1}^s | x_s}(\Upsilon_1^s, \dots, \Upsilon_{q+1}^s | X_s) = \prod_i p_{v_i^s | x_s}(\Upsilon_i^s | X_s) \quad (3.1)$$

By requiring only this property to hold, we obtain a much wider class of processes than that given by (1.16), but still retain the essential properties leading to the efficient algorithms described in Section 1.3 and Chapter 4. In particular, recalling that  $m(s)$  denotes the scale of node  $s$ , the property (3.1) not only implies that the tree processes are Markov in scale, from coarse-to-fine, but also that the conditional pdf of the state at node  $s$ , given the states at all previous scales, depends *only on the state at the parent node  $s\bar{\gamma}$* :

$$p_{x(s) | x(\sigma), m(\sigma) < m(s)}(X_s | X_\sigma, m(\sigma) < m(s)) = p_{x(s) | x(s\bar{\gamma})}(X_s | X_{s\bar{\gamma}}) \quad (3.2)$$

Such tree processes are naturally defined by specifying the parent-offspring conditional pdf's, along with a pdf for the state at the root node of the tree. A simple example of a stochastic process in this general class is the following discrete-state stochastic process  $x(s) \in \{0, 1, \dots, L\}$  with parent-offspring conditional probability mass functions given by:

$$p_{x(s) | x(s\bar{\gamma})}(X_s | X_{s\bar{\gamma}}) = \begin{cases} \theta_{m(s)} & \text{if } X_s = X_{s\bar{\gamma}} \\ (1 - \theta_{m(s)})/L & \text{if } X_s \neq X_{s\bar{\gamma}} \end{cases} \quad (3.3)$$

where  $p_{x_0}(X_0) = 1/(L + 1)$  for  $X_0 \in \{0, 1, \dots, L\}$  and  $\theta_{m(s)}$  is a number between 0 and 1 which may vary with scale  $m(s)$ . A class of processes with this structure and defined on a quadtree has been proposed by Bouman for segmentation applications [19].

The conditional independence property, (3.1) also implies that such a process can also be viewed as a Markov random field on the tree. In particular, define the neighborhood set  $D_s$  of node  $s$  as its nearest neighbors on the tree<sup>2</sup> (i.e. the parent and offspring nodes). The general multiscale process described above is then a Markov random field on the tree in the sense that:

$$p_{\mathbf{x}(s)|\mathbf{x}(\sigma),\sigma\neq s}(X_s|X_\sigma,\sigma\neq s) = p_{\mathbf{x}(s)|\mathbf{x}(\sigma),\sigma\in D_s}(X_s|X_\sigma,\sigma\in D_s) \quad (3.4)$$

The most general class of joint probability distribution functions which lead to conditional distributions satisfying (3.4) is given by the Hammersley-Clifford theorem [13]. Our focus here is on processes which also satisfy the one-sided Markov property (3.2), because this class of processes leads naturally to efficient scale-recursive algorithms.

### 3.3 Representation of 1-D Reciprocal Processes

In this section we describe the basic properties of reciprocal processes in one dimension, introduce and develop representations of reciprocal processes in terms of multiscale stochastic models, and present several examples.

#### 3.3.1 1-D Reciprocal Processes

A reciprocal process is a first-order MRF on the real line. More formally, a stochastic process  $z(t), t \in \mathcal{R}$  is said to be reciprocal<sup>3</sup> (or bilateral Markov, two-sided Markov or non-causal Markov) if it has the property that the probability distribution of a state in any open interval  $(T_1, T_2)$ , conditioned on the boundary states  $z(T_1), z(T_2)$  is independent of states outside of the interval [43, 81]. That is, for  $t \in (T_1, T_2)$ :

$$P_{z(t)|z(\tau),\tau\in(T_1,T_2)^c}(Z_t|Z_\tau,\tau\in(T_1,T_2)^c) =$$

---

<sup>2</sup>Obvious modifications of the neighborhood set must be made for the root node at the top of the tree, which has no parent, and the nodes at the finest level of the tree, which have no offspring.

<sup>3</sup>The discussion here refers only to *first-order* reciprocal processes. Extension to higher-order processes is straightforward [43].

$$p_{z(t)|z(T_1),z(T_2)}(Z_t|Z_{T_1}, Z_{T_2}) \quad (3.5)$$

where  $(T_1, T_2)^c$  denotes the complement of the open interval  $(T_1, T_2)$ . Reciprocal processes defined on the integers  $\mathcal{Z}$  satisfy the same property with the continuous interval  $(T_1, T_2)$  replaced by the discrete interval  $\{T_1 + 1, T_1 + 2, \dots, T_2 - 1\}$ :

$$p_{z(t)|z(\tau), \tau \in \{T_1+1, \dots, T_2-1\}^c}(Z_t|Z_\tau, \tau \in \{T_1 + 1, \dots, T_2 - 1\}^c) = p_{z(t)|z(T_1),z(T_2)}(Z_t|Z_{T_1}, Z_{T_2}) \quad (3.6)$$

Reciprocal processes are closely related to the class of Markov processes. A process  $z(t)$  on  $\mathcal{R}$  or  $\mathcal{Z}$  is Markov if past and future values of the state are independent given the present. This means that for  $t_2 < t_3$ :

$$p_{z(t_3)|z(t_1), t_1 \leq t_2}(Z_{t_3}|Z_{t_1}, t_1 \leq t_2) = p_{z(t_3)|z(t_2)}(Z_{t_3}|Z_{t_2}) \quad (3.7)$$

As shown in [1], if a process is Markov then it is also reciprocal. On the other hand, reciprocal processes are not necessarily Markov [43], although one can show that essentially all stationary Gaussian reciprocal processes are Markov [80].

### 3.3.2 Exact Representations of 1-D Reciprocal Processes

In the introduction we described a construction of a Brownian motion  $b(t)$  over the unit interval via mid-point deflection. As we noted, this corresponds precisely to a Gaussian multiscale stochastic model of the form (1.16) defined on a dyadic tree. To see this, consider the following multiscale process. At the coarsest level, the initial state  $\mathbf{x}_0$  is a three-dimensional vector whose pdf is given by the joint pdf for the values of a Brownian motion at the initial, middle and final points of the interval:

$$\mathbf{x}_0 \equiv \begin{bmatrix} b(0) \\ b(0.5) \\ b(1) \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, P_0) \quad (3.8)$$

$$P_0 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & 0.5 & 1 \end{bmatrix} \quad (3.9)$$

where we have used the facts that  $b(0) = 0$ ,  $b(t)$  is an independent increments process, and for  $t_1 < t_2$ ,  $b(t_2) - b(t_1) \sim \mathcal{N}(0, t_2 - t_1)$ .

Choosing a value for  $x_0$  as a sample from this distribution corresponds to the first step in the mid-point deflection construction of Brownian motion. The second step in the mid-point deflection construction is the specification of values for the Brownian motion at the one-fourth and three-fourths points. In the context of our multiscale modeling framework, we define two state vectors at the second level of the dyadic tree, each again a 3-tuple. The state on the left represents the values of the Brownian motion at the initial, one-fourth and middle points of the interval,  $[b(0), b(0.25), b(0.5)]$ , and the state on the right represents the corresponding values in the right half-interval,  $[b(0.5), b(0.75), b(1)]$ . The sample at the quarter point is given by linear interpolation of  $b(0)$  and  $b(0.5)$ , plus a Gaussian random variable with variance equal to the variance of the error in this prediction:

$$b(0.25) = \frac{1}{2}(b(0) + b(0.5)) + e(0.25) \quad (3.10)$$

$$e(0.25) \sim \mathcal{N}(0, 0.125) \quad (3.11)$$

Likewise,  $b(0.75)$  is chosen by averaging the end points of the right half-interval,  $b(0.5)$  and  $b(1)$ , and adding in a random value, independent of the deflection term used to create the sample at the one-fourth point:

$$b(0.75) = \frac{1}{2}(b(0.5) + b(1)) + e(0.75) \quad (3.12)$$

$$e(0.75) \sim \mathcal{N}(0, 0.125) \quad (3.13)$$

The above construction of  $b(0.25)$  and  $b(0.75)$  is precisely the same as the mid-point deflection construction of these values. Values of the process at successively finer sets

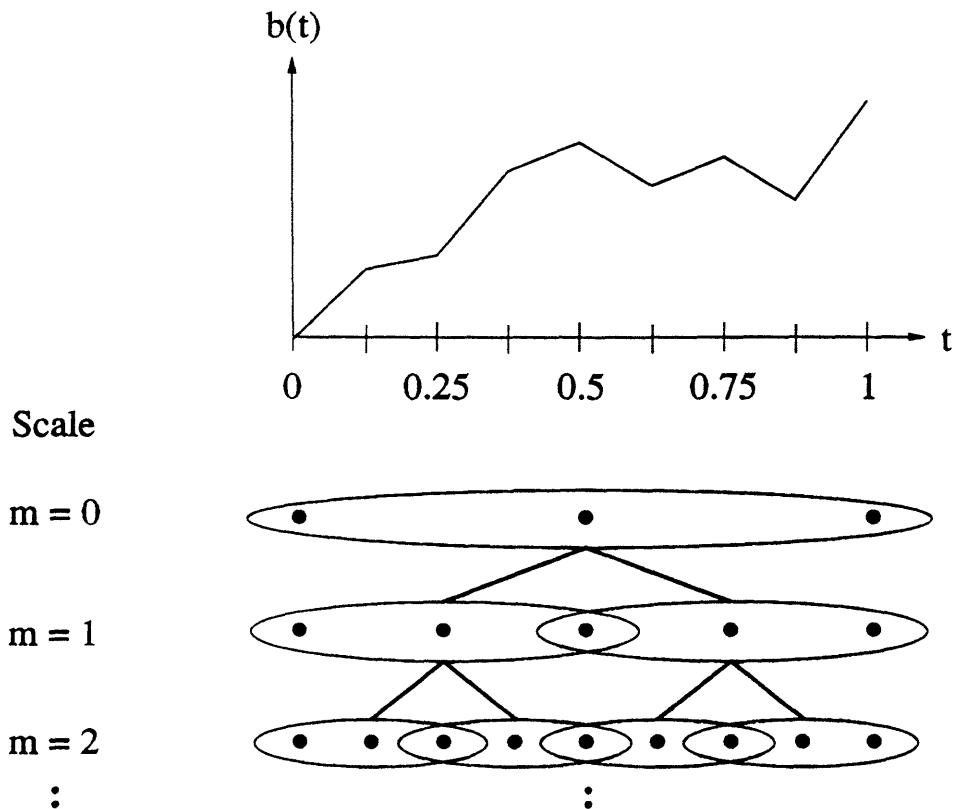


Figure 3-2: The state vectors for the first three levels of a multiscale model representing Brownian motion,  $b(t)$ , are illustrated. At the first level, the state is the vector  $[b(0), b(0.5), b(1)]$ , which is indicated by the three points at  $m = 0$  surrounded by an ellipse. The points are placed directly below the points  $t = 0, 0.5$  and  $t = 1$  on the graph above to indicate that the state of the multiscale process at the first level consists of the values of the Brownian motion at those three points. Likewise, at lower levels, the states are indicated by sets of three points surrounded by ellipses, with the horizontal location of the points in correspondence with time indices in the graph at the top. At the  $m^{\text{th}}$  level, there are  $2^m$  state vectors, each of which consists of the values of  $b(t)$  at three consecutive dyadic points, and which together represent the values of the Brownian motion at  $2^{m+1} + 1$  distinct points on the interval  $[0, 1]$ . The multiscale representation for Brownian motion can be generalized to the class of 1-D reciprocal process, which contains the class of 1-D Markov processes.

of dyadic points are generated in the same way. At the  $m^{\text{th}}$  scale, the values of the process at  $t = k/2^{m+1}, k = 0, 1, \dots, 2^{m+1}$  are represented with  $2^m$  state vectors, each containing the values of the process at three points, as shown in Figure 3-2. At any level, each state is a linear function of its parent, plus an independent noise term. Thus, this construction fits precisely into the multiscale modeling framework given by (1.16) (see Section 3.3.3 for the precise formulae for  $A(s)$  and  $B(s)$ ).

Representation of more general 1-D reciprocal processes via multiscale models is a simple extension of the above idea. To construct a multiscale model for a particular reciprocal process  $z(t), t \in [0, 1]$ , start by choosing the state at the coarsest level as a sample from the joint distribution:

$$P_{z(0), z(0.5), z(1)}(Z_0, Z_{0.5}, Z_1) \quad (3.14)$$

This generalizes the choice in (3.8) in which the state at the top level is chosen using the Gaussian distribution corresponding to a Brownian motion. The two state vectors at the second level are again the three-dimensional vectors  $[z(0), z(0.25), z(0.5)]$  and  $[z(0.5), z(0.75), z(1)]$ , where values for the half-interval mid-points are chosen as samples from the conditional distributions:

$$P_{z(0.25)|z(0), z(0.5)}(Z_{0.25}|Z_0, Z_{0.5}) \quad (3.15)$$

$$P_{z(0.75)|z(0.5), z(1)}(Z_{0.75}|Z_{0.5}, Z_1) \quad (3.16)$$

Since the process is reciprocal,  $z(0.25)$  and  $z(0.75)$  are conditionally independent given the state at the first level, and thus the modeling structure fits precisely into the more general non-linear model class described in Section 3.2.

The construction above assumes that the process is defined over a continuous interval. In practice, we are typically concerned with processes  $z(t)$  on a discrete interval,  $t \in \{0, 1, \dots, T\}$ . If  $T = 2^N$  for some integer  $N$ , then we can use essentially the same construction as for the continuous case above. Specifically,  $x_0 \equiv [z(0), z(T/2), z(T)]$  is a random vector chosen from the appropriate distribution for the process of interest. The states at the second level are  $[z(0), z(T/4), z(T/2)]$  and  $[z(T/2), z(3T/4), z(T)]$ , with the half-interval mid-points again chosen using the appropriate distribution. Since there are only a finite number of points in the discrete process, only a finite number of levels are needed to exactly represent it. In particular, with  $T = 2^N$ ,  $N$  levels are required.

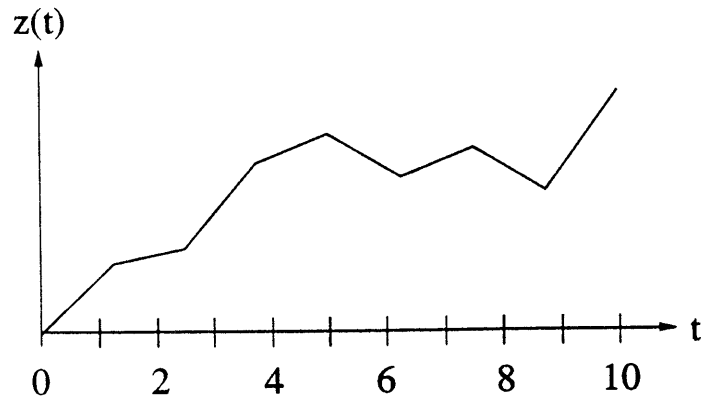
There are several observations to be made about the continuous and discrete-time



construction we have just described. The first is that there is no fundamental difficulty in choosing a point other than the mid-point at each level in these constructions. For example, in the construction of Brownian motion, starting from the initial set of points given in (3.8), we could next generate any pair of points on either side of 0.5, e.g.  $b(0.1)$  and  $b(0.7)$ . However, the regular structure implied by the choice of mid-points may be of some value for processes such as Brownian motion which have stationary increments, as they lead to models in which the model parameters, such as  $A(s)$  and  $B(s)$  in (1.16) have very simple and regular characterizations as a function of node  $s$  and scale  $m(s)$  (see, for example, (3.40),(3.41)). This regularity in turn leads to simplifications in the structure of algorithms for estimation and signal processing, requiring fewer distinct gains to be calculated and, if parallel implementation is considered, allowing SIMD (single instruction, multiple data) rather than MIMD (multiple instruction, multiple data) implementations.

Secondly, in discrete-time, there will always be at least some degree of irregularity in the multiscale model if the process is defined over  $t \in \{0, 1, \dots, T\}$  and  $T$  is not a power of two. In particular, in such a case the structure of the tree and/or the state needed in the multiscale representation of this process will need to be modified. For example, consider a process defined over  $t \in \{0, 1, \dots, 10\}$ . In this case, we can develop a model of the type we have described in which the tree is of non-uniform depth and in which we do not have mid-point deflection at some nodes, as indicated in Figure 3-3a (e.g. in the generation of the value at  $t = 3$  given values at 0 and 5). Alternatively, as shown in Figure 3-3b, we may be able to achieve some level of (and perhaps complete) symmetry by generating *more* than one new point at some nodes (e.g. in Figure 3-3b we generate values at both  $t = 2$  and  $t = 3$  given values at 0 and 5). Obviously, as in standard discrete signal processing applications in which the FFT is to be used, there are considerable efficiencies to be had if  $T$  is a power of 2.

Furthermore, as we have indicated previously, while our development has focused on first-order reciprocal processes, the extension to higher-order models is straightforward. Indeed, a  $K^{\text{th}}$ -order model defined on  $t \in \{1, 2, \dots, K(T + 1)\}$ , where  $T$  is a power of 2, can be accommodated by grouping states at adjacent points into sets of



Scale

$m = 0$



$m = 1$



$m = 2$



$m = 3$



a

Scale

$m = 0$



$m = 1$



$m = 2$



b

Figure 3-3: The state vectors are shown for two possible multiscale representations for a reciprocal process which is defined on a discrete interval of the form  $\{0, 1, \dots, 10\}$ . In (a), a dyadic tree with uniform state dimension, but non-uniform depth is used, whereas in (b) a dyadic tree of uniform depth but non-uniform state size is used.

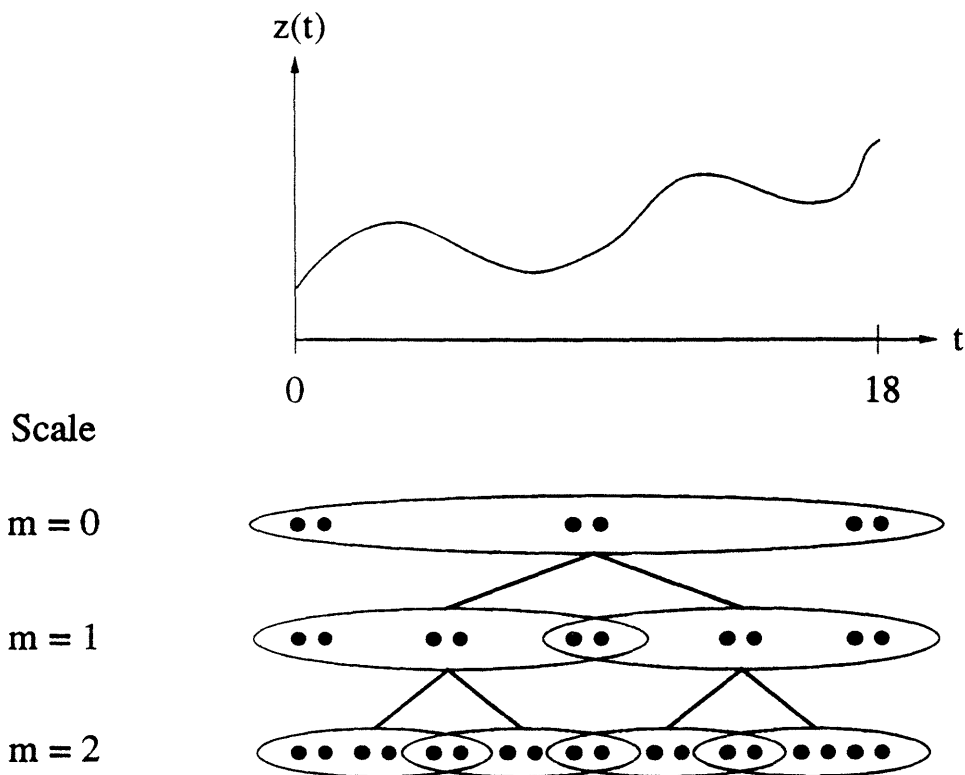


Figure 3-4: The state vectors are shown for a multiscale representation of a second-order reciprocal process defined on a discrete interval. In this case, the state vectors are composed of three groups of two points, reflecting the fact that the value at the current point, say  $z(t_0)$  is independent of the values at all other times, given the *pairs* of nearest neighbors  $z(t_0 - 1)$ ,  $z(t_0 - 2)$  and  $z(t_0 + 1)$ ,  $z(t_0 + 2)$ .

size  $K$ . The states at different levels of the tree might be as depicted in Figure 3-4 for a second-order reciprocal process defined over  $t \in \{0, 1, \dots, 18\}$ . Higher-order models can equivalently be represented by simply redefining the state of the process  $z(t)$  to be a vector of appropriate dimension.

The representations we have introduced to this point have obvious and substantial levels of redundancy. For example, the value of  $z(T/2)$  appears in the state vector at both nodes at the second level of the multiscale model we have described for discrete-time reciprocal processes. More generally, at the  $m^{\text{th}}$  level of the model for such a process there are  $2^m$  state vectors containing a total of  $3 \times 2^m$  values, only  $2^{m+1} + 1$  of which are distinct. This redundancy is actually of minimal consequence for estimation and likelihood calculation algorithms based on these models (see, for

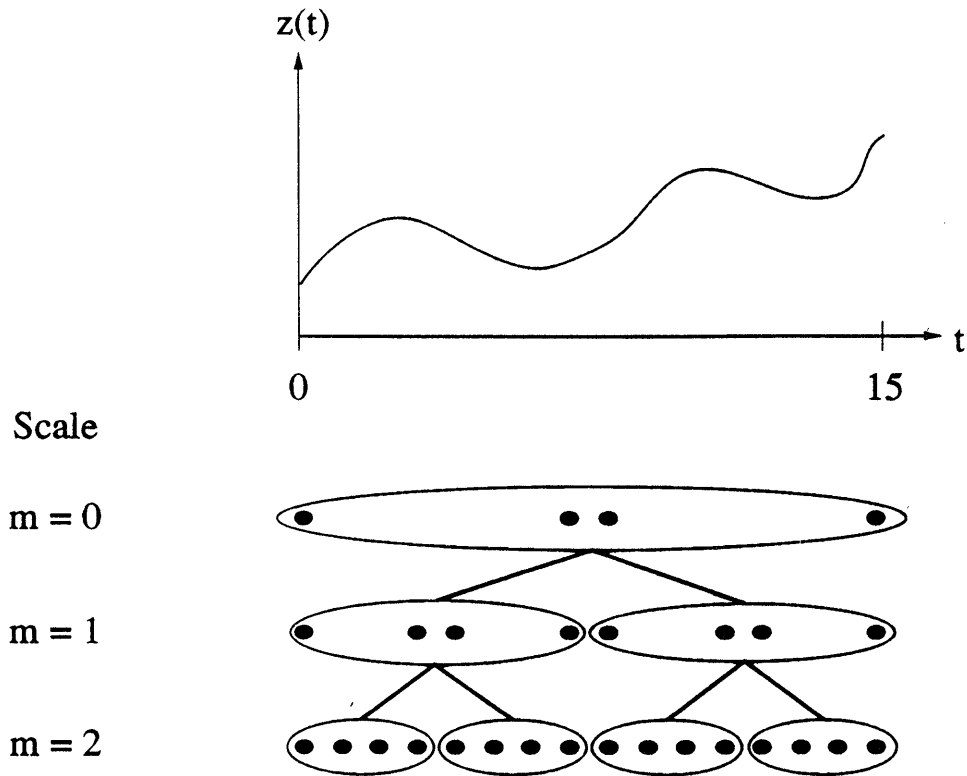


Figure 3-5: The state vectors are shown for a *non-redundant* multiscale representation of a 1-D reciprocal process. These non-redundant representations, appropriately generalized for the 2-D case, are useful in the context of wavelet-based *approximate* representations of Gaussian MRF's.

instance, Appendix C). However, it is also easy to eliminate the redundancy by a simple modification to the construction we have described. In particular, we may generate *two* internal points between each pair of points at each stage in the level-to-level recursion, yielding a four-dimensional state vector. For example, if the reciprocal process is defined over  $t \in \{1, 2, \dots, 16\}$ , then we can choose the non-redundant set of state vectors illustrated in Figure 3-5. In this case, a first-order reciprocal process is represented by a process with a four-dimensional state, instead of the process with a three-dimensional state used earlier. In general, at the  $m^{\text{th}}$  level of such a representation, there are  $2^m$  state vectors representing  $2^{m+2}$  distinct values of the process. Again, in the situation where  $T$  is not a power of two, some irregularity in the structure will be introduced.

Once we allow ourselves to consider such variants on the original mid-point deflec-

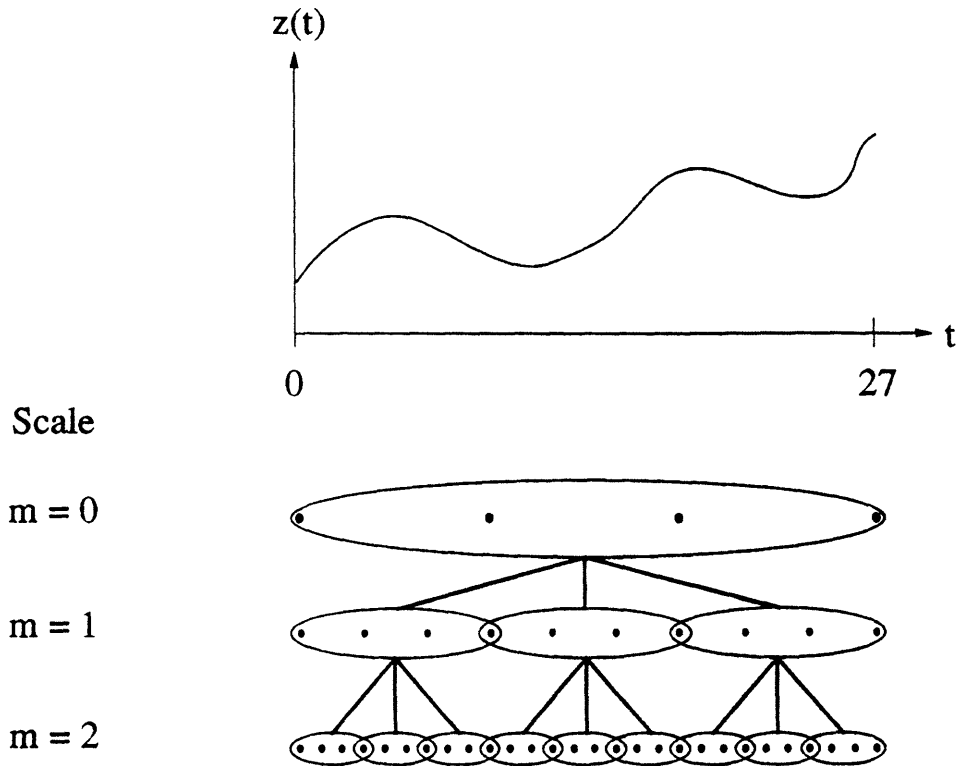


Figure 3-6: The state vectors are shown for a multiscale representation on a *third-order* tree.

tion construction in which more than one new point is generated between each pair of previously constructed points, we see immediately that it is possible to generate multiscale representations on trees that are not dyadic. For example, consider a reciprocal process defined on  $t \in \{0, 1, \dots, 3^N\}$ . This process is most conveniently represented on the regular structure of a third-order tree, as shown in Figure 3-6. This flexibility of the modeling framework allows the possibility of considering different tradeoffs in terms of level of parallelization and computational power of individual processors when implementing estimation algorithms such as the multiscale smoothing algorithm used in Chapter 2 or the likelihood calculation algorithm to be described in Chapter 4.

Finally, it is of interest to note that the construction we have described, and its several variants, can be interpreted as a non-iterative Gibb's sampler. The Gibb's sampler introduced in [53] is an iterative algorithm for the generation of sample paths of Markov random fields on a discrete lattice. For 1-D discrete-time reciprocal

processes this procedure reduces to using the nearest neighbor conditional probability functions to construct a Markov chain which has an asymptotic distribution equal to the joint distribution of the process. Specifically, at each step of the procedure we modify the current sample path  $z_k(t)$ ,  $t \in \{0, 1, \dots, T\}$  by replacing the value at some point in time, say  $t_0$  with a random value chosen according to the conditional distribution for the process at that point given the current values of the sample path at  $t_0 - 1$  and  $t_0 + 1$ . By cycling repeatedly through all of the time points, the sample path is guaranteed to converge to one with the correct joint statistics. The procedure is conceptually simple but computationally intensive, since the Markov chain requires many transitions for the probability function to converge. In contrast, in our construction, we successively generate samples at new points (e.g. mid-points) conditioned on values at previously generated points, which are *not* nearest neighbors but rather boundary points that partition the time interval of interest. For this reason, and since we begin at the root node with a decimated set of values with the correct distribution, we are *guaranteed* that at each stage the decimated process that is constructed has exactly the correct distribution. Thus, with this structure we visit each time point only once and construct a sample path non-iteratively.

In fairness, an important point to note here is that if a reciprocal process is specified directly in terms of a Gibb's distribution — i.e. in terms of nearest neighbor energy functions (see, for example, [53]) — then the calculation of the nearest neighbor pdf's required in the Gibb's sampler is simple. The question then is whether it is also simple to determine the conditional pdf's — e.g. the pdf for  $z(T/2)$  given  $z(0)$  and  $z(T)$  — needed to implement the non-iterative, multiscale procedures we have described. In general, this may not be a straightforward task, since, if we begin with a Gibb's distribution the computation of such pdf's requires explicit calculation of quantities related to the so-called *partition function* [53], which can be quite complex. On the other hand, such calculations need only be done once to construct the pdf's needed for the multiscale model. In addition, as we have seen for Brownian motion and as we now illustrate further, in many cases, including all vector Gauss-Markov processes and  $L$ -state Markov chains, closed form expressions can be derived for the

multiscale representations.

### 3.3.3 Examples

In this section we discuss several examples of reciprocal processes and their multiscale representations. The first examples describe multiresolution models for general vector Gauss-Markov processes specified in state-space form, and, in particular describe this construction in detail for two cases corresponding to the integral of white noise (i.e. Brownian motion) and the second integral of white noise. These examples allow us to illustrate the interpretation of these multiresolution models as providing approximations using non-orthogonal expansions. In particular, our model for Brownian motion corresponds to the use of the so-called “hat” function [127] in this expansion, leading to linear interpolation between dyadic points, while the model for the integral of Brownian motion leads to a multiresolution approximation using *cubic* interpolation.

The second part of this section presents several discrete-state examples, the first of which investigates general  $L$ -state Markov chains and allows us to make contact with the models used in [17, 19, 20] for segmentation applications. The second example is a general two-state process, which is used to demonstrate that the class of multiscale models is in fact far *richer* than the class of Markov processes. In particular, through this example we gain insight into the very particular conditions that the parent-to-child transition pdf’s must satisfy in order for the finest level process to be Markov. This analysis suggests that Markov chains are, in fact, only a small subset of the processes realizable with multiscale models and, in particular, directly motivates several other multiscale mid-point deflection processes which are not Markov.

Finally, we derive a multiscale representation for the 1-D Ising model. This model and its multidimensional generalizations are widely studied in the statistical mechanics community due to the fact that the Ising model is the simplest example of a Gibb’s distribution which exhibits a *phase transition*.

## Gauss-Markov Processes

Consider a vector Gauss-Markov process defined on the interval  $[0, 1]$  and given by:

$$\dot{z}(t) = F_t z(t) + G_t \mu(t) \quad (3.17)$$

where:

$$z(0) \sim \mathcal{N}(0, \Pi_0) \quad (3.18)$$

$$\mathbf{E}\{\mu(t)\mu(\tau)^T\} = I \delta(t - \tau) \quad (3.19)$$

$$\mathbf{E}\{\mu(t)z(0)^T\} = 0 \quad (3.20)$$

Define the state transition matrix  $\Phi(t, \tau)$  by:

$$\dot{\Phi}(t, \tau) = F_t \Phi(t, \tau) \quad (3.21)$$

$$\Phi(t, t) = I \quad (3.22)$$

and state covariance matrix  $\Pi_t = \mathbf{E}\{z(t)z(t)^T\}$ . As is well known [41], the state covariance matrix satisfies the following differential and integral equations:

$$\dot{\Pi}_t = F_t \Pi_t + \Pi_t F_t^T + G_t G_t^T \quad (3.23)$$

$$\Pi_t = \Phi(t, 0) \Pi_0 \Phi(t, 0)^T + \int_0^t \Phi(t, \tau) G_\tau G_\tau^T \Phi(t, \tau)^T d\tau \quad (3.24)$$

Also, define the conditional expectation of the state at time  $t_2$  given the states  $z(t_1)$  and  $z(t_3)$ , and the corresponding covariance of the error as:

$$\hat{z}_{t_2|t_1, t_3} = \mathbf{E}\{z(t_2)|z(t_1), z(t_3)\} \quad (3.25)$$

$$P_{t_2|t_1, t_3} = \mathbf{E}\{(z(t_2) - \hat{z}_{t_2|t_1, t_3})(z(t_2) - \hat{z}_{t_2|t_1, t_3})^T\} \quad (3.26)$$

Since:

$$p_{z(t_2)|z(t_1), z(t_3)}(Z_{t_2}|Z_{t_1}, Z_{t_3}) = \mathcal{N}(Z_{t_2}; \hat{z}_{t_2|t_1, t_3}, P_{t_2|t_1, t_3}) \quad (3.27)$$



the conditional expectation (3.25) and error covariance (3.26) are the quantities we require to construct a multiscale representation of the process (3.17) – (3.20). In fact, it is easy to show that for  $t_1 < t_2 < t_3$ :

$$\hat{z}_{t_2|t_1,t_3} = \begin{bmatrix} \Pi_{t_1} \Phi(t_2, t_1)^T \\ \Phi(t_3, t_2) \Pi_{t_2} \end{bmatrix}^T \begin{bmatrix} \Pi_{t_1} & \Pi_{t_1} \Phi(t_3, t_1)^T \\ \Phi(t_3, t_1) \Pi_{t_1} & \Pi_{t_3} \end{bmatrix}^{-1} \begin{bmatrix} z(t_1) \\ z(t_3) \end{bmatrix} \quad (3.28)$$

$$P_{t_2|t_1,t_3} = \Pi_{t_2} - \begin{bmatrix} \Pi_{t_1} \Phi(t_2, t_1)^T \\ \Phi(t_3, t_2) \Pi_{t_2} \end{bmatrix}^T \begin{bmatrix} \Pi_{t_1} & \Pi_{t_1} \Phi(t_3, t_1)^T \\ \Phi(t_3, t_1) \Pi_{t_1} & \Pi_{t_3} \end{bmatrix}^{-1} \begin{bmatrix} \Pi_{t_1} \Phi(t_2, t_1)^T \\ \Phi(t_3, t_2) \Pi_{t_2} \end{bmatrix} \quad (3.29)$$

Equations (3.28) and (3.29) directly provide us with the parameters of the matrices  $A(s)$ ,  $B(s)$  and  $P_0$  in the multiscale model (1.16) corresponding to our mid-point deflection construction. In particular, let us identify the abstract index  $s$  with a pair of numbers  $(m, \varphi)$  which denote the scale and horizontal shift of the node  $s$ , respectively. The horizontal shift  $\varphi$ , running from 0 to  $2^m - 1$ , indexes the nodes at scale  $m$ . For instance, the root node is associated with the pair  $(1,0)$ , and the left and right nodes at the first level are associated with  $(2,0)$  and  $(2,1)$ , respectively. With this notation, the state at node  $s$  on the tree contains the values of the process  $z(t)$  at the particular three points:

$$x(s) \equiv x((m, \varphi)) = \begin{bmatrix} z(2\varphi/2^{m+1}) \\ z((2\varphi + 1)/2^{m+1}) \\ z((2\varphi + 2)/2^{m+1}) \end{bmatrix} \quad (3.30)$$

From the description of the general construction, the *form* of the matrix  $A(s)$  in (1.16)

is clear:

$$A(\mathbf{s}) \equiv A((m, \varphi)) = \begin{cases} \begin{bmatrix} I & 0 & 0 \\ K_1 & K_2 & 0 \\ 0 & I & 0 \end{bmatrix} & \text{if } \varphi \text{ is even} \\ \begin{bmatrix} 0 & I & 0 \\ 0 & K_1 & K_2 \\ 0 & 0 & I \end{bmatrix} & \text{if } \varphi \text{ is odd} \end{cases} \quad (3.31)$$

In particular, if  $\varphi$  is even, then the first and third components of the state  $\mathbf{x}(\mathbf{s})$  in (3.30) correspond to the *first and second* components of  $\mathbf{x}(s\bar{\gamma})$ . Thus, the identity matrices in (3.31) for  $\varphi$  even simply map the first and second components of  $\mathbf{x}(s\bar{\gamma})$  to the first and third components of  $\mathbf{x}(\mathbf{s})$ . In addition, the mid-point prediction of  $\mathbf{z}((2\varphi + 1)/2^{m+1})$  is just a linear function of the first two components of the parent  $\mathbf{x}(s\bar{\gamma})$ , which is expressed via the matrices  $K_1$  and  $K_2$  in the second row of (3.31). The matrix  $A(\mathbf{s})$  for  $\varphi$  odd is similar, and in fact is just a “shifted” version of  $A(\mathbf{s})$  for  $\varphi$  even (reflecting the fact that the interpolation down to the state on the right depends on the *last* two components of  $\mathbf{x}(s\bar{\gamma})$ ).

The gain matrices in (3.31) can be computed directly from (3.28). Using standard formulas for the inversion of a block  $2 \times 2$  matrix, we compute:

$$K_1 = \Phi(t_2, t_1) + \Phi(t_2, t_1)\Pi_{t_1}\Phi(t_3, t_1)^T(\Pi_{t_3} - \Phi(t_3, t_1)\Pi_{t_1}\Phi(t_3, t_1)^T)^{-1}\Phi(t_3, t_1) \\ - \Pi_{t_2}\Phi(t_3, t_2)^T(\Pi_{t_3} - \Phi(t_3, t_1)\Pi_{t_1}\Phi(t_3, t_1)^T)^{-1}\Phi(t_3, t_1) \quad (3.32)$$

$$K_2 = -\Phi(t_2, t_1)\Pi_{t_1}\Phi(t_3, t_1)^T(\Pi_{t_3} - \Phi(t_3, t_1)\Pi_{t_1}\Phi(t_3, t_1)^T)^{-1} \\ + \Pi_{t_2}\Phi(t_3, t_2)^T(\Pi_{t_3} - \Phi(t_3, t_1)\Pi_{t_1}\Phi(t_3, t_1)^T)^{-1} \quad (3.33)$$

where  $t_1 = 2\varphi/2^{m+1}$ ,  $t_2 = (2\varphi + 1)/2^{m+1}$  and  $t_3 = (2\varphi + 2)/2^{m+1}$ .

The matrix  $B(s)$  in (1.16) has the following block structure:

$$B(s) \equiv B((m, \varphi)) = \begin{bmatrix} 0 \\ K_3 \\ 0 \end{bmatrix} \quad (3.34)$$

where  $K_3$  is any matrix such that  $K_3 K_3^T = P_{t_2|t_1, t_3}$  and, again,  $t_1 = 2\varphi/2^{m+1}$ ,  $t_2 = (2\varphi + 1)/2^{m+1}$  and  $t_3 = (2\varphi + 2)/2^{m+1}$ . The matrix  $B(s)$  in (3.34) reflects the fact that *no* noise is added to the first and third components of the state  $x(s)$ , (which are simply copied from the preceding level), while noise corresponding to the error (3.29) is added to the second. In particular, in this case the covariance of the additive noise term  $B(s)w(s)$  in (1.16) is given by:

$$\begin{aligned} \mathbf{E}\{B(s)w(s)w^T(s)B^T(s)\} &= B(s)B^T(s) \\ &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & P_{(2\varphi+1)/2^{m+1}|2\varphi/2^{m+1}, (2\varphi+2)/2^{m+1}} & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{aligned} \quad (3.35)$$

where  $s \equiv (m, \varphi)$ .

Finally, the initial covariance matrix  $P_0$  for the state at the root node of the tree is given by:

$$P_0 \equiv \mathbf{E} \left\{ \begin{bmatrix} z(0) \\ z(0.5) \\ z(1) \end{bmatrix} \begin{bmatrix} z(0) \\ z(0.5) \\ z(1) \end{bmatrix}^T \right\} \quad (3.36)$$

$$= \begin{bmatrix} \Pi_0 & \Pi_0 \Phi(1/2, 0)^T & \Pi_0 \Phi(1, 0)^T \\ \Phi(1/2, 0) \Pi_0 & \Pi_{1/2} & \Pi_{1/2} \Phi(1, 1/2)^T \\ \Phi(1, 0) \Pi_0 & \Phi(1, 1/2) \Pi_{1/2} & \Pi_1 \end{bmatrix} \quad (3.37)$$

For instance, if  $z(t)$  is the standard Brownian motion, then  $F_t = 0$ ,  $\Phi(t, \tau) = 1$ ,

and  $\Pi_t = t$ . Thus for this example (3.28) and (3.29) become:

$$\hat{z}_{t_2|t_1,t_3} = \frac{t_3 - t_2}{t_3 - t_1} z(t_1) + \frac{t_2 - t_1}{t_3 - t_1} z(t_2) \quad (3.38)$$

$$P_{t_2|t_1,t_3} = \frac{(t_2 - t_1)(t_3 - t_2)}{t_3 - t_1} \quad (3.39)$$

and we obtain:

$$A(s) \equiv A((m, \varphi)) = \begin{cases} \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1/2 & 0 \\ 0 & 1 & 0 \end{bmatrix} & \text{if } \varphi \text{ is even} \\ \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1/2 & 1/2 \\ 0 & 0 & 1 \end{bmatrix} & \text{if } \varphi \text{ is odd} \end{cases} \quad (3.40)$$

$$B(s) \equiv B((m, \varphi)) = \begin{bmatrix} 0 \\ 1/2^{(m+2)/2} \\ 0 \end{bmatrix} \quad (3.41)$$

$$P_0 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & 0.5 & 1 \end{bmatrix} \quad (3.42)$$

The formula (3.28) for the conditional expectation  $\hat{z}_{t_2|t_1,t_3}$ , which specifies  $A(s)$  as just described, also provides us with the required formula for interpolating between dyadic sample points at any level in our multiscale representation and hence provides us with a direct interpretation of this representation as providing a sequence of multiresolution approximations. For example, Brownian motion provides us with the linear interpolation formula given in (3.38) and illustrated in Figure 3-1. This corresponds to a multiresolution linear spline approximation or, as also illustrated in Figure 3-1, as a non-orthogonal multiresolution decomposition using the so-called "hat" function [127].

As a second example, consider the movement of a particle whose *velocity* is given by a Brownian motion. This motion can be described using the following Gauss-Markov process:

$$\dot{z}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} z(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mu(t) \quad (3.43)$$

In (3.43), the first component of  $z(t)$  is the particle position and the second component is its velocity. The state transition matrix  $\Phi(t, \tau)$  and the state covariance matrix  $\Pi_t$  are given by:

$$\Phi(t, \tau) = \begin{bmatrix} 1 & t - \tau \\ 0 & 1 \end{bmatrix} \quad (3.44)$$

$$\Pi_t = \begin{bmatrix} t^3/3 & t^2/2 \\ t^2/2 & t \end{bmatrix} \quad (3.45)$$

One can show by direct computation that the terms  $\Pi_{t_1} \Phi(t_2, t_1)^T$  and  $\Phi(t_3, t_2) \Pi_{t_2}$  in the leftmost block matrix on the right side of (3.28) contain only cubic powers of  $t_2$ . Note also that the block matrix in the middle of the right side does not depend on  $t_2$ . Thus, the interpolation of  $z(t_2)$  between  $t_1$  and  $t_3$  is a cubic polynomial in  $t_2$ :

$$\hat{z}_{t_2|t_1, t_3} = \begin{bmatrix} c_1 + c_2 t_2 + c_3 t_2^2 + c_4 t_2^3 \\ c_2 + 2c_3 t_2 + 3c_4 t_2^2 \end{bmatrix} \quad (3.46)$$

where from (3.43), the second component of  $\hat{z}_{t_2|t_1, t_3}$  is just the first derivative of the first. One can use (3.28) directly to calculate the values of the coefficients in (3.46). Alternatively, it is clear from the definition of  $\hat{z}_{t_2|t_1, t_3}$  in (3.25) that  $\hat{z}_{t_1|t_1, t_3} = z(t_1)$  and  $\hat{z}_{t_3|t_1, t_3} = z(t_3)$ . These two constraints provide four linear equations in the four unknown coefficients in (3.46), and thus uniquely determine the interpolating function (3.46). Note that the interpolating polynomial for the first component of the state (the position of the particle) has a continuous derivative at the knot locations  $t = k/2^{m+1}, k = 0, 1, \dots, 2^{m+1}$ . The interpolation of the first component of the state

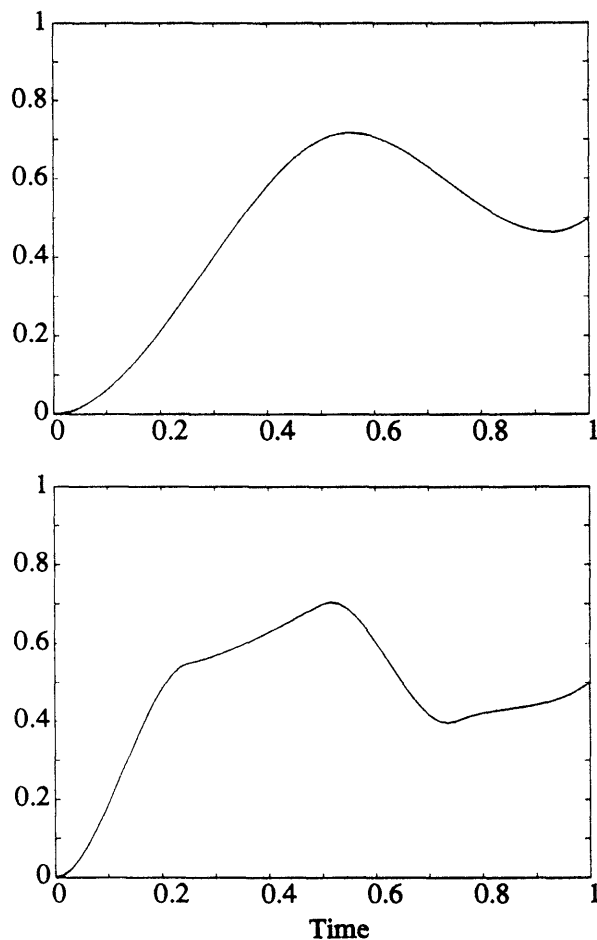


Figure 3-7: The first two scales in a multiscale representation of a process which is equal to the *second integral* of white noise are shown. The representation consists of samples of the process at dyadic points along with a piecewise-cubic interpolation. Compare these curves with the graphs of Figure 3-1, which depict the piecewise linear interpolation of the *first integral* of white noise.

is shown in Figure 3-7 for the first two levels of a sample path of the multiscale realization. The top of Figure 3-7 is the cubic interpolation the first components of  $z(0)$ ,  $z(0.5)$  and  $z(1)$ , while the bottom illustrates the cubic interpolation of the first components of  $z(0)$ ,  $z(0.25)$ ,  $z(0.5)$ ,  $z(0.75)$  and  $z(1)$ .

We make two final points before ending our discussion on the multiscale representation of Gaussian-Markov processes. First, we note that the Brownian motion process over an interval can actually be represented as a *two-state* multiscale process. We point this out not only because it is interesting in its own right, and because it suggests that by taking into account more details about the structure of the process

under consideration (i.e. by exploiting more than just reciprocity or Markovianity), elegant and parsimonious alternative representations can be obtained. Second, while we have focused on the construction of multiscale models for the class of continuous-time Gauss-Markov processes, an exactly analogous set of calculations can be performed for the discrete-time process:

$$z(t+1) = F_t z(t) + G_t \mu(t) \quad (3.47)$$

Also, as discussed in Section 3.3, in this case we can either construct models with three or four-point state vectors. The former of these is exactly analogous to what we have done here in continuous-time and has, as we have indicated, a high degree of redundancy, while the latter does not. We defer explicit discussion and illustration of such non-redundant representations until Section 3.4 where we describe them in the context of modeling 2-D MRF's.

### Finite-State Markov Processes

Next, consider a general finite-state Markov process  $z(t) \in \{1, 2, \dots, L\}$  defined over a discrete interval  $t \in \{0, 1, \dots, T\}$ . The probability structure of the process is completely determined by the initial conditions:

$$\Pr[z(0) = k] \quad (3.48)$$

for  $k \in \{1, 2, \dots, L\}$  and by the one-step transition probabilities:

$$P_{i,j} \equiv \Pr[z(t) = i | z(t-1) = j] \quad (3.49)$$

We define the one-step transition matrix:

$$P = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,L} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,L} \\ \vdots & \vdots & \ddots & \vdots \\ P_{L,1} & P_{L,2} & \cdots & P_{L,L} \end{bmatrix} \quad (3.50)$$

Note that the multistep transition probabilities are given by powers of the matrix<sup>4</sup>  $P$ :

$$\Pr[z(t + \tau) = i | z(t) = j] = [P^\tau]_{i,j} \quad (3.51)$$

Using (3.51) and Bayes' rule it is straightforward to calculate that for  $t_1 < t_2 < t_3$ :

$$\Pr[z(t_2) = j | z(t_1) = i, z(t_3) = k] = \frac{[P^{t_3-t_2}]_{k,j} [P^{t_2-t_1}]_{j,i}}{[P^{t_3-t_1}]_{k,i}} \quad (3.52)$$

These conditional probabilities, in addition to the probability function required for the state at the root node of the tree, namely

$$\Pr[z(0) = i, z(T/2) = j, z(T) = k] = [P^{T/2}]_{k,j} [P^{T/2}]_{j,i} \Pr[z(0) = i] \quad (3.53)$$

allow us to construct the multiscale representation of the process. Note that (3.52) is the counterpart of the conditional probability equations for Gauss-Markov processes given in (3.27) – (3.29), and that the pdf for the state at the root node (3.53) is the counterpart of the initial covariance matrix (3.37).

One special case of this process is the following:

$$P_{i,i} = \mu \quad (3.54)$$

$$P_{i,j} = \frac{1 - \mu}{L - 1}, \quad j \neq i \quad (3.55)$$

$$\Pr[z(0) = i] = 1/L, \quad i = 1, 2, \dots, L \quad (3.56)$$

---

<sup>4</sup> $[A]_{i,j}$  stands for the  $(i, j)$  element of the matrix  $A$ .



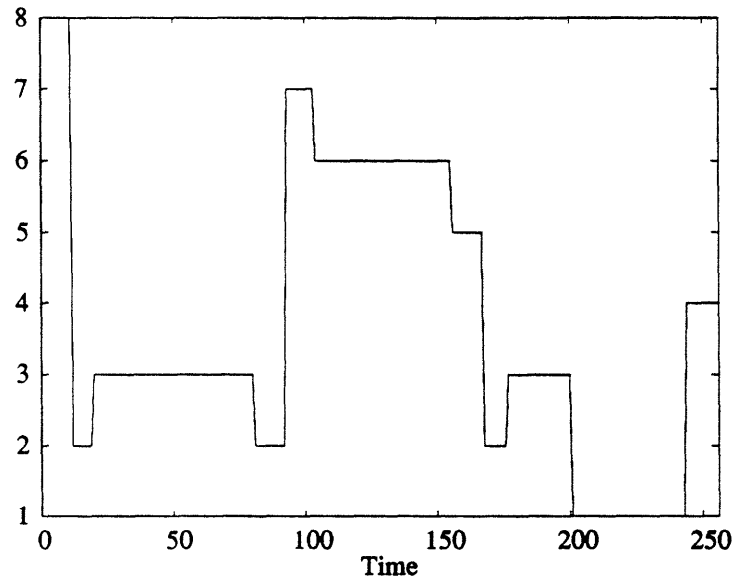


Figure 3-8: A sample path of a discrete-state Markov process is shown. Multiscale representation of this and other discrete-state processes are developed in Section 3.3.3.

A sample path of such a process is shown in Figure 3-8 for  $L = 8$  and  $\mu = 0.97$ . Neighboring states of this process tend to be the same, and when the process does change state, no particular change is preferred. Thus, this model would seem to be a natural one to use in segmentation applications and can in fact be viewed as an alternative to the 1-D multiscale model (3.3) introduced in [17, 19, 20]. As noted in Section 3.2, the model in (3.3) does *not* in general produce a Markov chain or reciprocal process at the finest level. On the other hand (3.54) – (3.56) *is* a Markov model and for this process:

$$[P^k]_{i,j} = \begin{cases} (1 + (L-1)\vartheta^k)/L & \text{if } i = j \\ (1 - \vartheta^k)/L & \text{if } i \neq j \end{cases} \quad (3.57)$$

where:

$$\vartheta = (L\mu - 1)/(L - 1) \quad (3.58)$$

The conditional probability function (3.57) can be verified by noting that, given (3.54), (3.55), the one-step transition matrix (3.50) is circulant, and thus diagonalized by the  $L \times L$  Discrete Fourier Transform matrix.

Using (3.52), for this example we can write down the transition probabilities for the mid-point deflection model. In particular, assuming that  $T$  is a power of two, we can associate the state at node  $s$  with the following values of the process:

$$x(s) \equiv x((m, \varphi)) = \begin{bmatrix} z(2\varphi T/2^{m+1}) \\ z((2\varphi T + T)/2^{m+1}) \\ z((2\varphi T + 2T)/2^{m+1}) \end{bmatrix} \quad (3.59)$$

where, as in (3.30), the pair of numbers  $(m, \varphi)$  denote the scale and horizontal shift of the node  $s$ , respectively. Thus, to generate the state at node  $s$ , given the state at the parent node  $s\bar{\gamma}$ , we require the following conditional pdf:

$$\Pr\left[z\left(\frac{2\varphi T + T}{2^{m+1}}\right) = j \mid z\left(\frac{2\varphi T}{2^{m+1}}\right) = i, z\left(\frac{2\varphi T + 2T}{2^{m+1}}\right) = k\right] = \begin{cases} \zeta_1 \zeta_1 / \zeta_2 & \text{if } i = j = k \\ \zeta_1 \xi_1 / \xi_2 & \text{if } i \neq j = k \\ \zeta_1 \xi_1 / \xi_2 & \text{if } i = j \neq k \\ \xi_1 \xi_1 / \zeta_2 & \text{if } i = k \neq j \\ \xi_1 \xi_1 / \xi_2 & \text{if } i, j, k \text{ distinct} \end{cases} \quad (3.60)$$

where:

$$\zeta_1 = (1 + (L-1)\vartheta^{T/2^{m+1}})/L \quad (3.61)$$

$$\zeta_2 = (1 + (L - 1)\vartheta^{T/2^m})/L \quad (3.62)$$

$$\xi_1 = (1 - \vartheta^{T/2^{m+1}})/L \quad (3.63)$$

$$\xi_2 = (1 - \vartheta^{T/2^m})/L \quad (3.64)$$

To gain additional insight concerning the structure of our multiscale models, consider the particular example of a stationary two-state binary process with one-step transition matrix and initial state probabilities equal to:

$$P = \begin{bmatrix} 1 - \mu & \eta \\ \mu & 1 - \eta \end{bmatrix} \quad (3.65)$$

$$\Pr[z(0) = i] = \begin{cases} \eta/(\eta + \mu) & \text{if } i = 1 \\ \mu/(\eta + \mu) & \text{if } i = 2 \end{cases} \quad (3.66)$$

For this process one can show that:

$$P^k = \frac{1}{\eta + \mu} \begin{bmatrix} \eta + \mu(1 - \eta - \mu)^k & \eta - \eta(1 - \eta - \mu)^k \\ \mu - \mu(1 - \eta - \mu)^k & \mu + \eta(1 - \eta - \mu)^k \end{bmatrix} \quad (3.67)$$

and thus using (3.67) with (3.52), (3.53) one can build multiscale representations for the class of stationary binary Markov processes. While we have focused in this example and in the previous one on processes with stationary state transition probabilities, it is straightforward to apply this construction to the representation of non-stationary discrete-state Markov processes as well, simply by choosing the conditional probability functions required in the multiscale representation correctly.

Moreover, the mid-point deflection structure can also be used to generate non-Markov processes on the tree. For instance, consider the following binary mid-point “selection” process defined over  $t \in \{0, 1, \dots, 2^N\}$  [133]:

$$\Pr[z(0) = i, z(2^{N-1}) = j, z(2^N) = k] = 1/8 \quad \text{for all } i, j, k \quad (3.68)$$

$$\Pr[z(t_2) = i | z(t_1) = j, z(t_3) = k] = \begin{cases} \mu & \text{if } i = j = k \\ 1 - \mu & \text{if } i \neq j \text{ and } j = k \\ 0.5 & \text{if } j \neq k \end{cases} \quad (3.69)$$

where  $i, j, k \in \{1, 2\}$  and where  $t_1, t_2, t_3$  are any 3-tuple of dyadic points corresponding to one of the state vectors in the multiscale representation. At the coarsest “scale” of this process, the three components of the state vector  $x_0$  are independent and identically distributed random variables, each equally likely to be 1 or 2. It is easy to show that the process resulting from this construction is *not* Markov in general, and thus we can conclude that the set of binary stochastic processes which can be constructed within the mid-point deflection framework is strictly larger than the class of binary Markov processes over intervals.

In fact, a bit of thought shows that the class of processes realizable by multiscale models is quite a bit larger than the class of Markov chains. Indeed, any binary stochastic process defined over  $t \in \{0, 1, \dots, 2^N\}$  when represented via mid-point deflection has a probability structure which is determined by  $4(2^N - 1)$  parameters, corresponding to the required conditional probability functions. In particular, the conditional probabilities  $\Pr[z(t_2) = i | z(t_1) = j, z(t_3) = k]$  for specific choices of  $t_1 < t_2 < t_3$  are uniquely determined by the four parameters:

$$\Pr[z(t_2) = 1 | z(t_1) = 1, z(t_3) = 1] = \lambda_1 \quad (3.70)$$

$$\Pr[z(t_2) = 1 | z(t_1) = 1, z(t_3) = 2] = \lambda_2 \quad (3.71)$$

$$\Pr[z(t_2) = 1 | z(t_1) = 2, z(t_3) = 1] = \lambda_3 \quad (3.72)$$

$$\Pr[z(t_2) = 1 | z(t_1) = 2, z(t_3) = 2] = \lambda_4 \quad (3.73)$$

Since the process is represented with an  $N$  level multiscale process, there are  $2^N - 2$  of these conditional densities which must be specified, corresponding to each of the nodes except the root node. However, the probability function for the state at the root node also requires four parameters, and thus the total number of parameters to be specified is  $4(2^N - 1)$ . In contrast, a non-stationary binary Markov process defined



Figure 3-9: A sample path of a 1-D Ising process is shown. The process can take on two states or *spins*, represented above by empty and filled circles.

over the time interval  $t \in \{0, 1, \dots, 2^N\}$  requires at most  $1 + 2 \times 2^N$  parameters (one corresponding to the initial probability, and 2 for each transition from  $t$  to  $t + 1$ , for  $t = 0, 1, \dots, 2^N - 1$ ). Since each of the parameters in each of these models is a probability, i.e. a number in the interval  $[0, 1]$ , we see that the set of processes arising from  $N$ -level multiscale models is in one-to-one correspondence with the  $4(2^N - 1)$ -dimensional unit cube, while the set of non-stationary Markov chains over the same length interval  $(2^N + 1)$  corresponds to the  $2(2^N + 1)$ -dimensional unit cube. Thus, for  $N > 1$ , Markov processes constitute only a “thin” subset of the entire class of binary tree processes.

### 1-D Ising Model

The Ising model was originally proposed by Ernst Ising in 1925 in the context of ferromagnetic modeling [63]. The 2-D version of the model has been widely studied in the statistical mechanics community as it is the simplest example of a Gibb’s distribution which exhibits a phase transition [96, 9, 102]. The 1-D version of the model describes a binary random function defined over an interval  $\{0, 1, \dots, N - 1\}$  and taking on the values 1 or  $-1$ . A sample path of the Ising model is shown in Figure 3-9, with the symbols  $\bullet$  and  $\circ$  corresponding to 1 and  $-1$ , respectively.

The probability structure of the Ising model is characterized by an energy function or *Hamiltonian*  $H(Z)$ :

$$p_z(Z) = \frac{\exp\{-H(Z)\}}{\Theta} \quad (3.74)$$

where  $Z \equiv \{Z_t\}_{t=0}^{N-1}$  and  $\Theta$  is called the *partition function*.<sup>5</sup> The Hamiltonian is a sum of local energy terms which depend on neighboring pairs of sites.

Two slightly different formulations of the Ising model, which differ in the way the boundary values are treated, are commonly encountered [96]. In the cyclic model, the sites at  $t = 0$  and  $t = N - 1$  are considered to be neighbors, so that the interaction energy due to the pair  $(Z_0, Z_{N-1})$  is the same as the interaction energy for any other two neighbors  $(Z_t, Z_{t+1})$ :

$$H_c(Z) = - \sum_{t=0}^{N-1} \alpha Z_t Z_{t+1} - \sum_{t=0}^{N-1} \beta Z_t \quad (3.75)$$

where the indices are interpreted modulo  $N$ , so that  $Z_N \equiv Z_0$ .

In the free boundary model, there is no contribution to the Hamiltonian from an interaction between  $Z_0$  and  $Z_{N-1}$ :

$$H_f(Z) = - \sum_{t=0}^{N-2} \alpha Z_t Z_{t+1} - \sum_{t=0}^{N-1} \beta Z_t \quad (3.76)$$

In either case, the partition functions in the cyclic and free boundary models, which we denote  $\Theta_c$  and  $\Theta_f$  and which depend on the model parameters  $N, \alpha$  and  $\beta$ , normalize the probability function so that it sums to one:

$$\begin{aligned} \Theta &= \sum_{Z_0 \in \{-1,1\}} \sum_{Z_1 \in \{-1,1\}} \cdots \sum_{Z_{N-1} \in \{-1,1\}} \exp\{-H(Z)\} \\ &= \sum_{Z_t, t \in [0, N-1]} \exp\{-H(Z)\} \end{aligned} \quad (3.77)$$

where we define  $[i, j] \equiv \{i, i + 1, \dots, j\}$ . We also use the notation  $(i, j) \equiv [i + 1, j - 1]$  below.

The 1-D Ising model is one of the few Gibb's distributions for which the partition function can be calculated exactly. We repeat this calculation here because it will turn out that a modification of this approach can be used to calculate the conditional

---

<sup>5</sup>The partition function is normally denoted with the symbol  $Z$ ; we use  $\Theta$  to avoid confusion with the process  $Z(t)$ .

probability functions of the form (3.14) – (3.15) that we will use to develop a multiscale representation of the Ising model.

Consider first the calculation of the partition function for the cyclic model. The idea is to associate the two values that  $Z_t$  can take on with a basis for a 2-D vector space. In particular, to each random variable  $Z_t$  we associate a vector  $\bar{Z}_t$  as follows:

$$Z_t = 1 \leftrightarrow \bar{Z}_t = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (3.78)$$

$$Z_t = -1 \leftrightarrow \bar{Z}_t = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (3.79)$$

Next, if we write the Hamiltonian for the cyclic model in the symmetric form:

$$H_c(Z) = - \sum_{t=0}^{N-1} \alpha Z_t Z_{t+1} + \frac{1}{2} \beta (Z_t + Z_{t+1}) \quad (3.80)$$

then we can associate the summand with the contribution to the Hamiltonian from the pair  $(Z_t, Z_{t+1})$ . Next, define a matrix  $\mathbf{V}$  such that:

$$\bar{Z}_t^T \mathbf{V} \bar{Z}_{t+1}^T = \exp\{\alpha Z_t Z_{t+1} + \frac{1}{2} \beta (Z_t + Z_{t+1})\} \quad (3.81)$$

Written explicitly, the matrix  $\mathbf{V}$  is given by:

$$\begin{aligned} \mathbf{V} &= \begin{bmatrix} V(1,1) & V(1,-1) \\ V(-1,1) & V(-1,-1) \end{bmatrix} \\ &= \begin{bmatrix} e^{\alpha+\beta} & e^{-\alpha} \\ e^{-\alpha} & e^{\alpha-\beta} \end{bmatrix} \end{aligned} \quad (3.82)$$

where:

$$V(Z, Z') \equiv \exp\{\alpha Z Z' + \frac{1}{2} \beta (Z + Z')\} \quad (3.83)$$

The summation in (3.77) for the cyclic model is then given by:

$$\Theta_C = \sum_{Z_t, I \in \{0, N-1\}} V(Z_0, Z_1) V(Z_1, Z_2) \cdots V(Z_{N-1}, Z_0) \quad (3.84)$$

$$\begin{aligned} &= \sum_{Z_t, I \in \{0, N-1\}} \bar{Z}_0^T \mathbf{V} \bar{Z}_1 \bar{Z}_1^T \mathbf{V} \bar{Z}_2 \cdots \bar{Z}_{N-1}^T \mathbf{V} \bar{Z}_0 \\ &= \text{trace } \mathbf{V}^N \\ &= \lambda_1^N - \lambda_2^N \end{aligned} \quad (3.85)$$

where the eigenvalues  $\lambda_1$  and  $\lambda_2$  of the matrix  $\mathbf{V}$  are given by:

$$\lambda_1 = e^\alpha \cosh \beta + (e^{2\alpha} \sinh^2 \beta + e^{-2\alpha})^{1/2} \quad (3.86)$$

$$\lambda_2 = e^\alpha \cosh \beta - (e^{2\alpha} \sinh^2 \beta + e^{-2\alpha})^{1/2} \quad (3.87)$$

and where we have used the fact that:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \sum_{Z_t \in \{-1, 1\}} \bar{Z}_t \bar{Z}_t^T \quad (3.88)$$

Similarly, in the case of free boundary conditions, the Hamiltonian can be written:

$$H_f(Z) = -\frac{1}{2}\beta Z_0 - \frac{1}{2}\beta Z_{N-1} - \sum_{t=0}^{N-2} \alpha Z_t Z_{t+1} + \frac{1}{2}\beta(Z_t + Z_{t+1}) \quad (3.89)$$

The partition function calculation requires a bit more work in this case because the boundary values  $Z_0$  and  $Z_{N-1}$  must be specially treated. In particular, we define a vector  $\mathbf{v}$  which allows us to represent the contribution of the boundary points to the Hamiltonian (without including a term corresponding to an energy between them):

$$\mathbf{v}^T \bar{Z}_t = \exp\left\{\frac{1}{2}\beta Z_t\right\}, \quad \text{for } t = 0 \text{ or } N-1 \quad (3.90)$$



Written explicitly, the vector  $\mathbf{v}$  is given by:

$$\mathbf{v} = \begin{bmatrix} \exp\{\frac{1}{2}\beta\} \\ \exp\{-\frac{1}{2}\beta\} \end{bmatrix} \quad (3.91)$$

Then the summation in (3.77) becomes:

$$\begin{aligned} \Theta_f &= \sum_{\mathbf{z}_i, i \in [0, N-1]} \exp\{\frac{1}{2}\beta Z_0\} V(Z_0, Z_1) \cdots V(Z_{N-2}, Z_{N-1}) \exp\{\frac{1}{2}\beta Z_{N-1}\} \quad (3.92) \\ &= \sum_{\mathbf{z}_i, i \in [0, N-1]} \mathbf{v}^T \bar{Z}_0 \bar{Z}_0^T \mathbf{V} \bar{Z}_1 \bar{Z}_1^T \mathbf{V} \bar{Z}_2 \cdots \bar{Z}_{N-2}^T \mathbf{V} \bar{Z}_{N-1} \bar{Z}_{N-1}^T \mathbf{v} \\ &= \mathbf{v}^T \mathbf{V}^{N-1} \mathbf{v} \end{aligned} \quad (3.93)$$

We can evaluate (3.93) by using the following eigen-decomposition of  $\mathbf{V}$ :

$$\mathbf{V} = \mathbf{U}^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \mathbf{U} \quad (3.94)$$

$$\mathbf{U} = \begin{bmatrix} 1 & e^{\alpha(\lambda_2 - e^{\alpha-\beta})} \\ e^{\alpha(\lambda_1 - e^{\alpha+\beta})} & 1 \end{bmatrix} \quad (3.95)$$

which yields:

$$\begin{aligned} \Theta_f &= [1 - e^{2\alpha}(\lambda_1 - e^{\alpha+\beta})(\lambda_2 - e^{\alpha-\beta})]^{-1} \\ &\quad \times \{ \lambda_1^{N-1} [e^{\alpha+(1/2)\beta}(e^{\alpha-\beta} - \lambda_1) - e^{-(1/2)\beta}]^2 \\ &\quad + \lambda_2^{N-1} [e^{\alpha-(1/2)\beta}(e^{\alpha+\beta} - \lambda_2) - e^{(1/2)\beta}]^2 \} \end{aligned} \quad (3.96)$$

Now, in the context of multiscale representations, we need to calculate probabilities of the type  $p_{z(j)|z(i), z(k)}(Z_j | Z_i, Z_k)$  where  $i < j < k$ . Using the fact that  $Z_t$  is a reciprocal process it is clear that (3.97) below holds. That is, from (3.6) we see that the conditional distribution of  $Z_j$  given  $Z_i, Z_k$  is the same as the conditional distribution of  $Z_j$  given  $Z_l, l \in \{1, \dots, i, k, \dots, N-1\} \equiv (i, k)^c$ . Then, (3.98) is just an application of the conditional probability rule  $\Pr(A|B) = \Pr(A, B)/\Pr(B)$ . Next, to obtain (3.99), note that the sets  $\{j\} \cup (i, k)^c$  and  $[0, N-1] \setminus \{(i, j) \cup (j, k)\}$  are

equivalent. Thus, the joint probability for  $Z_l, l \in \{j\} \cup (i, k)^c$  can be obtained by starting with the joint probability for  $Z_l, l \in [0, N - 1]$ , which is given by (3.74), and summing out over  $\{(i, j) \cup (j, k)\}$ . This leads to the numerator of (3.99), and a similar analysis gives the denominator. Finally, (3.100) is obtained by substituting the summand of (3.84) or (3.92) for the joint pdf of  $Z_l, l \in [0, N - 1]$ , and canceling common factors. Note in particular that all factors containing only terms in  $(i, k)^c$  cancel.

$$p_{z(j)|z(i),z(k)}(Z_j|Z_i, Z_k) = p_{z(j)|z(l),l \in (i,k)^c}(Z_j|Z_l, l \in (i, k)^c) \quad (3.97)$$

$$= \frac{p_{z(j),z(l),l \in (i,k)^c}(Z_j, Z_l, l \in (i, k)^c)}{p_{z(l),l \in (i,k)^c}(Z_l, l \in (i, k)^c)} \quad (3.98)$$

$$= \frac{\sum_{Z_l, l \in (i,j) \cup (j,k)} p_{z(t),t \in [0,N-1]}(Z_t, t \in [0, N - 1])}{\sum_{Z_l, l \in (i,k)} p_{z(t),t \in [0,N-1]}(Z_t, t \in [0, N - 1])} \quad (3.99)$$

$$= \frac{\sum_{Z_l, l \in (i,j) \cup (j,k)} V(Z_i, Z_{i+1}) \cdots V(Z_{k-1}, Z_k)}{\sum_{Z_l, l \in (i,k)} V(Z_i, Z_{i+1}) \cdots V(Z_{k-1}, Z_k)} \quad (3.100)$$

Note also that (3.100) does not depend on the boundary condition. This is due to the obvious fact that the boundary values are never contained in the interval  $(i, k)$ . What the boundary condition *does* affect is the probability calculation at the root node. This root node calculation is closely related to (3.100), and hence we focus on (3.100) and simply state the result for the root node calculation. From (3.100), our main problem is clear. We need to perform summations which are similar in nature to those required to compute the partition function, with the only difference being that we do not want to sum out over everything in the interval.

There are two cases of interest. The first case is  $j - i = k - j = 1$ , i.e. the nearest neighbor probability. In this case the summations in (3.100) are easy to compute. Indeed, nothing needs to be done in the numerator and we only need to sum over  $Z_j$

in the denominator. The well-known result is:

$$p_{z(j)|z(j-1),z(j+1)}(Z_j|Z_{j-1}, Z_{j+1}) = \frac{\exp\{\alpha(Z_{j-1}Z_j + Z_jZ_{j+1}) + \beta Z_j\}}{\cosh(\alpha(Z_{j-1} + Z_{j+1}) + \beta)} \quad (3.101)$$

The second case is the situation in which  $j - i > 1$  and  $k - j > 1$ , i.e. situations in which there is a non-trivial summation to be done in the numerator of<sup>6</sup> (3.100). A modified version of the transfer matrix approach defined above can be used in this case. In particular, let us define the vector  $\mathbf{w}_t$  such that:

$$\mathbf{w}_t^T \bar{Z}_{t+1} = V(Z_t, Z_{t+1}) \quad (3.102)$$

Written explicitly,

$$\mathbf{w}_t = \begin{bmatrix} \exp\{\alpha Z_t + \frac{1}{2}\beta(Z_t + 1)\} \\ \exp\{-\alpha Z_t + \frac{1}{2}\beta(Z_t - 1)\} \end{bmatrix} \quad (3.103)$$

Then, using (3.81), (3.100) and (3.102):

$$\begin{aligned} p_{z(j)|z(i),z(k)}(Z_j|Z_i, Z_k) &= \frac{\sum_{Z_1, l \in (i,j) \cup (j,k)} \mathbf{w}_i^T \bar{Z}_{i+1} \cdots \bar{Z}_{j-1} \mathbf{w}_j \mathbf{w}_j^T \bar{Z}_{j+1}^T \cdots \bar{Z}_{k-1} \mathbf{w}_k}{\sum_{Z_1, l \in (i,k)} \mathbf{w}_i^T \bar{Z}_{i+1} \cdots \bar{Z}_{k-1} \mathbf{w}_k} \\ &= \frac{\mathbf{w}_i^T \mathbf{V}^{j-i-2} \mathbf{w}_j \mathbf{w}_j^T \mathbf{V}^{k-j-2} \mathbf{w}_k}{\mathbf{w}_i^T \mathbf{V}^{k-i-2} \mathbf{w}_k} \\ &= w(i, j)w(j, k)/w(i, k) \end{aligned} \quad (3.104)$$

where:

$$\begin{aligned} w(m, n) &= \mathbf{w}_m^T \mathbf{U}^{-1} \begin{bmatrix} \lambda_1^{m-n-2} & 0 \\ 0 & \lambda_2^{m-n-2} \end{bmatrix} \mathbf{U} \mathbf{w}_n \\ &= [1 - e^{2\alpha}(\lambda_1 - e^{\alpha+\beta})(\lambda_2 - e^{\alpha-\beta})]^{-1} \\ &\quad \times \{\lambda_1^{m-n-2} [e^{\alpha Z_m + (1/2)\beta(Z_m+1)} - (\lambda_1 - e^{\alpha+\beta})e^{((1/2)\beta-\alpha)(Z_m-1)}] \\ &\quad \times [e^{\alpha Z_n + (1/2)\beta(Z_n+1)} + (\lambda_2 - e^{\alpha-\beta})e^{((1/2)\beta-\alpha)(Z_n-1)}] \end{aligned}$$

<sup>6</sup>The case in which  $j - i > 1$  and  $k - j = 1$  (or in which  $j - i = 1$  and  $k - j > 1$ ) is a simple combination of the two cases we discuss.

$$\begin{aligned}
& + \lambda_2^{m-n-2} [e^{-\alpha Z_m + (1/2)\beta(Z_m-1)} - (\lambda_2 - e^{\alpha-\beta})e^{((1/2)\beta+\alpha)(Z_m+1)}] \\
& \times [e^{-\alpha Z_n + (1/2)\beta(Z_n-1)} + (\lambda_1 - e^{\alpha+\beta})e^{((1/2)\beta+\alpha)(Z_n+1)}] \quad (3.105)
\end{aligned}$$

Using analysis similar to the above, we can show that the root node probability in the cyclic case is given by:

$$\begin{aligned}
& p_{z(0),z(N/2),z(N-1)}(Z_0, Z_{N/2}, Z_{N-1}) = \\
& \mathbf{w}_0^T \mathbf{V}^{(N-4)/2} \mathbf{w}_{N/2} \mathbf{w}_{N/2}^T \mathbf{V}^{(N-4)/2} \mathbf{w}_{N-1} \exp\{\alpha Z_0 Z_{N-1} + \frac{1}{2}\beta(Z_0 + Z_{N-1})\} / \Theta_c \quad (3.106)
\end{aligned}$$

and in the free boundary case by:

$$\begin{aligned}
& p_{z(0),z(N/2),z(N-1)}(Z_0, Z_{N/2}, Z_{N-1}) = \\
& \mathbf{w}_0^T \mathbf{V}^{(N-4)/2} \mathbf{w}_{N/2} \mathbf{w}_{N/2}^T \mathbf{V}^{(N-4)/2} \mathbf{w}_{N-1} \exp\{\frac{1}{2}\beta(Z_0 + Z_{N-1})\} / \Theta_f \quad (3.107)
\end{aligned}$$

Together, (3.104) – (3.107) can be used to construct a multiscale representation of the Ising process, with free or cyclic boundary conditions.

## 3.4 Representation of 2-D Markov Random Fields

In this section we first review a few of the properties of Markov random fields and then describe how Markov random fields can be represented exactly with our multiscale modeling structure. Next, we introduce a family of approximate representations for Gaussian MRF's employing 1-D wavelet transforms.

### 3.4.1 2-D Markov Random Fields

Markov random fields (MRF's) are a multidimensional generalization of 1-D reciprocal processes. A continuous space stochastic process  $z(t), t \in \mathcal{R}^n$  is said to be a Markov random field if the probability distribution of the process in the interior of any closed set  $\Omega$  is independent of the process outside, conditioned on the values of  $z(t)$  on the

boundary  $\Gamma$  of  $\Omega$ . That is, for  $t \in \Omega \setminus \Gamma$ :

$$p_{z(t)|z(\tau), \tau \in (\Omega \setminus \Gamma)^c}(Z_t | Z_\tau, \tau \in (\Omega \setminus \Gamma)^c) = p_{z(t)|z(\tau), \tau \in \Gamma}(Z_t | Z_\tau, \tau \in \Gamma) \quad (3.108)$$

where the notation  $\Omega \setminus \Gamma$  denotes the set of elements in  $\Omega$  which are not in  $\Gamma$  (in this case, the interior of  $\Omega$ ). The definition for Markov random fields on discrete lattices requires the specification of the notion of the “boundary” of a set in  $\mathcal{Z}^n$  [148, 43]. Typically, this is accomplished through the specification of a *neighborhood system*. The idea is that the probability distribution of  $z(t)$ , conditioned on a set  $\{z(\tau), \tau \in D_t\}$  in the neighborhood,  $D_t$ , of  $t$ , is independent of the process outside the neighborhood:

$$p_{z(t)|z(\tau), \tau \in \mathcal{Z}^n \setminus \{t\}}(Z_t | Z_\tau, \tau \in \mathcal{Z}^n \setminus \{t\}) = p_{z(t)|z(\tau), \tau \in D_t}(Z_t | Z_\tau, \tau \in D_t) \quad (3.109)$$

We will focus on 2-D MRF's, i.e.  $t \in \mathcal{Z}^2$ , and in this context there is a hierarchical sequence of neighborhoods frequently used in image processing applications [23]. The *first order* neighborhood of a lattice point consists of its four nearest neighbors, and the *second-order* neighborhood consists of its eight nearest neighbors. The sequence of neighborhoods up to order seven is illustrated in Figure 3-10.

A given neighborhood system implicitly determines the boundary set of any particular region. In particular, given the neighborhood system  $D_t, t \in \mathcal{Z}^2$ , the boundary  $\Gamma$  of a subset  $\Omega$  of  $\mathcal{Z}^2$  is given by the set of points which are neighbors of elements in  $\Omega$ , but not elements of  $\Omega$ :

$$\Gamma = \{\tau | \tau \in D_t, t \in \Omega\} \setminus \Omega \quad (3.110)$$

The conditional distribution and neighborhood structure cannot be chosen arbitrarily if one is to obtain a consistent joint distribution for the elements of the random field. First, the neighborhood system must have the properties that  $t \notin D_t$  and (2) if

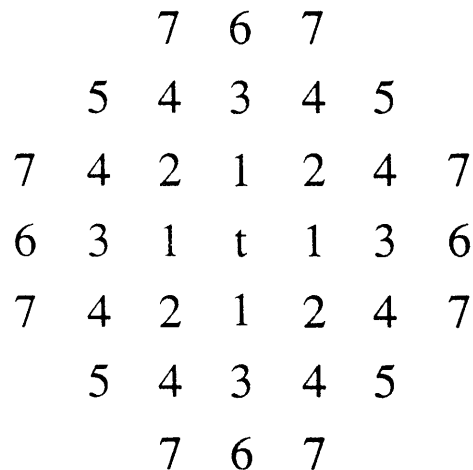


Figure 3-10: The first through seventh-order neighborhoods of lattice site  $t$  are shown. The first-order neighborhood consists of just the two vertical and two horizontal nearest neighbors.

$t \in D_\tau$  then  $\tau \in D_t$ . Second, the conditional distribution functions must satisfy the consistency conditions given by the Hammersley-Clifford theorem [13]. For detailed accounts of these issues and MRF's in general, we refer the reader to a few of the widely referenced papers in the field [47, 113, 126, 13, 148, 71, 53, 43].

### 3.4.2 Exact Representations of 2-D Markov Random Fields

The representations of 1-D reciprocal and Markov processes in Section 3.3 relied on the conditional independence of regions inside and outside a boundary set, and we use the same idea here to represent Markov random fields on a square lattice. The only change is that we now use multiscale models defined on the quadtree shown in Figure 1-2.

Consider a 2-D MRF  $z(t)$  defined on a  $(2^N + 1) \times (2^N + 1)$  lattice. The construction of reciprocal processes in one-dimension started with the values of the process at the initial, middle and end points of an interval. In two dimensions, the analogous top level description consists of the values of the MRF around the outer boundary of the lattice and along the vertical and horizontal "mid-lines" which divide the lattice into four quadrants of equal size. For instance, on a  $17 \times 17$  lattice, the state vector  $x_0$  at the root node of the quadtree contains the values of the MRF at the shaded boundary

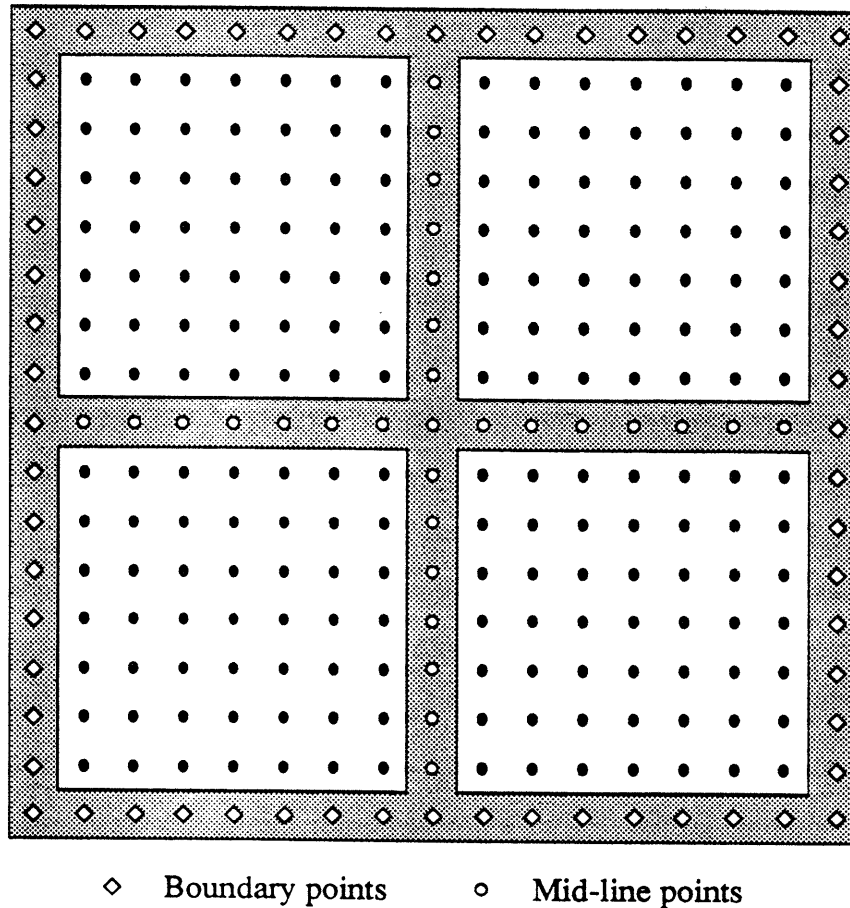


Figure 3-11: The state vector at the root node in the MRF multiscale representation consists of the MRF values at the boundary and “mid-line” points, shown in the shaded region here for a  $17 \times 17$  lattice. To construct a sample path of the MRF using the “mid-line” deflection construction, we start by choosing a sample from the joint distribution of the values in the root node state.

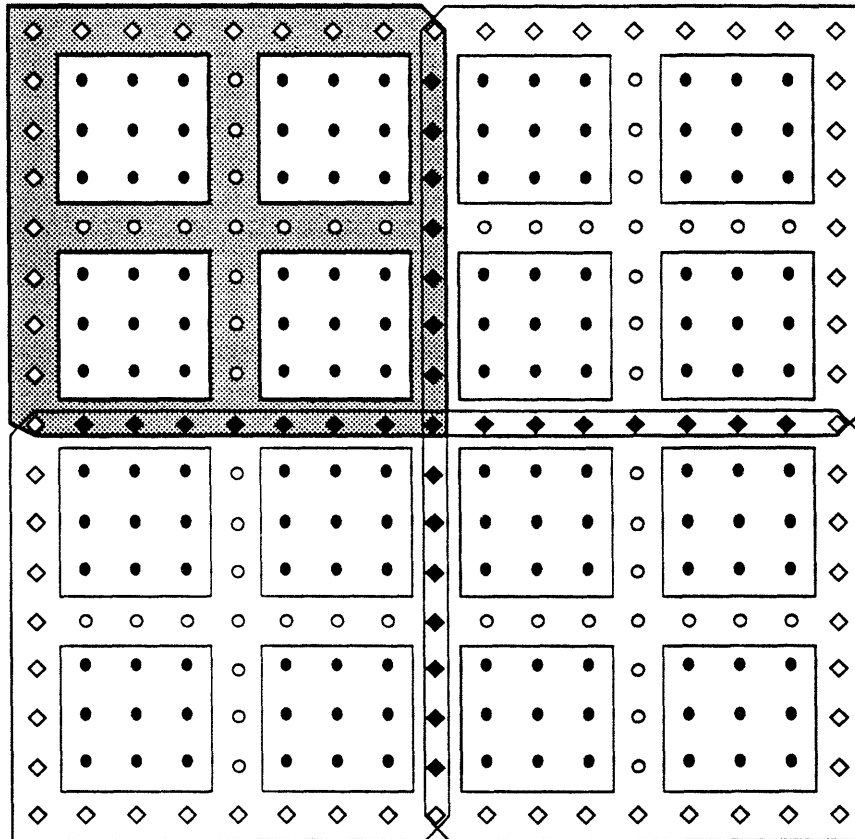
and mid-line points shown in Figure 3-11. The boundary points are denoted with  $\diamond$  and  $\circ$  symbols, respectively. In general, the state at the root node is a  $(6 \times 2^N - 3)$ -dimensional vector (given some ordering of the boundary and mid-line lattice points). To construct a sample path of the MRF, we begin by choosing a sample from the joint pdf of the MRF values defined on the boundary and mid-line set. This is the 2-D counterpart to choosing a sample from the pdf in (3.14) when constructing a 1-D reciprocal process.

In the 1-D case, transitions from the first to second level consisted of obtaining a sample from the conditional distribution of the state at the mid-points of the left and

right half-intervals. In two dimensions, we predict the set of values at the mid-lines in each of the four quadrants. The components of the four state vectors at the second level of the quadtree are illustrated in Figure 3-12 for the  $17 \times 17$  MRF. The points corresponding to the state in the north-west corner are shaded, and correspond to a scaled and shifted version of the points at the top level. The boundary points of the north-west state are denoted with open and blackened diamond symbols and the new mid-line points are denoted with open circles. Note that the four states at the second level share the black diamond mid-line points of the state at the first level. This is analogous to the 1-D construction in which the mid-point at the first level corresponds to an end point in *both* states at the second level (cf. Figure 3-2). In general, the state vectors at the four nodes at the second level each specify the MRF at  $6 \times 2^{N-1} - 3$  lattice points. Each of these states at the second level consists of points carried down from the root node (namely the diamond boundaries of each of the quadrants in Figure 3-12) as well as new mid-line points within each quadrant (the open circles in Figure 3-12). These mid-line values are chosen as samples from their joint conditional distribution, given the state at the root node. The key point here is that given the values of the field around the boundary of each quadrant, the values of the field along the mid-lines of that quadrant are *independent* of the values outside this quadrant. Said another way, the four states at the second level of the tree are conditionally independent given the values of the MRF on their respective boundaries, i.e. given precisely that information captured in the state at the first level. Thus, the values along the new mid-lines at the second level can be chosen independently and in parallel, in analogy to the way the two mid-points in (3.15), (3.16) are chosen.

Now, we can iterate the construction by defining the states at successive levels to be the values of the MRF at boundary and mid-line points of successively smaller subregions. Indeed, by subdividing each quadrant in the same way as we did in going from the first level to the second, at the  $m^{\text{th}}$  level the  $4^m$  state vectors each contain the values of the MRF at  $6 \times 2^{N-m} - 3$  boundary and mid-line points. Note that the dimension of the state varies from level to level, reflecting the obvious fact that





- ◇ Boundary points at both first and second levels
- ◆ Boundary points at the second level and mid-line points at the first level
- New (second level) mid-line points

Figure 3-12: The components of the four state vectors at the second level of the tree are scaled and shifted versions of the components of the state at the root node. For instance, the state corresponding to the north-west corner at the second level of a representation for an MRF defined on a  $17 \times 17$  lattice consists of the values of the process at the shaded points. The values of the MRF at the boundary points in these second level states are mapped down from the root node state, and the values at the new mid-lines in each of the four quadrants are chosen independently. In particular, the new mid-line values in any given quadrant are independent of values of the MRF outside that quadrant, given the boundary. Thus, in the construction of a sample path, we can choose values along each of the four sets of new mid-lines independently and in parallel. This process can then be iterated, by defining the states of the multiscale process at lower levels in the quadtree with respect to successively smaller subdomains, and constructing the process (along boundary and mid-line points) independently within each subdomain.

the number of points in the boundary of a 2-D region depends on the size of the region. The multiscale representation is defined at scales  $0, 1, \dots, N - 1$ , and each of the  $4^N$  states at scale  $N - 1$  represents 9 values in a  $3 \times 3$  square. Because of the Markov property, at each level the states are conditionally independent, given their parent state at the next higher level. Thus, the MRF can be thought of precisely as a multiscale stochastic process, and, in the Gaussian case, this leads to models *exactly* as in (1.16).

As in the 1-D case, there are several comments to make. First, we have described a construction in which the lattice is square. If the MRF is defined over a non-square lattice, then the same basic approach will work. In particular, all we require is some sequence of subregions whose boundaries eventually become dense in the set of lattice points. Second, just as our 1-D multiscale model has a natural interpretation in terms of decimation — e.g. if the points on the finest scale correspond to integers, i.e. to  $\mathcal{Z}$ , then at the next most fine scale they correspond to even integers, i.e.  $2\mathcal{Z}$  — so does our 2-D model, although it differs from the usual notion of decimation in 2-D. Specifically, if the points on the finest scale correspond to  $\mathcal{Z}^2 = \mathcal{Z} \times \mathcal{Z}$ , then the usual notion of decimation would be  $2\mathcal{Z} \times 2\mathcal{Z}$ . In contrast, the notion of decimation associated with our multiscale models yields the set  $(2\mathcal{Z} \times \mathcal{Z}) \cup (\mathcal{Z} \times 2\mathcal{Z})$  at the next finest scale.

Indeed, the obvious difference between our multiscale MRF representations and those of [61, 54, 77] is that these latter representations *do* correspond to multiscale representations using the usual notion of decimation. That is, the usual decimation leads to representations of the field at coarser levels which correspond roughly to 2-D lowpass filtered and subsampled versions of that at the finest level. Hence, the interpolating functions which generate a process at the finest level from a coarse scale sample naturally correspond to 2-D Haar scaling functions or more generally to localized interpolation operators such as those commonly used for coarse-to-fine grid transfer in multigrid applications. In contrast, the interpolation functions in our representation naturally correspond, in the case of Gaussian MRF's, to the solutions of specific differential (or partial differential) equations determined by the covariance

structure of the process. To see this more clearly, note that the linear spline interpolation formula for Brownian motion given values at two points  $z(0) = Z_0$  and  $z(T) = Z_T$  is simply the solution to the second-order differential equation:

$$\frac{d^2}{dt^2} \hat{z}_{t|0,T} = 0 \quad (3.111)$$

Similarly the interpolation of the first component of the second-order process (3.43) is given by the solution of:

$$\frac{d^4}{dt^4} \hat{z}_{t|0,T} = 0 \quad (3.112)$$

given  $z(0), \dot{z}(0), z(T)$  and  $\dot{z}(T)$ . The 2-D example analogous to the linear spline model for Brownian motion is Laplace's equation:

$$\nabla^2 \hat{z} = 0 \quad (3.113)$$

given values of  $z$  on the boundary of a square region, while the counterpart to (3.112), corresponding to a second-order model, would be the solution of a homogeneous biharmonic equation:

$$\nabla^4 \hat{z} = 0 \quad (3.114)$$

given boundary values and normal derivatives along the boundary. (see, for example [146, 79], for related discussions).

Third, note that the representation we have defined is redundant. In the 1-D case this redundancy was removed by predicting two mid-points at each level instead of one (cf. Figure 3-5). Likewise, in two dimensions we can eliminate the redundancy by predicting *two* mid-lines at each level instead of one. For instance, the state vector at the top level of the quadtree for an MRF defined on a  $16 \times 16$  lattice would consist of the values of the MRF at the shaded points shown in Figure 3-13, and the four state vectors at the next level would consist of the values of the process at the points shaded

in Figure 3-14. In the 2-D case this latter representation is important in the context of defining approximate models, as will be seen in the next section, and illustrated in Section 3.5.

Fourth, higher-order models can be easily accommodated. For example, if the Markov random field has up to a fifth-order neighborhood (cf. Figure 3-10), the field can be represented by taking as state the values of the process along boundaries and mid-lines of "width" two. In general, neighborhoods of any order can be handled by increasing the boundary and mid-line widths appropriately.

Finally, we note that for domains of substantial size, the representations may be of prohibitively large dimension. This issue is addressed for Gaussian MRF's in the next section, where we introduce a family of low-dimensional *approximate* representations based on *one-dimensional* wavelet transforms of the MRF along 1-D boundaries.

### 3.4.3 Approximate Representation of 2-D Gaussian MRF's

In this section we propose a family of approximate representations for Gaussian MRF's that provide low-dimensional alternatives to the exact multiscale representations. The basic idea behind the approximate representations is to take as the state not boundaries of regions, but rather some reduced-order representation of them. Conceptually, we would like to retain only those components of the boundary that are required to maintain nearly complete conditional independence of regions. In general, exact conditional independence will be lost unless the entire boundary is kept, but as we discuss and illustrate here and in the next section, in many cases only a small amount of information needs to be retained in order to obtain adequate representations of the important statistical and qualitative features of a Gaussian MRF.

The basis for our approximation methodology is a change of coordinates in representing the values of MRF's along 1-D boundaries. A family of models can then be generated by making different choices for the set of coordinates to be retained and those to be discarded at each level of the multiscale representation. These models range from being exact (if all coordinates are retained) to increasingly approximate

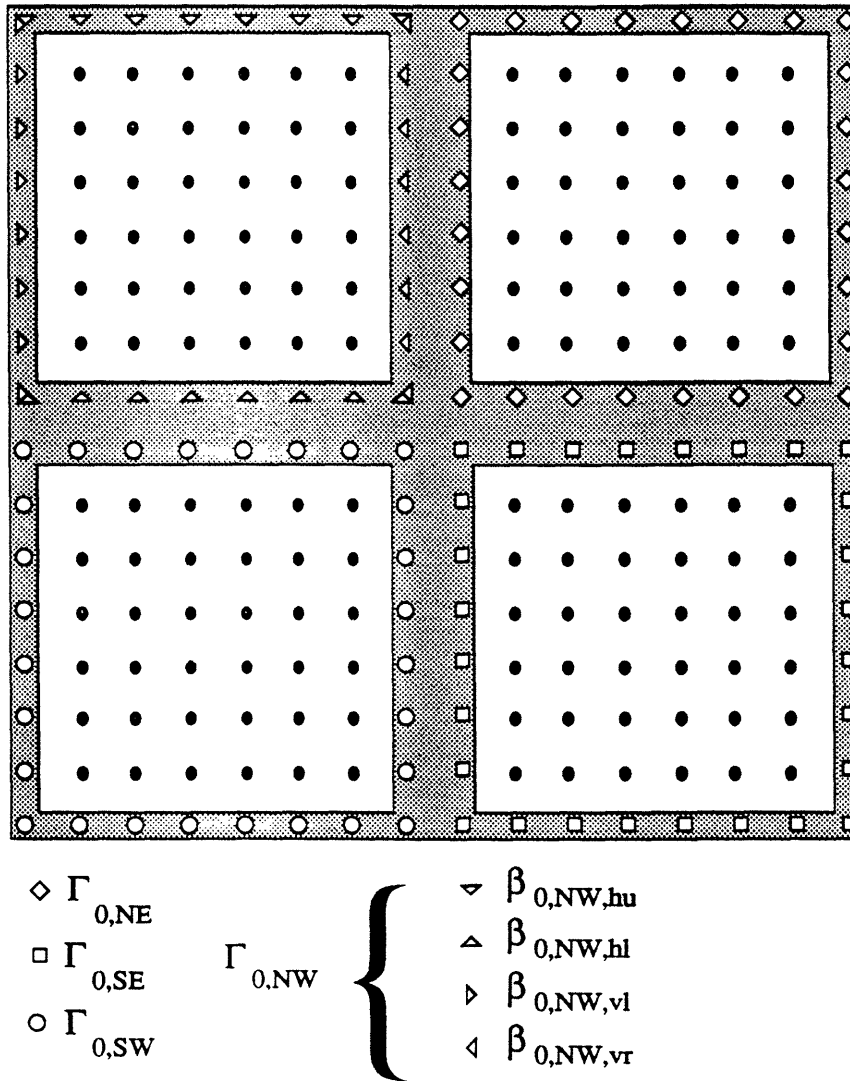


Figure 3-13: The state at the root node in a *non-redundant* exact multiscale representation of an MRF defined on a  $16 \times 16$  lattice consists of the values of the process at the shaded points. The redundancy in the exact representation is eliminated by generating the values of the process along two mid-lines instead of one. The figure also illustrates the sets  $\Gamma_{s,i}$ , and the sequences  $\beta_{s,i,j}(k)$  defined in the context of approximate representations in Section 3.4. The  $\beta_{s,i,j}(k)$  are 1-D sequences corresponding to values of the MRF along boundaries of square subdomains (which, at the first level, are the white areas in the figure). These sequences overlap at the corner points of boundaries. In the figure, this is represented by putting two symbols at the same lattice point, e.g.  $\nabla$  and  $\triangleright$  in the upper left corner. The approximate representations take as the state subsets of the coefficients in 1-D wavelet expansions of the  $\beta_{s,i,j}(k)$  sequences.

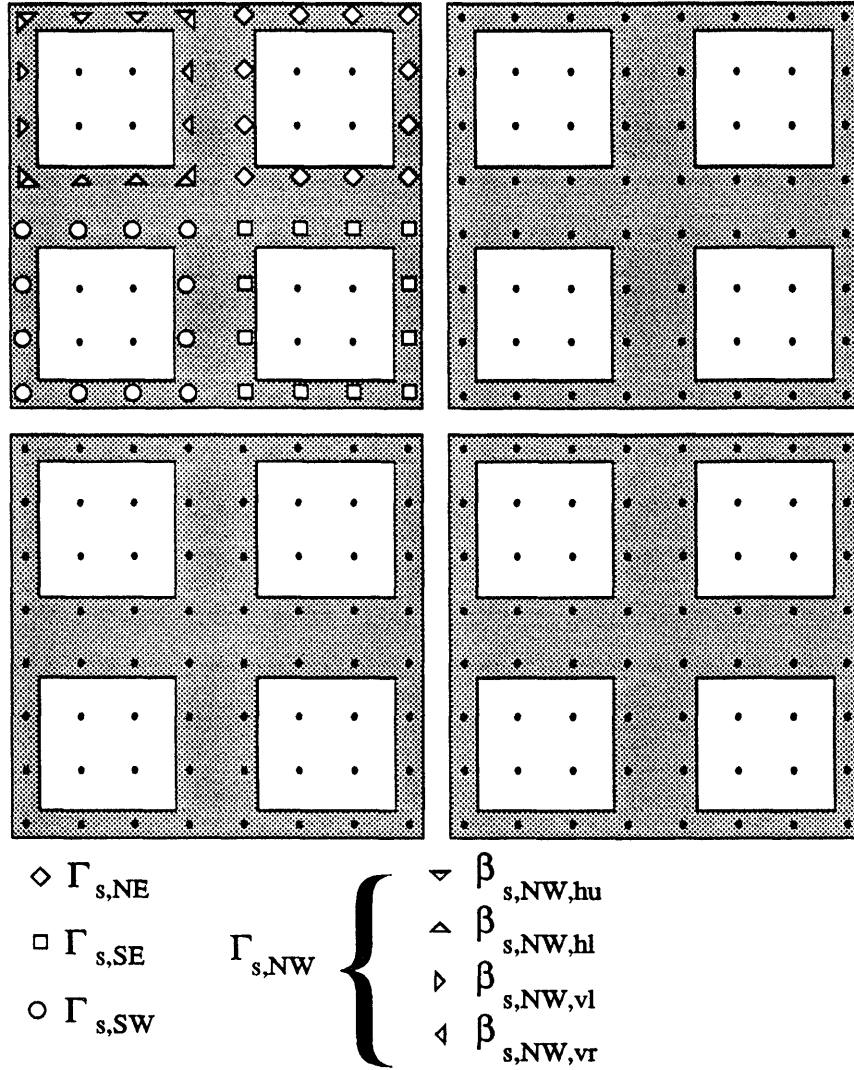


Figure 3-14: The four states at the second level of the tree in a non-redundant exact multiscale representation are scaled and shifted versions of the state at the root node, and are shown here for an MRF defined on a  $16 \times 16$  lattice. The state in the north-west corner contains the values of the process at the shaded points in the north-west  $8 \times 8$  quadrant. With the node  $s$  corresponding to this north-west corner state, the sets  $\Gamma_{s,i}$  and sequences  $\beta_{s,NW,j}$  are illustrated. Note again that the sequences  $\beta_{s,i,j}$  overlap.

and simple as fewer and fewer coefficients are retained. While one can imagine using a wide variety of different coordinate transformations, including a 1-D Fourier transform or a Karhunen-Loève expansion, we focus here on a choice that is particularly well matched to the self-similar, multiresolution nature of our exact representation. Specifically, as illustrated in Figures 3-13 and 3-14, as we proceed from one level to the next finer level in our multiscale representation of a Gaussian MRF, the boundaries at the coarser levels are essentially halved in length (and new, shorter boundaries are added as well). This self-similar structure suggests representing the values of the MRF along such boundaries in terms of wavelet bases. In this case, if at each level we keep only the wavelet coefficients up to a particular level of resolution, then at each level in our representation we are actually adding only one new level of detail.

The approximation that is made in keeping only coarse wavelet approximations along boundaries actually has two parts. The first is that we assume that the new detail to be added along each boundary from level to level is independent of the previously generated coarse approximations. The second is that we neglect the residual correlation between MRF values in neighboring subregions when we are given only a coarse approximation, rather than complete knowledge of, the values along their common boundary. The former of these approximations, namely the scale-to-scale decorrelation capability of wavelet transforms, has already been studied and well documented in several papers on 1-D stochastic processes [31, 135]. The latter of these, which deals explicitly with the full statistical structure of 2-D MRF's, has not, to our knowledge, been investigated previously (in fact, the use of 1-D wavelets for 2-D random fields is, we believe, a completely new idea). As the results presented here illustrate, this appears to be an extremely effective method for many MRF's.

The approximate models are based on the *non-redundant* exact representations for MRF's described in the previous section. The states at the first and second levels of this representation for an MRF defined on a  $16 \times 16$  lattice are shown in Figures 3-13 and 3-14. The root node state in Figure 3-13 contains the values of the MRF at 112 points. More generally, in a multiscale representation of an MRF defined on a  $2^N \times 2^N$  lattice, a state at the  $m^{\text{th}}$  level in this exact representation

represents the values of the MRF at  $16(2^{N-m-1} - 1)$  points. We denote this set of points as  $\Gamma_s$ , and we view it as the union of four mutually exclusive subsets. In particular, consider again the 112 points associated with the root node state in Figure 3-13. We can view these as four sets of 28 points, each of which corresponds to the boundary of one  $8 \times 8$  quadrant. In general, we can divide the set  $\Gamma_s$  into four sets of  $4(2^{N-m(s)-1} - 1)$  points in a similar fashion, and we denote these mutually exclusive subsets as  $\Gamma_{s,i}, i \in \{NW, NE, SE, SW\}$ , where the subscripts refer to the spatial location of the subset. For instance, with  $s = 0$  corresponding to the root node, the four subsets  $\Gamma_{0,i}, i \in \{NW, NE, SE, SW\}$  are illustrated in Figure 3-13 with the symbols:

$$\Gamma_{0,NW} \leftrightarrow \nabla, \triangleleft, \triangleright, \triangle, \text{ and combinations of these.} \quad (3.115)$$

$$\Gamma_{0,NE} \leftrightarrow \curvearrowright \quad (3.116)$$

$$\Gamma_{0,SE} \leftrightarrow \sqsupset \quad (3.117)$$

$$\Gamma_{0,SW} \leftrightarrow \circ \quad (3.118)$$

Next, we interpret the set of values  $\{z(t), t \in \Gamma_{s,i}\}$  for each of these quadrant boundaries, as four 1-D sequences of length  $2^{N-m(s)-1}$ , corresponding to each of the sides of the quadrant boundary. Thus, there are a total of *sixteen* 1-D boundary sequences associated with the set  $\Gamma_s$ , and we denote these as:

$$\beta_{s,i,j}, i \in \{NW, NE, SE, SW\}, j \in \{hu, hl, vl, vr\} \quad (3.119)$$

where the latter four subscripts refer to the “horizontal, upper”, “horizontal, lower”, “vertical, left” and “vertical, right”, respectively. For instance, for the  $16 \times 16$  lattice, the sequences  $\beta_{0,i,j}$  are shown in Figure 3-13 and defined below:<sup>7</sup>

$$\beta_{0,NW,hu}(k) = z(0, k), \quad \text{corresponding to the points denoted with } \nabla,$$

---

<sup>7</sup>We will use  $z(i, j)$  to denote the value of the MRF at lattice site  $(i, j)$ . If the lattice has  $T_1$  rows and  $T_2$  columns, then  $(i, j) \in \{0, 1, \dots, T_1 - 1\} \times \{0, 1, \dots, T_2 - 1\}$ .



the combination of  $\nabla$  and  $\triangleleft$ ,  
and the combination of  $\nabla$  and  $\triangleright$  in Fig. 15.

(3.120)

$\beta_{0,NW,vr}(k) = z(k, 7)$ , corresponding to the points denoted with  $\triangleleft$ ,  
the combination of  $\triangleleft$  and  $\nabla$ ,  
and the combination of  $\triangleleft$  and  $\triangle$  in Fig. 15.

(3.121)

$\beta_{0,NW,hl}(k) = z(7, k)$ , corresponding to the points denoted with  $\triangle$ ,  
the combination of  $\triangle$  and  $\triangleleft$ ,  
and the combination of  $\triangle$  and  $\triangleright$  in Fig. 15.

(3.122)

$\beta_{0,NW,vl}(k) = z(k, 0)$ , corresponding to the points denoted with  $\triangleright$ ,  
the combination of  $\triangleright$  and  $\nabla$ ,  
and the combination of  $\triangleright$  and  $\triangle$  in Fig. 15.

(3.123)

$$\beta_{0,NE,hu}(k) = z(0, k + 8) \quad (3.124)$$

$$\beta_{0,NE,vr}(k) = z(k, 15) \quad (3.125)$$

$$\beta_{0,NE,hl}(k) = z(7, k + 8) \quad (3.126)$$

$$\beta_{0,NE,vl}(k) = z(k, 8) \quad (3.127)$$

$$\beta_{0,SE,hu}(k) = z(8, k + 8) \quad (3.128)$$

$$\beta_{0,SE,vr}(k) = z(k + 8, 15) \quad (3.129)$$

$$\beta_{0,SE,hl}(k) = z(15, k + 8) \quad (3.130)$$

$$\beta_{0,SE,vl}(k) = z(k + 8, 8) \quad (3.131)$$

$$\beta_{0,SW,hu}(k) = z(8, k) \quad (3.132)$$

$$\beta_{0,SW,vr}(k) = z(k + 8, 7) \quad (3.133)$$

$$\beta_{0,SW,hl}(k) = z(15, k) \quad (3.134)$$

$$\beta_{0,SW,vl}(k) = z(k + 8, 0) \quad (3.135)$$

for  $k = 0, 1, \dots, 7$ . Note there is overlap in the sequences  $\beta_{s,i,j}$ . For instance,  $\beta_{0,NW,hu}$  and  $\beta_{0,NW,vl}$  both contain the value of the process at  $(0,0)$ , and this fact is reflected in Figure 3-13 by the presence of both  $\nabla$  and  $\triangleright$  at this lattice point.

Let us now consider the simplest of our approximate models. Specifically, we take as the state of the approximate representation just the *averages* of the sequences  $\beta_{s,i,j}$ . The state at any node then has sixteen components:

$$x(s) = \begin{bmatrix} x_{NW}(s) \\ x_{NE}(s) \\ x_{SE}(s) \\ x_{SW}(s) \end{bmatrix} \quad (3.136)$$

where:

$$x_i(s) = \begin{bmatrix} W_0\beta_{s,i,hu} \\ W_0\beta_{s,i,vr} \\ W_0\beta_{s,i,hl} \\ W_0\beta_{s,i,vl} \end{bmatrix} \quad (3.137)$$

for  $i \in \{NW, NE, SE, SW\}$  and where  $W_0\beta_{s,i,j}$  denotes the average of the sequence  $\beta_{s,i,j}(k)$ . Given the definition of the state (3.136),(3.137) (which will be generalized shortly to allow general wavelet transform approximations to the sequence  $\beta_{s,i,j}$ ), the conditional parent-offspring pdf's need to be obtained from the MRF being approximated. Instead of using these directly, we make an additional approximation. Let us define the downshift operators  $\alpha_i, i \in \{NW, NE, SE, SW\}$ , which are the counterparts of the upshift operator  $\bar{\gamma}$  defined previously. In particular, we denote the four offspring of node  $s$  as  $s\alpha_i, i \in \{NW, NE, SE, SW\}$ , where the subscript refers to the relative spatial location of the offspring nodes. In the *exact, non-redundant*

representations, the following relationship holds:

$$\begin{aligned}
 p_{\mathbf{z}(t), t \in \Gamma_{s\alpha_i} | \mathbf{z}(\tau), \tau \in \Gamma_s}(\mathbf{Z}_t, t \in \Gamma_{s\alpha_i} | \mathbf{Z}_\tau, \tau \in \Gamma_s) = \\
 p_{\mathbf{z}(t), t \in \Gamma_{s\alpha_i} | \mathbf{z}(\tau), \tau \in \Gamma_{s,i}}(\mathbf{Z}_t, t \in \Gamma_{s\alpha_i} | \mathbf{Z}_\tau, \tau \in \Gamma_{s,i})
 \end{aligned} \tag{3.138}$$

for  $i \in \{NW, NE, SE, SW\}$ . What (3.138) says is that the conditional pdf for the state at node  $s\alpha_i$  depends only on a subset of the values making up the state at the parent node  $s$ . For example, in the case of the *NW* offspring of node  $s$ , the state in the exact representation at node  $s\alpha_{NW}$  (that is,  $\mathbf{z}(t), t \in \Gamma_{s\alpha_{NW}}$ ) depends *only* on the *NW* component of the state at node  $s$  (that is, on the values  $\mathbf{z}(t), t \in \Gamma_{s,NW}$ ). Thus, in the exact representation the state at node  $s\alpha_{NW}$  is *independent* of the values of the MRF at the points in  $\Gamma_{s,NE}, \Gamma_{s,SE}$  and  $\Gamma_{s,SW}$ , given the values at  $\Gamma_{s,NW}$ . In contrast, it is *not* true in general in the simple approximate representations just described that the state  $\mathbf{x}(s\alpha_{NW})$  is independent of  $\mathbf{x}_{NE}(s), \mathbf{x}_{SE}(s)$  and  $\mathbf{x}_{SW}(s)$ , given  $\mathbf{x}_{NW}(s)$ . That is, simply knowing the average value of a process along each side of a square region does not completely decorrelate the values of the field inside and outside the region. Nevertheless, in our approximate modeling framework we will make exactly this assumption. More generally and precisely, our approximate modeling methodology yields a sequence of models corresponding to differing resolution approximations to the boundary processes  $\beta_{s,i,j}(k)$ , where (3.136) – (3.137) corresponds to the coarsest of these. Using the same symbols  $\mathbf{x}_i(s)$  and  $\mathbf{x}(s)$  to denote the state components and state of any of these models, we construct our model by making the approximation corresponding to assuming that the conditional independence property holds, i.e. that:

$$p_{\mathbf{x}(s\alpha_i) | \mathbf{x}(s)}(X_{s\alpha_i} | X_s) = p_{\mathbf{x}(s\alpha_i) | \mathbf{x}_i(s)}(X_{s\alpha_i} | X_i(s)) \tag{3.139}$$

Since the field being approximated is assumed to be jointly Gaussian, the conditional density function (3.139) is parameterized by conditional means and covariances

as in (3.27) – (3.29):

$$p_{\mathbf{x}(s\alpha_i)|\mathbf{x}_i(s)}(X_{s\alpha_i}|X_i(s)) = \mathcal{N}(X_{s\alpha_i}; \hat{\mathbf{x}}_{s\alpha_i}, P_{s\alpha_i}) \quad (3.140)$$

where:

$$\begin{aligned} \hat{\mathbf{x}}_{s\alpha_i} &= \mathbf{E}\{\mathbf{x}(s\alpha_i)|\mathbf{x}_i(s)\} \\ &= \mathbf{E}\{\mathbf{x}(s\alpha_i)\mathbf{x}_i(s)^T\}(\mathbf{E}\{\mathbf{x}_i(s)\mathbf{x}_i(s)^T\})^{-1}X_i(s) \end{aligned} \quad (3.141)$$

$$\begin{aligned} P_{s\alpha_i} &= \mathbf{E}\{(\mathbf{x}(s\alpha_i) - \hat{\mathbf{x}}_{s\alpha_i})(\mathbf{x}(s\alpha_i) - \hat{\mathbf{x}}_{s\alpha_i})^T\} \\ &= \mathbf{E}\{\mathbf{x}(s\alpha_i)\mathbf{x}(s\alpha_i)^T\} \\ &\quad - \mathbf{E}\{\mathbf{x}(s\alpha_i)\mathbf{x}_i(s)^T\}(\mathbf{E}\{\mathbf{x}_i(s)\mathbf{x}_i(s)^T\})^{-1}(\mathbf{E}\{\mathbf{x}(s\alpha_i)\mathbf{x}_i(s)^T\})^T \end{aligned} \quad (3.142)$$

One can then derive the matrices  $A(s)$ ,  $B(s)$  and  $P_0$  in the multiscale representation of the random field:

$$A(s\alpha_{NW}) = [K_1, 0, 0, 0] \quad (3.143)$$

$$A(s\alpha_{NE}) = [0, K_2, 0, 0] \quad (3.144)$$

$$A(s\alpha_{SE}) = [0, 0, K_3, 0] \quad (3.145)$$

$$A(s\alpha_{SW}) = [0, 0, 0, K_4] \quad (3.146)$$

where:

$$K_i = \mathbf{E}\{\mathbf{x}(s\alpha_i)(\mathbf{x}_i(s))^T\}(\mathbf{E}\{\mathbf{x}_i(s)(\mathbf{x}_i(s))^T\})^{-1} \quad (3.147)$$

Also:

$$B(s\alpha_i)B(s\alpha_i)^T = P_{s\alpha_i} \quad (3.148)$$

$$P_0 = \mathbf{E}\{\mathbf{x}_0\mathbf{x}_0^T\} \quad (3.149)$$

The assumption (3.139) is directly reflected in (3.143) – (3.146). In particular, the state  $x(s\alpha_i)$  is a function *only* of the  $i^{\text{th}}$  component of the parent (cf. (3.136)). Thus, the assumption in (3.139) leads to relatively simple level-to-level interpolations. Indeed, if the MRF is *stationary*, from symmetry we see that not only do the parameters  $A(s), B(s)$  depend only on the *scale* of node  $s$ , but also,  $K_1 = K_2 = K_3 = K_4$ . Thus, in this case, the representations are quite simply described, and more importantly, this simple structure, in addition to the substantially reduced dimensionality of the approximate representations, leads to considerable efficiencies for smoothing and likelihood calculation algorithms (cf. Chapter 4).

As we have indicated, the generalization of the coarsest approximate model, with state given by (3.136), (3.137) corresponds to using wavelet transforms to obtain different resolution representations of the sequences  $\beta_{s,i,j}(k)$ . We utilize the wavelet transform for discrete sequences as described in [15]. The wavelet transform of  $\beta_{s,i,j}(k), k \in \{1, 2, \dots, 2^{N-m(s)-1}\}$  is a set consisting of a single “scaling” coefficient and  $2^{N-m(s)-1} - 1$  “detail” coefficients<sup>8</sup>. These are computed recursively according to<sup>9</sup>:

$$f_k^{j-1} = \sum_{n=1}^{n=2M} h_n f_{n+2k-2}^j \quad (3.150)$$

$$d_k^{j-1} = \sum_{n=1}^{n=2M} g_n f_{n+2k-2}^j \quad (3.151)$$

where the scaling coefficients and detail coefficients are  $f_k^j$  and  $d_k^j$  respectively,  $h_n, g_n$  are impulse responses of quadrature mirror filters [39, 125] of length  $2M$ , and where  $f_k^{N-m(s)} = \beta_{s,i,j}(k)$ . We say that a  $p^{\text{th}}$ -order representation of the sequence  $\beta_{s,i,j}(k)$  is a set consisting of the scaling coefficient and the wavelet coefficients up to order  $p$  in the wavelet expansion, and that a zeroth-order representation is a set consisting of just

---

<sup>8</sup>To be concrete, we assume that the wavelet transform filter/downsample operations are iterated until the sequence of scaling coefficients, i.e. the downsampled output of the lowpass component of the wavelet filter bank, is of length one. More generally, one could stop at any point in the decomposition.

<sup>9</sup>Our notation is slightly different from that in [15]. In particular, in [15], increasing superscript  $j$  corresponds to *lower* levels in the decomposition (i.e., *fewer* wavelet and scaling coefficients), while here it corresponds to *higher* levels.

the scaling coefficient. We denote the operator which maps the sequence  $\beta_{s,i,j}(k)$  to its  $p^{\text{th}}$ -order representation as  $W_p$ . Note that if  $p = N - m(s) - 1$  the representation is complete, since it contains the scaling coefficient and *all* of the wavelet coefficients. For  $p > N - m(s) - 1$  we take  $W_p = W_{N-m(s)-1}$  (i.e. if there are fewer than  $p$  scales of wavelet coefficients, we keep all of them).

The generalization of the approximate representation based on averages of the 1-D sequences  $\beta_{s,i,j}(k)$  discussed previously now just involves a new definition for the state variables  $x(s)$ . In particular, simply replace (3.137) with:

$$x_i(s) = \begin{bmatrix} W_p \beta_{s,i,hu} \\ W_p \beta_{s,i,vr} \\ W_p \beta_{s,i,hl} \\ W_p \beta_{s,i,vl} \end{bmatrix} \quad (3.152)$$

where  $W_p \beta_{s,i,j}$  denotes the  $p^{\text{th}}$ -order representation of the sequence  $\beta_{s,i,j}(k)$  (a vector of length  $2^p$  if  $p \leq N - m(s) - 1$  and of length  $2^{N-m(s)-1}$  if  $p > N - m(s) - 1$ ). Thus, the state at any given node consists of sixteen components, each a  $p^{\text{th}}$ -order representation of one of the 1-D boundary sequences  $\beta_{s,i,j}(k)$  associated with the state  $x(s)$ . Using this generalized definition for the state, and making the assumption in (3.139), the parameters  $A(s)$ ,  $B(s)$  and  $P_0$  are again given by (3.143) – (3.149).

Several comments are in order. First, note that a simple generalization of the above representation would be to allow *different* levels of approximation for different components of the boundary sequences (e.g. one might use a  $p_1^{\text{th}}$ -order approximation for “vertical” boundary sequences  $\beta_{s,i,j}$ ,  $j \in \{vr, vl\}$  and a  $p_2^{\text{th}}$ -order approximation for “horizontal” boundary sequences  $\beta_{s,i,j}$ ,  $j \in \{hu, hl\}$ ). Examples of such a generalization will be given in the next section in the context of approximate representations for MRF texture models.

Second, note that even if all of the wavelet coefficients are retained at all levels (i.e. if the boundary representations are complete), the representation we have just described will be exact only if the GMRF is Markov with respect to either the first or second-order neighborhood in Figure 3-10. As we have discussed, higher-order

neighborhoods lead to thicker boundaries, and this leads naturally to the idea of taking wavelet expansions of boundaries of width two or more, and utilizing these as the state. When the boundaries are expanded to have a width of  $q$  lattice sites, the state at node  $s$  will be broken up into the  $p^{\text{th}}$ -order representations of  $16q$  sequences of length  $2^{N-m(s)-1}$ . With this expanded family, the approximate representations can be made exact for any GMRF by keeping complete wavelet expansions of all boundary sequences  $\beta_{s,i,j}(k)$  at all scales. An example of an approximate representation which keeps wavelet coefficient along boundaries of width two is discussed in the next section.

Third, note that the covariance matrices required in (3.147) and (3.148) are not invertible if the representation of the 1-D boundary sequences is complete, due to the fact, as mentioned previously, that these sequences overlap. For instance, in Figure 3-13, both  $\beta_{0,NW,hu}$  and  $\beta_{0,NW,vl}$  contain the value of the state at pixel location  $(0,0)$ . In this case we have *redundant* information and hence the conditional expectation and error covariance formulas must be modified to deal with this. This modification is a straightforward matter as discussed in [101, 132].

Fourth, note that not only has the dimensionality of the representations been reduced in going from the exact to the approximate representations, but it has, in fact, been made *constant* at the first  $N-p$  levels of the quadtree, where  $p$  is the order of the approximation and the MRF is defined on a  $2^N \times 2^N$  lattice. In particular, the dimension of the state at node  $s$  is equal to  $16 \times 2^p$ , for  $m(s) \leq N-p-1$ . When  $m = N-p-1$ , the boundary sequence representations are *complete* and the dimension of the state drops by a factor of 2 at each level thereafter.

Fifth, because our approximate models keep only limited resolution versions of MRF values along 1-D boundaries, the quadtrees for these approximate models may require more levels than the exact model. For example, consider an MRF over a  $4 \times 4$  region. The exact representation of this field in our framework has only a single level, since the exterior boundaries and mid-lines form the *entire*  $4 \times 4$  region (consider Figure 3-13 adapted to a  $4 \times 4$  grid). On the other hand, a first-order Haar approximation would retain only the sixteen average values of pairs of horizontal or vertical points at the first level, only twelve of which are independent thanks to the

overlap in the  $\beta_{s,i,j}$  sequences. Consequently, in this case, we need a second level, corresponding to “averages” of single points, to completely represent the field.

Finally, the order of the approximations required to achieve a desired level of fidelity in the approximate model depends, of course, on the statistical structure of the specific GMRF under study. In the next section we present examples which illustrate this for several GMRF’s and a number of different approximate representations.

### 3.5 Examples of 2-D GMRF Approximate Representations

In this section we illustrate sample paths generated by our approximate representations for two examples of separable Gaussian MRF’s and then for two examples of non-separable Gaussian MRF’s. Separable MRF’s were one of the first widely used image models because of their simple covariance structure [65, 64], while non-separable GMRF’s have been widely used in the context of texture representation [23, 74, 35, 34, 33, 91, 90].

#### 3.5.1 Separable Gaussian MRF Examples

Consider a separable Gaussian MRF defined on a  $2^N \times 2^N$  lattice with a covariance function given by:

$$E\{z(i, j)z(k, l)\} = \sigma^2 \rho_x^{|i-k|} \rho_y^{|j-l|} \quad (3.153)$$

where  $\rho_x$  and  $\rho_y$  are one-step correlation parameters in the vertical and horizontal directions. These random fields are Markovian with respect to the second-order neighborhood given in Figure 3-10.

Let us construct a zeroth-order Haar approximate representation of this MRF. In this case, the state at the root node of the tree consists of sixteen values, representing the averages across each of the 1-D components of the boundary. Since this state variable is just a linear function of the values of the random field, its covariance



structure can be calculated directly from knowledge of the averages taken and the covariance function in (3.153). Indeed, the covariance matrix for the four averages corresponding to the north-west corner of the state at the root node is given by:

$$\mathbf{E} \left\{ \begin{bmatrix} W_0\beta_{0,NW,hu} \\ W_0\beta_{0,NW,vr} \\ W_0\beta_{0,NW,hl} \\ W_0\beta_{0,NW,vl} \end{bmatrix} \begin{bmatrix} W_0\beta_{0,NW,hu} \\ W_0\beta_{0,NW,vr} \\ W_0\beta_{0,NW,hl} \\ W_0\beta_{0,NW,vl} \end{bmatrix}^T \right\} = \begin{bmatrix} \omega_y & \psi & \varrho_x\omega_y & \psi \\ \psi & \omega_x & \psi & \varrho_y\omega_x \\ \varrho_x\omega_y & \psi & \omega_y & \psi \\ \psi & \varrho_y\omega_x & \psi & \omega_x \end{bmatrix} \quad (3.154)$$

where:

$$\omega_y = (2^{N-1}(1 + \rho_y)/(1 - \rho_y) - 2\rho_y(1 - \rho_y^{2^{N-1}})/(1 - \rho_y)^2) \quad (3.155)$$

$$\omega_x = (2^{N-1}(1 + \rho_x)/(1 - \rho_x) - 2\rho_x(1 - \rho_x^{2^{N-1}})/(1 - \rho_x)^2) \quad (3.156)$$

$$\psi = ((1 - \rho_x^{2^{N-1}})/(1 - \rho_x))((1 - \rho_y^{2^{N-1}})/(1 - \rho_y)) \quad (3.157)$$

$$\varrho_x = \rho_x^{2^{N-2}-1} \quad (3.158)$$

$$\varrho_y = \rho_y^{2^{N-2}-1} \quad (3.159)$$

Figures 3-15a and 3-15b illustrate  $256 \times 256$  sample paths of exact representations of the separable random fields for  $\rho_x = \rho_y = 0.7$  and  $\rho_x = \rho_y = 0.9$ , respectively. Sample paths of zeroth-order Haar approximations of these MRF's are shown in Figures 3-15c and 3-15d, respectively<sup>10</sup>. Note that for  $\rho_x = \rho_y = 0.7$ , the zeroth-order approximation is visually similar to the exact representation, with only minor boundary effects apparent, caused by the approximation in the first-order Haar model, i.e. the neglecting of the residual correlation in adjoining regions when we are given only the coarse Haar approximation of the field along common boundaries. As the coupling between pixels is increased, the effects of this coarse approximation become more apparent, as seen in the first-order approximation of the separable MRF with

<sup>10</sup>We emphasize that Figures 3-15c and 3-15d depict *sample paths* of the approximate representations, and *not* approximate representations of the sample paths in Figures 3-15a and 3-15b.

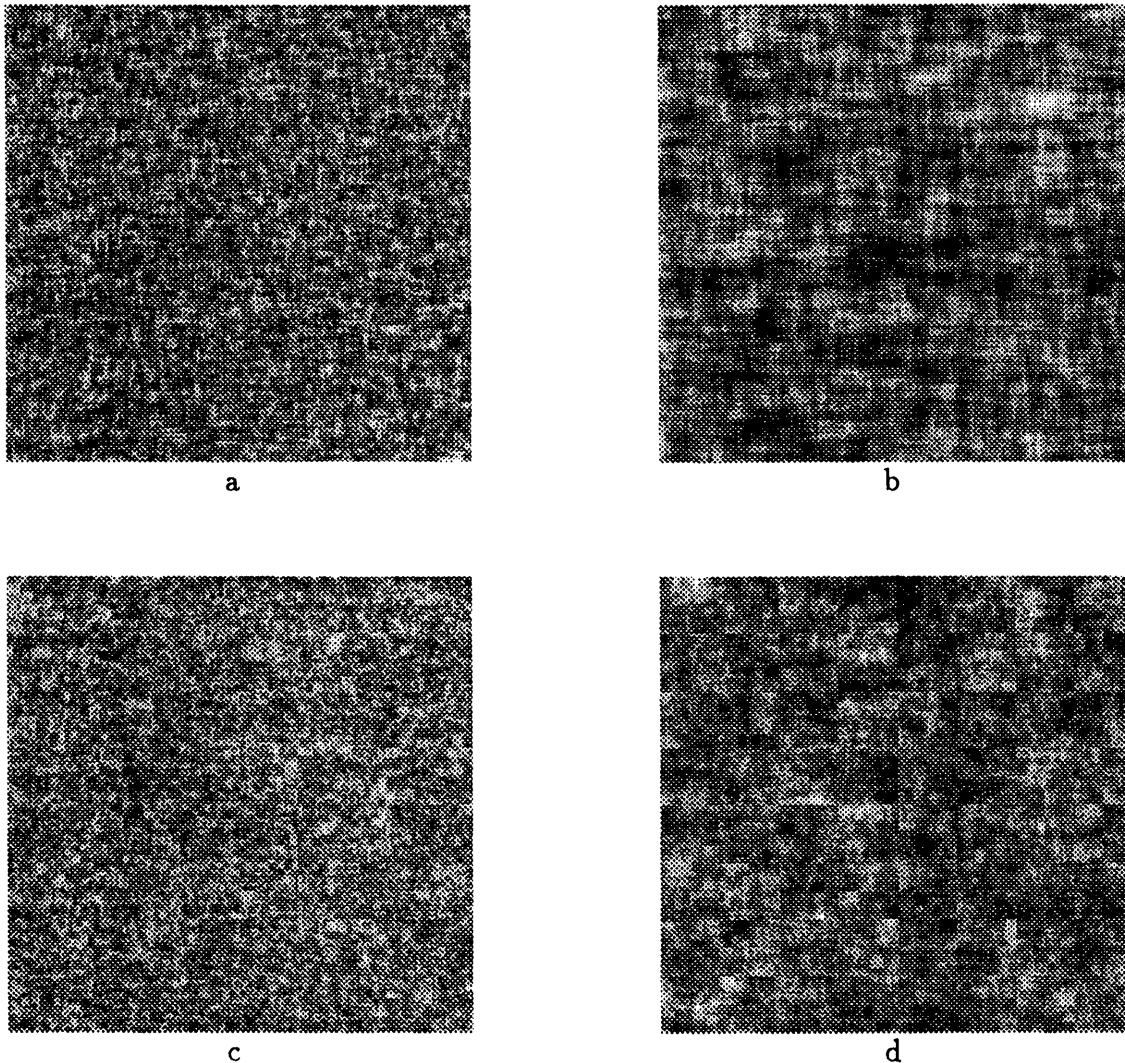


Figure 3-15: Sample paths of a separable Gaussian MRF with correlation structure  $\mathbf{E}\{z(i, j)z(k, l)\} = \sigma^2 \rho_x^{|i-k|} \rho_y^{|j-l|}$  with  $\rho_x = \rho_y = 0.7$  and  $\rho_x = \rho_y = 0.9$  are shown in (a) and (b) respectively. Sample paths of zeroth-order approximate representations of these fields, based on the Haar wavelet, are shown in (c) and (d). The stronger correlation between neighboring pixels in (b) leads to boundary effects in the sample path of the approximate representation shown in (d), which can be eliminated by using higher-order approximations.

$\rho_x = \rho_y = 0.9$ . This indicates that such fields will in general require higher-order approximations. We defer the illustration of such higher-order approximations of fields to the following subsection, in which we describe several examples of the use of our modeling methodology to represent natural textures.

### 3.5.2 Non-Separable Gaussian MRF Examples

Consider the class of GMRF's defined by the following 2-D autoregressive model [24, 71]:

$$z(i, j) = \sum_{(k, l) \in D} h_{k, l} z(i - k, j - l) + e(i, j) \quad (3.160)$$

where  $h_{k, l} = h_{-k, -l}$ ,  $D$  is a neighborhood around the origin  $(0, 0)$ , the Gaussian driving noise  $e(i, j)$  is a locally correlated sequence of random variables, and  $(i, j) \in \{0, 1, \dots, T_1 - 1\} \times \{0, 1, \dots, T_2 - 1\}$ . In the examples below, the set  $D$  corresponds to the neighborhood sets in Figure 3-10. For instance, the set corresponding to a first-order neighborhood is  $D = \{(0, 1), (0, -1), (1, 0), (-1, 0)\}$ . In addition, we interpret the lattice as a *toroid*, i.e. the independent variables  $(i, j)$  in (3.160) are interpreted modulo  $(T_1, T_2)$ . For instance, the first-order neighborhood of lattice site  $(0, 0)$  is given by the set  $\{(1, 0), (0, 1), (0, T_2 - 1), (T_1 - 1, 0)\}$ . Finally, the correlation structure of the driving noise is given by:

$$\mathbf{E}\{e(i, j)e(i - k, j - l)\} = \begin{cases} \sigma^2 & \text{if } k = l = 0 \\ -\sigma^2 h_{k, l} & \text{if } (k, l) \in D \\ 0 & \text{if } (k, l) \notin D \end{cases} \quad (3.161)$$

and has the property that:

$$\mathbf{E}\{e(i, j)z(k, l)\} = \begin{cases} \sigma^2 & \text{for } i = k, j = l \\ 0 & \text{else} \end{cases} \quad (3.162)$$

From (3.162), and the fact that the random field is Gaussian, one can prove that the autoregressive model above *does* imply that  $z(i, j)$  is a Markov random field [148]. We refer to the model (3.160) as a  $n^{\text{th}}$ -order MRF if the set  $D$  corresponds to the  $n^{\text{th}}$ -order neighborhood of Figure 3-10.

Infinite lattice versions of the processes in (3.160) were introduced in [148] and their toroidal lattice counterparts have been thoroughly studied in the context of

texture representation [23, 74, 35, 34, 33, 91, 90]. The correlation structure of these MRF's cannot be explicitly written down as in the previous example. However, the specific statistics and correlations (as in (3.143) – (3.146)) required to construct our multiscale approximate models can be computed efficiently using 2-D FFT's because of the fact that correlation matrices for these random fields, assuming lexicographic ordering, are block circulant with circulant blocks and hence these random fields are *whitened* by the 2-D Fourier transform [71]. In particular, denote by  $\mathbf{z}$  the set of values  $z(t)$  stacked into a vector (with lexicographic ordering), and denote the correlation matrix of the MRF by  $R_{\mathbf{z}\mathbf{z}}$ . Then:

$$FR_{\mathbf{z}\mathbf{z}}F^* = \Lambda \quad (3.163)$$

where  $F$  is the 2-D Fourier transform matrix and  $\Lambda$  is a diagonal matrix of the eigenvalues of  $R_{\mathbf{z}\mathbf{z}}$ . The wavelet coefficients required in the approximate representation correspond to linear functions of the values  $z(t)$ . That is:

$$W_p \beta_{s,i,j} = W_p S \mathbf{z} \quad (3.164)$$

$$= L \mathbf{z} \quad (3.165)$$

where the matrix  $L$  is the product of the wavelet transform operator  $W_p$  and a “selection” matrix  $S$  which generates the vector  $\beta_{s,i,j}$  from  $\mathbf{z}$ . Thus, to compute the correlation matrices required in the approximate representation, we need only compute functions of the form:

$$L_1 R_{\mathbf{z}\mathbf{z}} L_2^T = (L_1 F^*) \Lambda (F L_2^T) \quad (3.166)$$

Indeed, as described in Appendix C, the structure of the approximate representations and the stationarity of the GMRF allow us to compute the required correlations with only  $2^p$  1-D Fourier transform operations per level of the representation, where  $p$  is the order of the approximation. Furthermore, these calculations need only be performed *once*, since they are used simply to determine the parameters in the multiscale

$(k, l)$	$h_{k,l}$	$(k, l)$	$h_{k,l}$
(1,0)	0.4341	(0,2)	0.0592
(0,1)	0.2182	(-1,2)	-0.0302
(-1,1)	-0.0980	(1,2)	-0.0407
(1,1)	-0.0006	(-2,1)	0.0406
(2,0)	-0.0836	(2,1)	-0.0001

Table 3.1: Coefficients in the model (3.160) for the “wool” texture.

approximate model.

Figure 3-16a illustrates the “wool” texture from [74]. Three sample paths of approximate representations of this field based on the Haar wavelet are shown in Figures 3-16b to 3-16d. The wool texture corresponds to a fourth-order version of the model (3.160), with the coefficients given in Table 3.1.

Figures 3-16b and 3-16c correspond to zeroth and first-order approximations, respectively. Note in Figure 3-16c that some of the boundary effects apparent in Figure 3-16b have disappeared, due to the increase in the approximation order. Since the wool texture is actually Markov with respect to the fourth-order neighborhood of Figure 3-10, an exact representation of this field would require that boundaries of width equal to two be kept. To this end, the approximation shown in Figure 3-16d takes as state variables first-order approximations to these double width boundaries. Essentially all of the boundary effects present in the Figures 3-16b to 3-16c have been eliminated, and this representation appears to have retained the essential statistical and qualitative features of the exact representation used to generate Figure 3-16a.

Figure 3-17a illustrates the “wood” texture from [74], and three approximations of this MRF based on the Haar wavelet are shown in Figures 3-17b to 3-17d. The wood texture corresponds to a fourth-order version of the model (3.160), with the coefficients given in Table 3.2.

This texture clearly has a very asymmetric correlation structure, and thus we represent the vertical and horizontal boundary with different levels of approximation. In Figure 3-17b, the horizontal and vertical boundaries are represented with second and zeroth-order approximations respectively. In Figures 3-17c and 3-17d, the horizontal

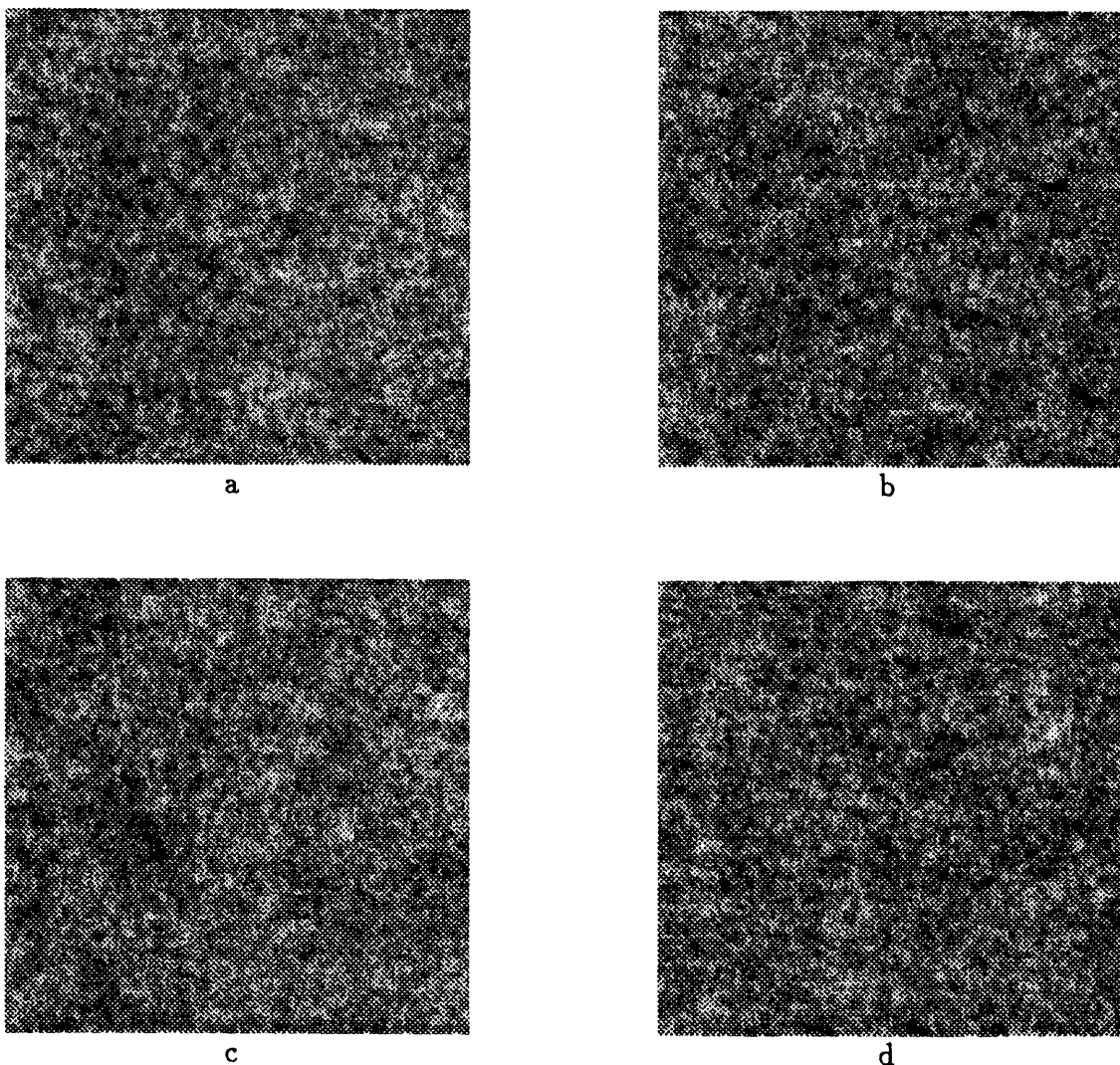


Figure 3-16: A sample path of a Gaussian MRF representing the “wool” texture of [74] is shown in (a). Figures 3-16b to 3-16d illustrate sample paths of approximate representations of this MRF based on the Haar wavelet. Zeroth and first-order approximations are used in (b) and (c), respectively. In (d), a first-order approximation based on boundaries of width two is used. Note that as the order of the approximate model is increased, the boundary effects disappear, and that for relatively low-order models an approximate representation which retains most of the qualitative and statistical features of the original MRF can be obtained.

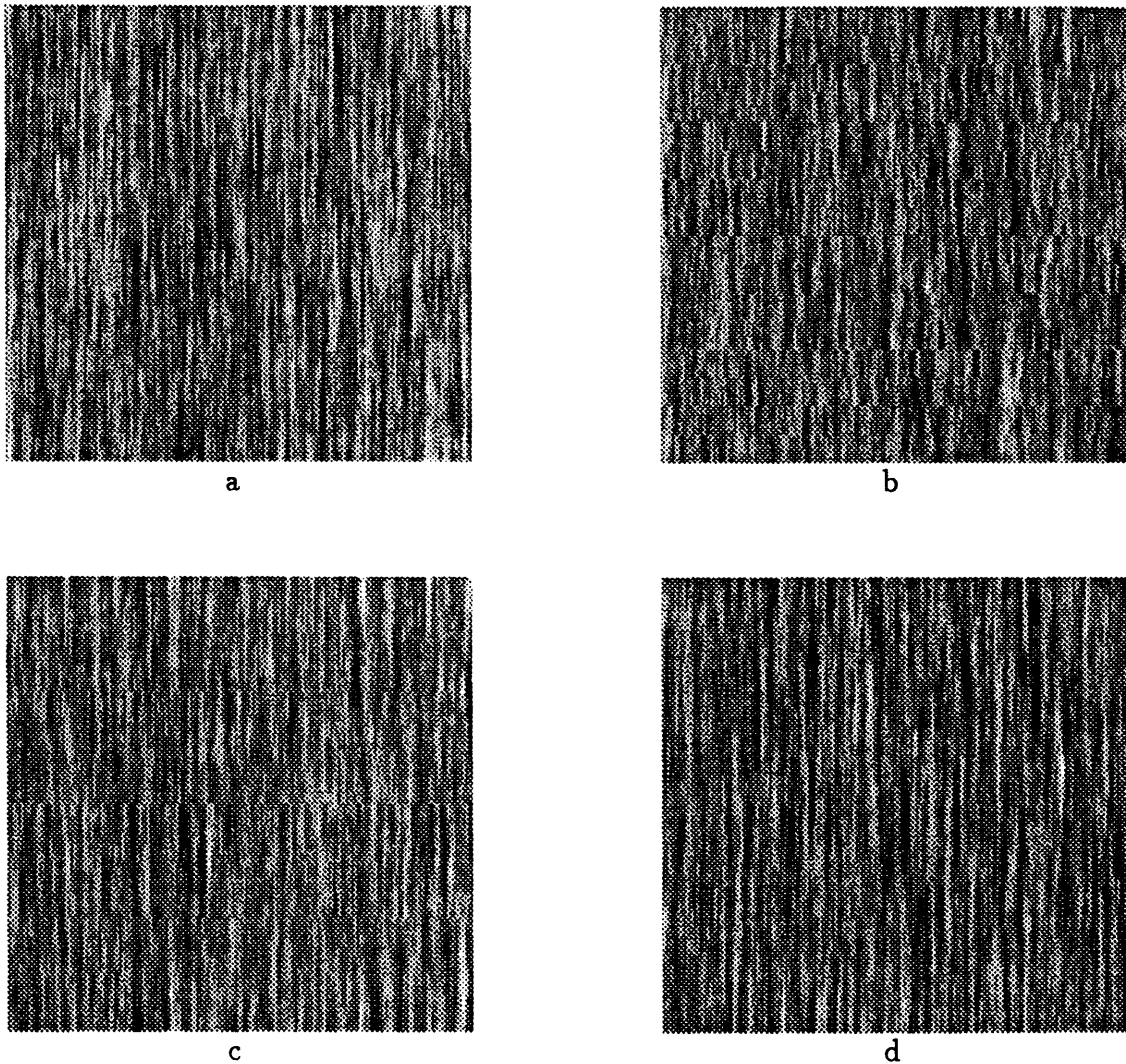


Figure 3-17: A sample path of a Gaussian MRF representing the “wood” texture of [74] is shown in (a). Figures 3-17b to 3-17d illustrate sample paths of approximate representations of the MRF based on the Haar wavelet. The structure of the MRF suggests using approximations which use relatively low order representations of vertical boundaries. The approximate representations used to generate Figures 3-17b to 3-17d used zeroth-order representations of the vertical boundaries, and second, fourth and sixth-order representations for the horizontal boundaries, respectively.

boundaries are represented with fourth and sixth-order approximations, respectively, whereas the vertical boundary is still represented with a zeroth-order approximation. As the complexity of the representation increases, the sample paths of the approximate random fields have fewer boundary effects. The approximate representations used to generate Figures 3-17c and 3-17d appear to accurately represent the qualita-

$(k, l)$	$h_{k,l}$	$(k, l)$	$h_{k,l}$
(1,0)	0.5508	(0,2)	0.0139
(0,1)	0.2498	(-1,2)	-0.0085
(-1,1)	-0.1164	(1,2)	-0.0058
(1,1)	-0.1405	(-2,1)	-0.0008
(2,0)	-0.0517	(2,1)	0.0091

Table 3.2: Coefficients in the model (3.160) for the “wood” texture.

tive and statistical features of the MRF. An interesting point here is that the level of representation only needs to be increased in one direction to obtain an excellent representation of the field. Also, the neighborhood of this MRF is fourth-order (see Figure 3-10) and thus double width boundaries would be needed in an exact representation. The fields shown in Figures 3-17b to 3-17d, however, use only the thinner boundaries in forming states. Several experiments were done in which we used the double width boundaries in forming states for models analogous to those in Figures 3-17b to 3-17d. It was found, however, that there were no visual differences between the single and double width approximate representations.

Three approximations of the “wood” texture based on the Daubechies 8 wavelet described in [39] are illustrated in Figures 3-18a to 3-18c. The order of the approximations are identical to the orders for Haar approximations in the previous example. We note that there is no apparent difference between the approximations based on the Haar wavelet and the Daubechies 8 wavelet. That is, at least for this example, and for the others we have examined, the critical issue in model fidelity appears to be model order rather than the particular choice of the wavelet used. Furthermore, as these examples indicate, we can achieve quite high quality results with low-order models, which in turn lead to extremely efficient algorithms as in [30, 27, 29, 31] and Chapter 4. In addition, as we briefly discuss in the next section, for GMRF’s which have particular directions in which correlation structures are oscillatory rather than monotonically decaying (such as the one describing the “wood” texture), there may be different choices of bases other than wavelets that lead to high fidelity models of even lower dimension.



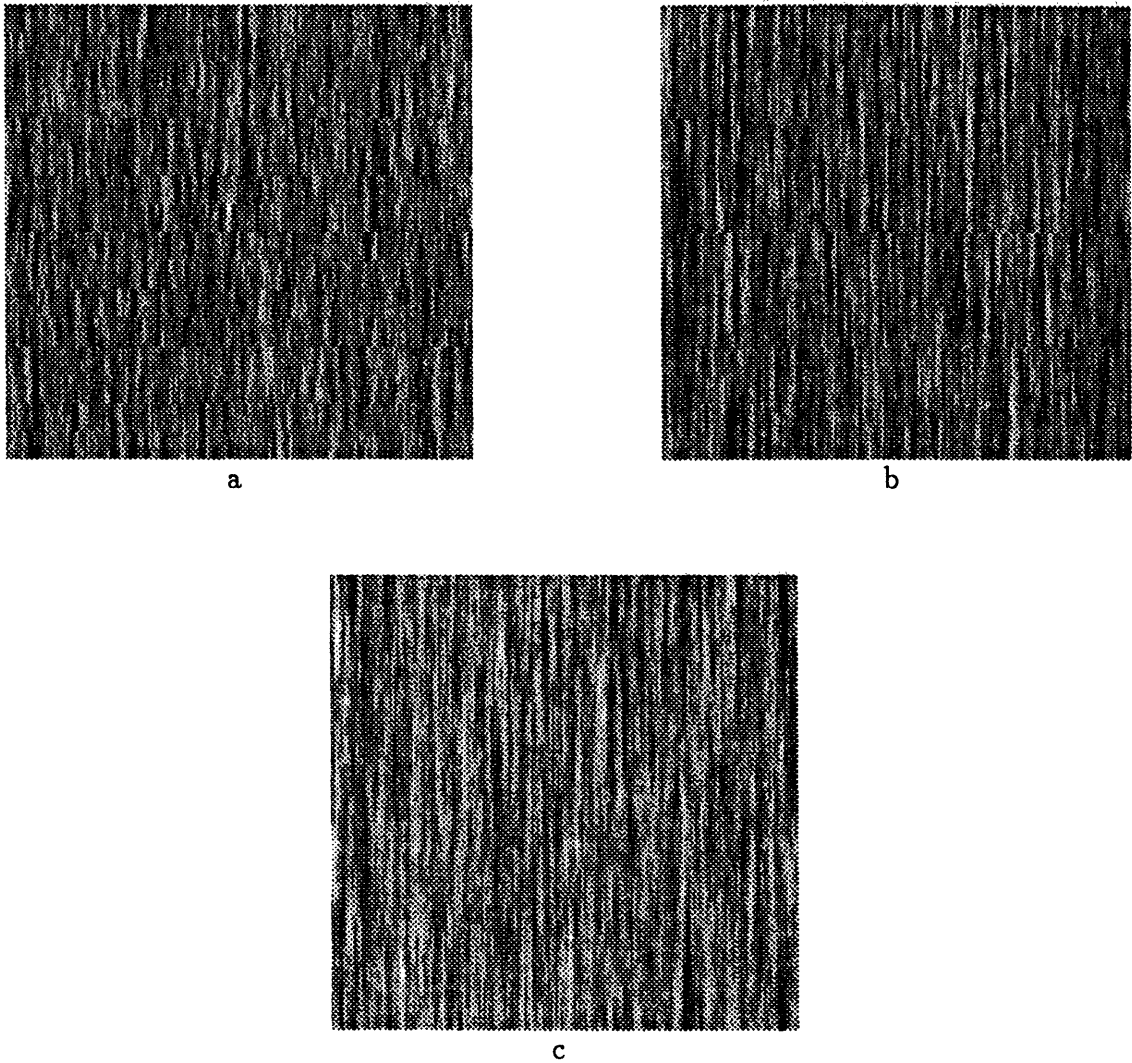


Figure 3-18: Figures 3-18a to 3-18c illustrate sample paths of approximate representations based on the Daubechies 8 wavelet, with the same orders of approximation as in Figure 3-17b to 3-17d.

### 3.6 Summary

In this chapter, we have shown how to represent reciprocal and Markov processes in one dimension and Markov random fields in two dimensions with a class of multiscale stochastic models. This modeling structure provides a framework for the development of efficient, scale-recursive algorithms for a variety of statistical signal processing problems. The representations in 1-D rely on a generalization of the mid-point deflection construction of Brownian motion. In 2-D, we introduced a “mid-line” construction

which leads to a class of models with scale-varying dimension. Since for reasonable size fields this state dimension may be prohibitively large, we also introduced a class of multiscale *approximate* MRF representations based on 1-D wavelet transforms of the MRF along 1-D boundaries of multiresolution partitionings of the 2-D domain of interest. This family of models allows one to tradeoff complexity and accuracy of the representations, and provides a framework for the development of extremely efficient estimation and likelihood calculation algorithms. Examples demonstrated that for relatively low-order models, an approximate representation which retains most of the qualitative and statistical features of the original MRF can be obtained. We use these models in Chapter 4 in the context of a texture discrimination application, and show how they lead to computationally efficient algorithms, with near optimal performance, in problems in which optimal GMRF-based approaches are computationally infeasible.

# Chapter 4

## Likelihood Calculations and their Applications

### 4.1 Introduction

Our development in Chapter 2 of an algorithm based on (1.16) for computing optical flow in an image sequence, and the construction in Chapter 3 of multiscale models for representing Markov random fields, provides substantial evidence that the multiscale model class can be used to model a broad range of phenomena and allows for the development of efficient image processing algorithms. Further algorithmic development is the focus of this chapter in which we introduce an algorithm for computing likelihoods for Gaussian multiscale models on  $q^{\text{th}}$  order trees. That is, we consider the problem of computing the conditional probability of a set of noisy observations, given that the data corresponds to a particular multiscale model. Likelihood calculations play a fundamental role in many problems, for instance Bayesian and Neyman-Pearson formulations of detection and classification problems [139, 119]. Our development exploits the scale-recursive structure of the multiscale model (1.16) thereby leading to a computationally efficiently and highly parallelizable algorithm. The approach allows for multiresolution data and parameters which vary in space as well as scale. In addition, it is non-iterative and in fact has a constant per-node computational complexity.

We illustrate one possible application of the algorithm to a texture classification problem. In the classical texture discrimination problem, one must choose from among a set of models the model which best represents or most likely corresponds to a given set of random field measurements [72]. As discussed in Chapter 3, Gaussian MRF (GMRF) and the multiscale models derived from them provide a good statistical representation of many textures. With stochastic models as a basis for texture representation, the texture classification problem is naturally formulated as a hypothesis test with a minimum probability of error criterion, and hence likelihood calculations play a fundamental role. Our purpose is to compare the relative computational requirements and classification accuracies of an approach based on the multiscale framework to that of standard GMRF-based approaches to the problem.

In this chapter, we will take the simplest of the approximate representations, namely the zeroth-order representation, as a basis for multiscale texture representation. These models, along with the GMRF models, then provide a basis for the hypothesis test associated with likelihood-based approaches to the texture discrimination problem. As we show in Section 4.3, even if the random field measurements actually do correspond to a GMRF, the probability of error performance loss in using the zeroth-order approximate GMRF models is only around 5%. Likewise, if the random field measurements correspond to a multiscale texture model (in particular, one of the zeroth-order approximate GMRF models) we show that the probability of error performance difference between GMRF and multiscale approaches is again on the order of 5%. Since in practice both of these models are idealizations, these performance results suggest the algorithms will provide similar performance on real data. In cases in which the random field measurements are available over a rectangular domain, our results argue for GMRF based approaches to the problem since in this case these provide superior computational performance, due to the fact that the required likelihood calculations can be carried out for GMRF models using 2-D FFT's. On the other hand, if the data are available over an irregularly shaped domain, if there are data dropouts, or if there are regions without data (due to camera blockage, for instance) then the 2-D FFT approaches break down for GMRF models and exact

likelihood calculation becomes computationally infeasible for even moderately sized domains. On the other hand, the multiscale approach can still be used under these more general conditions, and provides a computationally efficient method for carrying out the required likelihood calculations. Moreover, as we show, the multiscale approach provides probability of error performance which is substantially better than that of minimum-distance classifier approaches [23].

This chapter is organized as follows. In Section 4.2 the algorithm for computing the multiscale model likelihood function is developed. In Section 4.3 the results of experiments relative to the texture classification experiment described above are presented. A chapter summary is given in Section 4.4.

## 4.2 Likelihood Function Calculation

The *likelihood function* for the multiscale model (1.16) is defined as the log of the conditional probability of the data, given the model parameters. The unknown model parameters  $\theta$  may be entries in the matrices  $A, B, C, R$  and  $P_0$ , and they may also specify the state model dimension or the structure of the tree on which the model is defined (e.g. quadtree or dyadic tree). In this section, we provide an algorithm for computing the likelihood function.

To obtain an explicit form for the likelihood function, let us denote the set of nodes on the tree at which we have measurements as  $\mathcal{T}$  and then stack the measurements  $\{y(s)\}_{s \in \mathcal{T}}$  into a vector  $Y$ . Then the likelihood function  $\mathcal{L}(\theta)$  is given by:

$$\begin{aligned} \mathcal{L}(\theta) &= \log p_{y|\theta}(Y|\theta) \\ &= -\frac{1}{2} \log |\Lambda_y| - \frac{1}{2} Y^T \Lambda_y^{-1} Y - \frac{pS}{2} \log 2\pi \end{aligned} \quad (4.1)$$

where  $\Lambda_y$  is implicitly given by the model parameters,  $p$  is the dimension of the measurement  $y(s)$ , and  $S$  is the cardinality of the set  $\mathcal{T}$ .

The main problem in evaluating (4.1) is that the data covariance matrix  $\Lambda_y$  is generally full, and thus inverting it directly is difficult if the number of data points is

large. The algorithm below *whitens* the data, which allows the likelihood to be evaluated easily. That is, the data is invertibly transformed to a new set of data  $\{\nu(s)\}$ , such that  $\nu(s)$  and  $\nu(\sigma)$  are uncorrelated if  $s \neq \sigma$ . In particular, if we construct a vector  $\nu$  by stacking up the residuals  $\{\nu(s)\}_{s \in \mathcal{T}}$ , then  $\nu = Ty$  for some invertible matrix  $T$ , and the resulting covariance matrix,  $\Lambda_\nu = T\Lambda_y T^T$  is diagonal (or block diagonal). The fact that the transformation is invertible means that  $\{y(s)\}$  and  $\{\nu(s)\}$  contain the same information in the sense that the conditional statistics of the underlying state process are identical given  $\{y(s)\}$  or  $\{\nu(s)\}$ . The whitened measurements  $\{\nu(s)\}$  are preferred as these lead to simple calculation of the counterpart to (4.1):

$$\mathcal{L}(\theta) = -\frac{1}{2} \sum_{s \in \mathcal{T}} [p \log 2\pi + \log |\Lambda_{\nu(s)}| + \nu^T(s) \Lambda_{\nu(s)}^{-1} \nu(s)] \quad (4.2)$$

where we have made the assumption (for simplicity of presentation only) that the determinant of the transformation matrix  $T$  is equal to 1.

If the eigenvectors and eigenvalues of  $\Lambda_y$  are known, then a transformation based on this eigendecomposition provides a natural approach to the whitening problem. Indeed, if the multiscale model parameters vary only as a function of scale, then the Haar transform (and appropriate generalizations for trees of order  $q > 2$ ) can be used to whiten the data [30, 27, 28, 45]. On the other hand, when the model parameters vary in both space and scale, an eigendecomposition of the measurement data covariance matrix is not immediately available, and hence for this case we propose a Gram-Schmidt orthogonalization approach. This approach and the resulting algorithm are heavily influenced by related Gram-Schmidt approaches proposed for calculating likelihoods corresponding to Gauss-Markov models [121, 68]. In this latter case, the Kalman filter actually performs the whitening and the only additional computation that needs to be done corresponds to (4.2). For the multiscale model (1.16), we also have a Kalman filter (corresponding to the *upward* sweep of the smoothing algorithm described in Section 1.3). However, in this case, the Kalman filter provides only a *partial* whitening of the data due to the more complicated structure of higher order trees. For instance, while a measurement at any given node on a higher order

tree will be whitened with respect to the measurements in the subtree below, it will *not* be whitened with respect to other measurements at the same level.

In particular, we define the residuals generated by the multiscale Kalman filter as:

$$\nu_f(s) \equiv y(s) - C(s)\hat{x}(s|Y_s^{\alpha_q}) \quad (4.3)$$

where the subscript  $f$  is used to distinguished these filter residuals from the residuals  $\nu(s)$  which will be the result of the likelihood calculation algorithm. The fact that the set  $\{\nu_f(s)\}$  is not white is apparent from the update equation (1.33) in which the residual term  $\nu_f(s)$  is used to obtain  $\hat{x}(s|Y_s)$ . Since the estimate  $\hat{x}(s|Y_s^{\alpha_q})$  does not depend on nodes outside the subtree below node  $s$ , there is no reason to expect that  $\nu_f(s)$  is orthogonal to the corresponding residuals calculated at nodes at the same level as node  $s$ . The difficulty arises from the fact that while the Kalman filter on the first-order tree respects a total order on the first-order tree nodes (it processes the measurement at node  $t$  after that at  $t-1$ , etc.), the Kalman filter on higher order trees does not (it operates on many nodes in parallel). However, while the Kalman filter for the multiscale model does not completely whiten the data, we can take advantage of the partial whitening it performs. In particular, one of the main components of our likelihood calculation algorithm is a modified version of the multiscale Kalman filter; the other main part is associated with turning the partially whitened data into fully whitened data.

### 4.2.1 From Partial to Total Ordering

To see how this can be done, consider first the set of measurements  $Y_s$  associated with node  $s$  in Figure 4-1. The corresponding set of residuals generated by the Kalman filter is  $\{\nu_f(s), y(s\alpha_1), y(s\alpha_2), y(s\alpha_3)\}$ , since  $\hat{x}(s\alpha_i|Y_{s\alpha_i}^{\alpha_q})$  is initialized with zero. The residual  $\nu_f(s)$  is orthogonal to  $y(s\alpha_1), y(s\alpha_2)$  and  $y(s\alpha_3)$  and so we can set  $\nu(s) = \nu_f(s)$ . What we need to do is calculate  $\nu(s\alpha_i), i = 1, 2, 3$  such that these residuals are mutually orthogonal. The way to do this is with a Gram-Schmidt

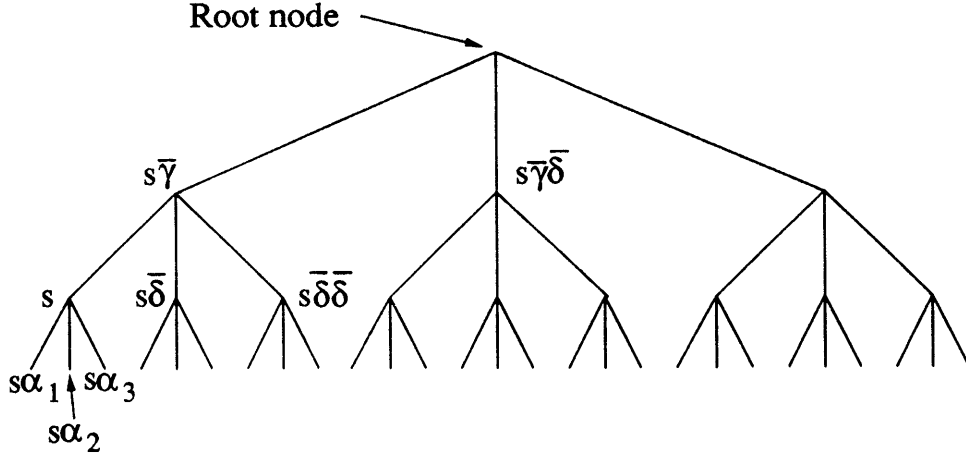


Figure 4-1: Third order tree.

procedure:

$$\nu(s\alpha_i) = y(s\alpha_i) - \mathbf{E}\{y(s\alpha_i)|y(s\alpha_j), j < i\} \quad (4.4)$$

For  $i = 1$ , we just have  $\nu(s\alpha_1) = y(s\alpha_1)$ . To perform the calculation for  $i = 2, 3$ , we need to propagate information around the tree structure in an efficient way. In particular, to obtain  $\nu(s\alpha_2)$  we use the upward dynamics (1.21) to calculate  $\hat{x}(s|Y_{s\alpha_1})$ , use the downward dynamics (1.16) to obtain  $\hat{x}(s\alpha_2|Y_{s\alpha_1})$ , and then use the linearity of the expected value operator to obtain the residual. This information flow is illustrated in Figure 4-2a, with the arrows representing the directed flow. The information flow required to obtain  $\nu(s\alpha_3)$  is shown in 4-2b. In particular, we again use the upward dynamics to propagate the information about  $y(s\alpha_3)$  contained in  $y(s\alpha_2)$  up the tree. The sufficient statistic for this information at node  $s$  is  $\hat{x}(s|Y_{s\alpha_2})$ , and this can be combined with  $\hat{x}(s|Y_{s\alpha_1})$  to give  $\hat{x}(s|Y_{s\alpha_1}, Y_{s\alpha_2})$ , which is then propagated back *down* the tree using (1.16) to obtain the residual at  $\nu(s\alpha_3)$ .

With the residuals at the four nodes  $s, s\alpha_1, s\alpha_2, s\alpha_3$  computed, we focus now on the computation required to whiten the measurements in the set  $Y_{s\bar{\delta}}$ . The procedure will be very similar to that used to whiten the measurement set  $Y_s$ , but with the additional complication that we will also need to whiten  $Y_{s\bar{\delta}}$  with respect to  $Y_s$ . The



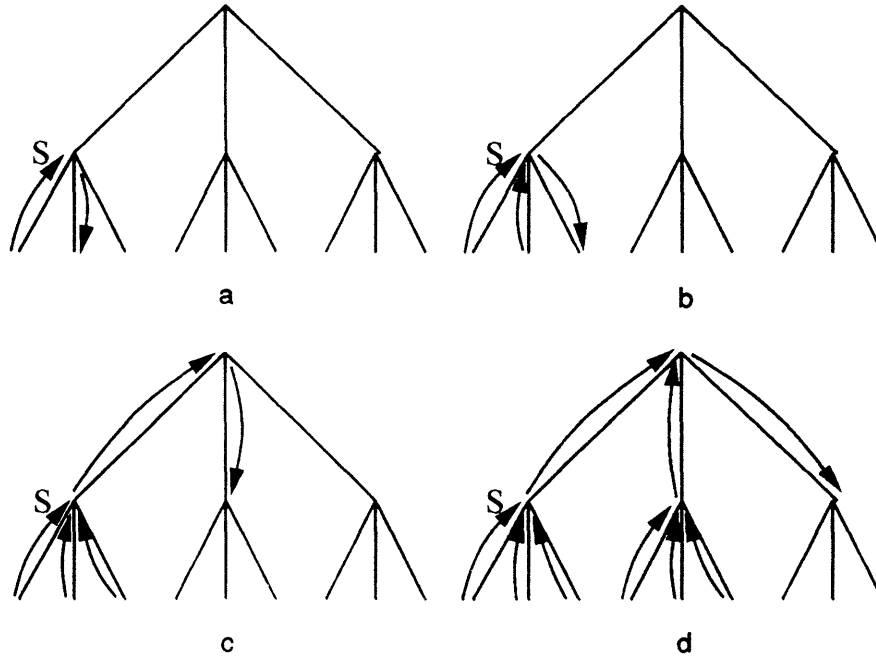


Figure 4-2: This figure illustrates the information flow in the likelihood calculation algorithm. (a) The residual at  $s\alpha_2$  is computed by using the model dynamics to propagate the information in  $y(s\alpha_1)$  up to node  $s$  and back down to  $s\alpha_2$ . (b) Likewise, to get the residual at  $s\alpha_3$ , we combine information from  $y(s\alpha_1)$  and  $y(s\alpha_2)$  at node  $s$ , and then use the dynamics to propagate this back down towards  $s\alpha_3$ . (c) Having whitened the measurements in the set  $Y_s$ , we consider next those at  $Y_{s\bar{\delta}}$ . We first propagate information contained about these measurements over from the set  $Y_s$ , and then combine this with the information in  $Y_{s\bar{\delta}}$  to get a set of residuals  $s\bar{\delta}, s\bar{\delta}\alpha_1, s\bar{\delta}\alpha_2, s\bar{\delta}\alpha_3$ , which are mutually orthogonal, and orthogonal to  $Y_s$ . (d) The process continues by propagating the information in  $Y_s$  and  $Y_{s\bar{\delta}}$  up to  $s\bar{\gamma}$ , and then back down to  $s\bar{\delta}$ .

information required to do this is contained in  $\hat{x}(s|Y_s)$  which we can propagate over to the subtree below  $s\bar{\delta}$  by using the upward and downward dynamics of the multiscale process, as shown in Figure 4-2c. Having obtained  $\hat{x}(s\bar{\delta}|Y_s)$ , we can form the residual at  $s\bar{\delta}\alpha_1$  by propagating this information down the tree. By construction, this residual is orthogonal to the residuals at the four nodes  $s, s\alpha_1, s\alpha_2, s\alpha_3$ . Likewise, we can get the residual at  $s\bar{\delta}\alpha_2$  by first computing  $\hat{x}(s\bar{\delta}|Y_{s\bar{\delta}\alpha_1})$ , combining this with  $\hat{x}(s\bar{\delta}|Y_s)$ , and then using the downward dynamics to propagate the combined information back down the tree towards  $s\bar{\delta}\alpha_2$ . In a similar way, we obtain the residuals at  $s\bar{\delta}\alpha_3$ . The residual at  $s\bar{\delta}$  is computed by combining the Kalman filter result  $\hat{x}(s\bar{\delta}|Y_{s\bar{\delta}}^{\alpha_3})$  with  $\hat{x}(s\bar{\delta}|Y_s)$ ,

and subtracting the resulting estimate of  $y(s)$  from  $y(s)$ . By construction, the four residuals at  $s\bar{\delta}, s\bar{\delta}\alpha_1, s\bar{\delta}\alpha_2, s\bar{\delta}\alpha_3$  are mutually orthogonal, and orthogonal to the set  $Y_s$ .

We continue this pattern of information flow to compute residuals at the set of nodes including and below node  $s\bar{\delta}\bar{\delta}$ . The construction of the residuals here is entirely similar to the construction at and below  $s\bar{\delta}$ , except that now we must be sure to whiten with respect to the measurements in both  $Y_s$  and  $Y_{s\bar{\delta}}$ . This leads to the information flow in Figure 4-2d, in which estimates  $\hat{x}(s|Y_s)$  and  $\hat{x}(s\bar{\delta}|Y_{s\bar{\delta}})$  are propagated up to  $s\bar{\gamma}$ , combined, and the propagated back down the tree.

The extrapolation of this idea to general  $q^{\text{th}}$ -order trees should now be apparent. We implicitly form a total ordering of the nodes on the tree which is compatible with the partial ordering induced by the Kalman filter, and perform the corresponding Gram-Schmidt calculations required to completely whiten the partially whitened data that the Kalman filter provides. This total ordering is illustrated in Figure 4-3 for a third order tree with four levels. The node  $s_0\alpha_1^{l-1}$ , where the scales are numbered, coarse-to-fine, as  $m = 0, 1, \dots, l$ , and  $s_0$  is the root node, is the starting point for our ordering. Starting there, we proceed to the right, labeling all of the offspring of a given parent, and then the parent. By labeling the offspring before the parent, we preserve the partial ordering induced by the Kalman filter. Specifically, since the Kalman filter residual  $\nu_f(s)$  is whitened with respect to measurements in the subtree below, the node  $s$  must be labelled *after* the nodes in the subtree below. This basic rule characterizes the ordering, up to some flexibility in the ordering of the offspring (which we label sequentially:  $s\alpha_1$  before  $s\alpha_2$  etc.) A formal description of the ordering is given in the pseudo-code algorithm in Table 4.1. The command "Label  $s$ " in the code means that node  $s$  is assigned the value of a counter (the counter is incremented after each labeling, and is initialized with 1).

## 4.2.2 Algorithm Description

The particular total ordering we have defined now allows us to carry out the whitening calculations in a parallel and decoupled fashion. The structure of the algorithm we

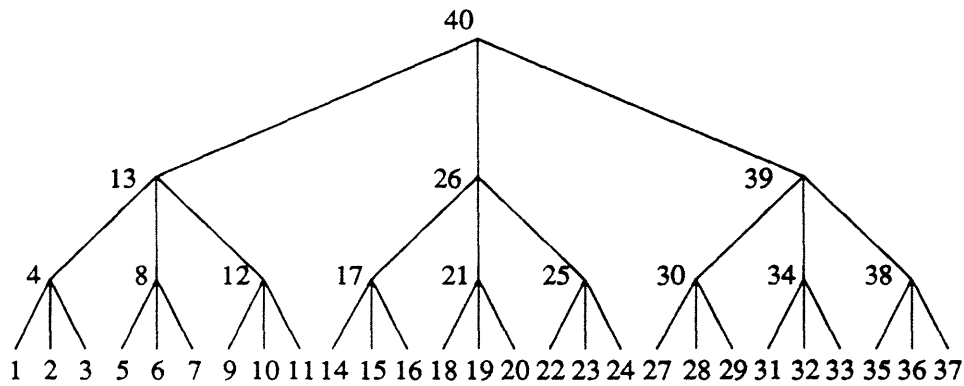


Figure 4-3: An ordering of the nodes on the tree convenient for computing the likelihood function is depicted.

```

 $s \leftarrow s_0 \alpha_1^{l-1};$ 
10 If all nodes below  $s$  are labeled then
    Label  $s$ ;
    If  $s$  is the root node then
        End;
    Elseif  $s = s \bar{\gamma} \alpha_q$  then
         $s \leftarrow s \bar{\gamma}$ ; Goto 10;
    Else
         $s \leftarrow s \bar{\delta}$ ; Goto 10;
    Else
         $s \leftarrow s \alpha_1$ ; Goto 10;

```

Table 4.1: Pseudo-code description of labeling sequence.

describe in this section is depicted in Figure 4-4. The algorithm can be broken up into three steps: an *upward sweep*, followed by a *downward sweep*, followed by the computation corresponding to (4.2). Before we describe the details of these steps, let us define a bit of notation and describe in general terms how the algorithm works.

First, note that the calculation of the residual at any given node  $s$  involves computing an estimate of  $x(s)$  based on the set  $Y_s^{\alpha_q}$ , plus possibly additional measurements corresponding to data that have already been whitened. To capture this other set of measurements succinctly, we define a bit of additional notation for subsets of

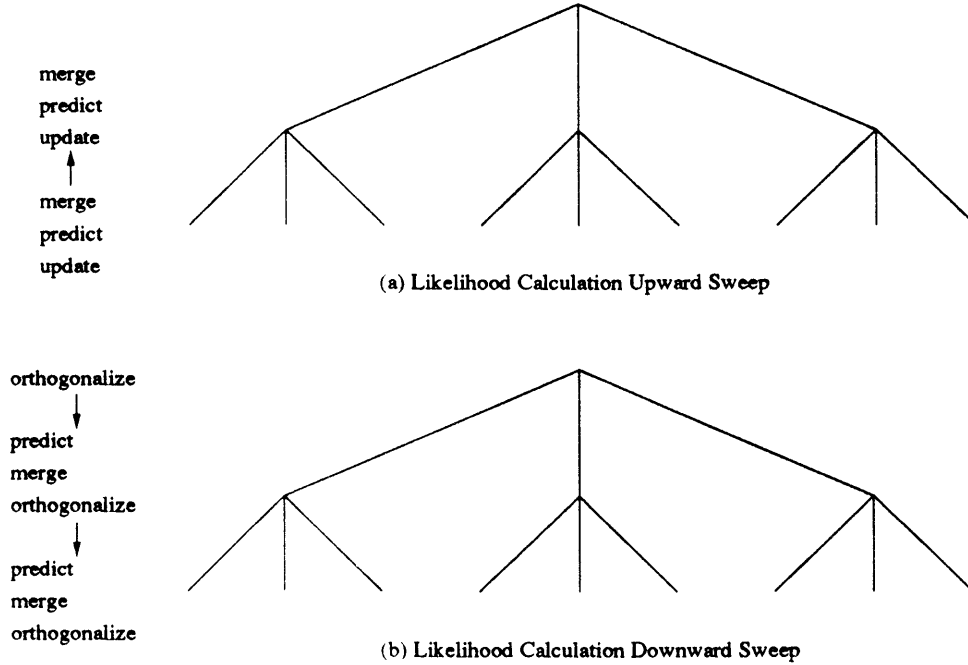


Figure 4-4: The general structure of the likelihood calculation algorithm is shown. The algorithm consists of an *upward sweep*, in which a series of update-predict-merge steps are used to recursively propagate information from the finest to coarsest levels, followed by a *downward sweep* consisting of a series of predict-merge-orthogonalize steps. The algorithm whitens noisy measurements of a multiscale process, so that the likelihood of the data can be easily computed using (4.2).

measurements on the tree. Specifically:

$$Y_s^{\alpha_i} = \cup_{j=1}^i Y_{s\alpha_j}, \quad \text{for } i = 1, 2, \dots, q \quad (4.5)$$

$$\bar{Y}_s = \{y(\sigma) | \sigma < s \text{ and } y(\sigma) \notin Y_s^{\alpha_q}\} \quad (4.6)$$

where the notation  $\sigma < s$  means that  $\sigma$  gets labelled before node  $s$  in the algorithm of Table 4.1. Examples of these subsets are shown in Figures 4-5 and 4-6. Note that the subsets  $Y_s^{\alpha_i}$  are nested,  $Y_s^{\alpha_1} \subset Y_s^{\alpha_2} \subset \dots \subset Y_s^{\alpha_q} \subset Y_s$ . Also, note that the sets  $\bar{Y}_s$  have a certain recursive structure. In particular, if  $s = s\bar{\gamma}\alpha_i$ :

$$\bar{Y}_s = \bar{Y}_{s\bar{\gamma}} \cup \left( \cup_{j=1}^{i-1} Y_{s\bar{\gamma}\alpha_j} \right) \quad (4.7)$$

$$= \bar{Y}_{s\bar{\gamma}} \cup Y_{s\bar{\gamma}}^{\alpha_{i-1}} \quad (4.8)$$

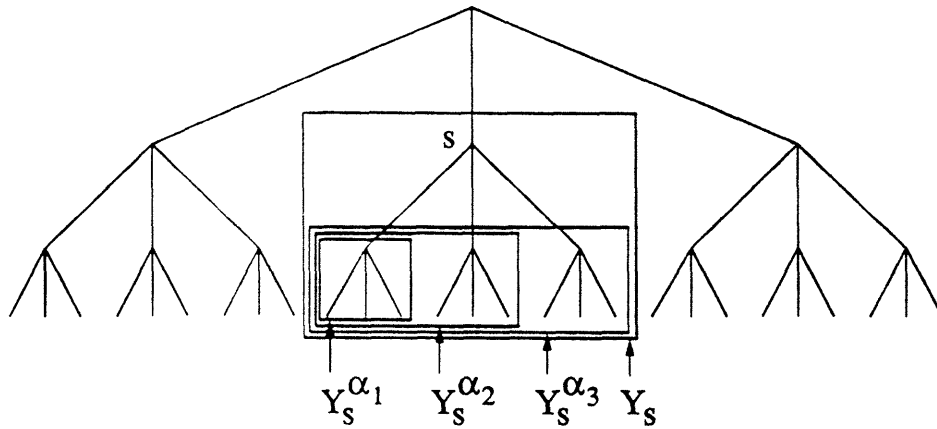


Figure 4-5: This figure provides examples of different subsets of nodes defined with respect to node  $s$ . Note that  $Y_s^{\alpha_1} \subset Y_s^{\alpha_2} \subset \dots \subset Y_s^{\alpha_q} \subset Y_s$ , where  $q = 3$  in the example above.

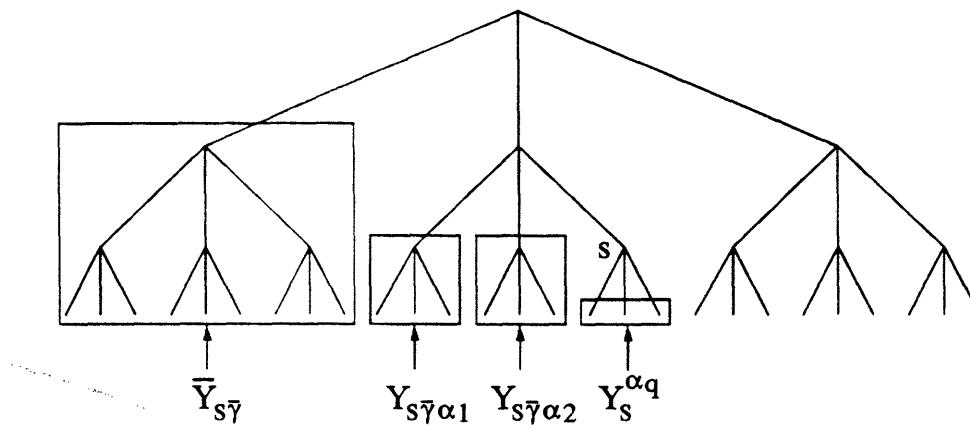


Figure 4-6: This figure shows examples of some other sets of nodes defined with respect to the node  $s$ . Note that  $\bar{Y}_s = \bar{Y}_{s\bar{\gamma}} \cup Y_{s\bar{\gamma}\alpha_1} \cup Y_{s\bar{\gamma}\alpha_2} = \bar{Y}_{s\bar{\gamma}} \cup Y_{s\bar{\gamma}}^{\alpha_2}$ .

with the convention that  $Y_{s\bar{\gamma}}^{\alpha_0} \equiv \emptyset$ . As an example, note that  $\bar{Y}_s = \bar{Y}_{s\bar{\gamma}} \cup Y_{s\bar{\gamma}\alpha_1} \cup Y_{s\bar{\gamma}\alpha_2} = \bar{Y}_{s\bar{\gamma}} \cup Y_{s\bar{\gamma}}^{\alpha_2}$  in Figure 4-6.

The general equation for the residual  $\nu(s)$  and its variance  $\Lambda_{\nu(s)}$  is:

$$\nu(s) = y(s) - C(s)\hat{x}(s|\bar{Y}_s, Y_s^{\alpha_q}) \quad (4.9)$$

$$\Lambda_{\nu(s)} = C(s)P(s|\bar{Y}_s, Y_s^{\alpha_q})C^T(s) + R(s) \quad (4.10)$$

Hence, the key quantities to compute are  $\hat{x}(s|\bar{Y}_s)$  and  $\hat{x}(s|Y_s^{\alpha_q})$ . Both  $\hat{x}(s|Y_s^{\alpha_q})$  and the set of estimates  $\hat{x}(s|Y_s^{\alpha_i})$  for  $i = 1, 2, \dots, q-1$ , are computed recursively through a

series of *update-predict-merge* steps, during the upward sweep of the algorithm. In the downward sweep, we use the estimates  $\hat{x}(s|Y_s^{\alpha_i})$  for  $i = 1, 2, \dots, q - 1$  to recursively compute  $\hat{x}(s|\bar{Y}_s)$  at each node, combine this with  $\hat{x}(s|Y_s^{\alpha_q})$ , and then use (4.10) to compute the residual  $\nu(s)$ . As we will see in the detailed development below, both the upward and downward sweeps have a highly regular, decoupled structure across scales and hence there is substantial potential for parallel implementation of the algorithm.

We begin with the upward sweep of the algorithm. As in the upward sweep of the smoothing algorithm described in Chapter 1, we start by initializing the estimates  $\hat{x}(s|Y_s^{\alpha_q})$  at the finest level of the tree to zero. Likewise, the covariances  $P(s|Y_s^{\alpha_q})$  at the finest level are initialized using the Lyapunov equation (1.26). Now, suppose we have the estimates  $\hat{x}(s|Y_s^{\alpha_q})$  for nodes  $s$  at a given scale. Then these estimates can be updated to include information available at that scale via:

$$\hat{x}(s|Y_s) = \hat{x}(s|Y_s^{\alpha_q}) + K(s)[y(s) - C(s)\hat{x}(s|Y_s^{\alpha_q})] \quad (4.11)$$

$$K(s) = P(s|Y_s^{\alpha_q})C^T(s)[C(s)P(s|Y_s^{\alpha_q})C^T(s) + R(s)]^{-1} \quad (4.12)$$

$$P(s|Y_s) = [I - K(s)C(s)]P(s|Y_s^{\alpha_q}) \quad (4.13)$$

The next step is a series of *predictions* in which  $q$  estimates of each state at the next level are computed based on measurements in each of the sets  $Y_{s\alpha_i}$ ,  $i = 1, 2, \dots, q$ . To describe this step, we need the upward model for the multiscale process given by (1.21) – (1.24). The upward dynamics and the updated estimates  $\hat{x}(s\alpha_i|Y_{s\alpha_i})$  from (4.11) – (4.13) are then used to compute estimates of the state at the parent node  $s$ , based on data in the subtree under each of its offspring. That is, for  $i = 1, 2, \dots, q$ , we compute:

$$\hat{x}(s|Y_{s\alpha_i}) = F(s\alpha_i)\hat{x}(s\alpha_i|Y_{s\alpha_i}) \quad (4.14)$$

$$P(s|Y_{s\alpha_i}) = F(s\alpha_i)P(s\alpha_i|Y_{s\alpha_i})F^T(s\alpha_i) + Q(s\alpha_i) \quad (4.15)$$

These  $q$  estimates are then *merged* to form another set of  $q$  estimates of the state at

node  $s$ , where the  $i^{\text{th}}$  estimate depends on the measurements at nodes  $Y_s^{\alpha_i} = \cup_{j=1}^i Y_{s\alpha_j}$ :

$$\begin{aligned}\hat{x}(s|Y_s^{\alpha_i}) &= P(s|Y_s^{\alpha_i}) \sum_{j=1}^i P^{-1}(s|Y_{s\alpha_j}) \hat{x}(s|Y_{s\alpha_j}) \\ &= P(s|Y_s^{\alpha_i}) [P^{-1}(s|Y_s^{\alpha_{i-1}}) \hat{x}(s|Y_s^{\alpha_{i-1}}) + P^{-1}(s|Y_{s\alpha_i}) \hat{x}(s|Y_{s\alpha_i})] \quad (4.16)\end{aligned}$$

$$\begin{aligned}P(s|Y_s^{\alpha_i}) &= [(1-i)P_s^{-1} + \sum_{j=1}^i P^{-1}(s|Y_{s\alpha_j})]^{-1} \\ &= [P^{-1}(s|Y_s^{\alpha_{i-1}}) + P^{-1}(s|Y_{s\alpha_i}) - P_s^{-1}]^{-1} \quad (4.17)\end{aligned}$$

Equations (4.16) – (4.17) for computing these merged estimates follow from the fact that the measurements in the sets  $Y_{s\alpha_i}, i = 1, 2, \dots, q$  are conditionally independent given  $x(s)$ .

After merging we return to the update step and in fact proceed recursively up the tree with sequences of *update-predict-merge* steps, until the root node is reached. At this point, we have obtained at each node the set of estimates  $\hat{x}(s|Y_s^{\alpha_i})$  for  $i = 1, 2, \dots, q$ .

The downward sweep begins by initializing the following estimates and error covariances at the root node ( $s = s_0$ ):

$$\hat{x}(s_0|\bar{Y}_{s_0}) = 0 \quad (4.18)$$

$$P(s_0|\bar{Y}_{s_0}) = P_0 \quad (4.19)$$

$$\hat{x}(s_0|\bar{Y}_{s_0}, Y_{s_0}^{\alpha_i}) = \hat{x}(s_0|Y_{s_0}^{\alpha_i}), \quad i = 1, 2, \dots, q \quad (4.20)$$

$$P(s_0|\bar{Y}_{s_0}, Y_{s_0}^{\alpha_i}) = P(s_0|Y_{s_0}^{\alpha_i}), \quad i = 1, 2, \dots, q \quad (4.21)$$

and computing the residual  $\nu(s_0)$  at the root node according to (4.9) – (4.10).

Next, note that we can use the recursive structure of the sets  $\bar{Y}_s$  to compute:

$$\begin{aligned}\hat{x}(s|\bar{Y}_s) &= A(s)\hat{x}(s\bar{\gamma}|\bar{Y}_s) \\ &= A(s)\hat{x}(s\bar{\gamma}|\bar{Y}_{s\bar{\gamma}\alpha_i}) \\ &= A(s)\hat{x}(s\bar{\gamma}|\bar{Y}_{s\bar{\gamma}}, Y_{s\bar{\gamma}}^{\alpha_{i-1}}) \quad (4.22)\end{aligned}$$

if  $s = s\bar{\gamma}\alpha_i$ . Similarly, the error covariance associated with this estimate is given by:

$$P(s|\bar{Y}_s) = A(s)P(s\bar{\gamma}|\bar{Y}_{s\bar{\gamma}}, Y_{s\bar{\gamma}}^{\alpha_i-1})A^T(s) + B(s)B^T(s) \quad (4.23)$$

Merging these estimates with the  $q$  estimates  $\hat{x}(s|Y_s^{\alpha_i})$  computed during the upward sweep, we obtain:

$$\hat{x}(s|\bar{Y}_s, Y_s^{\alpha_i}) = P(s|\bar{Y}_s, Y_s^{\alpha_i})[P^{-1}(s|Y_s^{\alpha_i})\hat{x}(s|Y_s^{\alpha_i}) + P^{-1}(s|\bar{Y}_s)\hat{x}(s|\bar{Y}_s)] \quad (4.24)$$

$$P(s|\bar{Y}_s, Y_s^{\alpha_i}) = [P^{-1}(s|Y_s^{\alpha_i}) + P^{-1}(s|\bar{Y}_s) - P_s^{-1}]^{-1} \quad (4.25)$$

The estimate  $\hat{x}(s|\bar{Y}_s, Y_s^{\alpha_i})$  is then used in the subsequent orthogonalization step given by (4.9) – (4.10), while the  $q$  estimates  $\hat{x}(s|\bar{Y}_s)$ ,  $\hat{x}(s|\bar{Y}_s, Y_s^{\alpha_i})$ ,  $i = 1, 2, \dots, q - 1$  are propagated down the tree according to (4.22) – (4.23). At the end of the downward sweep we have obtained  $\nu(s)$  and  $\Lambda_{\nu(s)}$  at each node  $s$ , and (4.2) can then be used to compute the associated likelihood.

In our discussion of the multiscale models in Chapter 1, we assumed that the dimension of the state, driving noise and measurements was constant over the tree, i.e.,  $x(s) \in \mathcal{R}^n$ ,  $w(s) \in \mathcal{R}^m$  and  $y(s) \in \mathcal{R}^p$  for all  $s$ . However, this assumption is not used in the algorithmic development and indeed, our algorithm applies to this more general case. The only equation that needs to be modified is (4.2), which becomes:

$$\mathcal{L}(\theta) = -\frac{1}{2} \sum_{s \in \mathcal{T}} [p_s \log 2\pi + \log |\Lambda_{\nu(s)}| + \nu^T(s)\Lambda_{\nu(s)}^{-1}\nu(s)] \quad (4.26)$$

reflecting the possibility that the measurement dimension may depend on  $s$ . It is also straightforward to generalize the algorithm to *non-homogeneous* tree structures, i.e., those in which different nodes may have different numbers of offspring, and to situations in which the state covariance matrix is not invertible (see Appendix D.1). These generalizations are not merely of theoretical interest, since, as we have seen in Chapter 3, multiscale models of this sort naturally arise (cf. the comments after the discussion of the approximate GMRF representation in Chapter 3).

Finally, we discuss the complexity of the algorithm as a function of the model



parameterization and order of the tree. Assume now that the model dimensions are fixed, with  $x(s) \in \mathcal{R}^n$ ,  $w(s) \in \mathcal{R}^m$  and  $y(s) \in \mathcal{R}^p$ . Using the approximation that inversion of an  $N \times N$  symmetric matrix requires  $N^3/3$  floating point operations (flops) we can calculate the number of flops required by the algorithm in various situations. In particular, if we assume that the measurements  $y(s)$  are available at *all* nodes on the tree, then the algorithm requires

$$8n^3 + \frac{2}{3}p^3 + 7n^2p + 4np^2 + 6np + 5p^2 + 19n^2 \quad (4.27)$$

flops per node (there are  $(q^l - 1)/(q - 1)$  nodes on a tree with  $l + 1$  scales). On the other hand, if measurements are available only at the finest level, then the algorithm requires

$$\frac{q-1}{q}(n^3 + \frac{2}{3}p^3 + 7n^2p + 4np^2 + 6np + 5p^2) + 7n^3 + 19n^2 \quad (4.28)$$

flops per node. Note that the total per-node computation in either case is not a function of the number of nodes, assuming that the order of the tree is a fixed parameter of the model. Also, the numbers above are in some sense a worst-case complexity analysis — the structure of the model can often be exploited to substantially reduce the required computation, e.g. in the context of the multiscale model used for computing optical flow in Chapter 2, we can use the fact that the dynamics (2.20) are diagonal. Likewise, in the texture discrimination application in the next section in which we use the approximate multiscale models defined in Chapter 3, simplification results from the fact that the matrices  $A(s)$  given in (3.143) – (3.146) have large blocks of zeros.

### 4.3 A Texture Discrimination Application

The classical texture discrimination problem is the following. Given a set of texture models and a set of noisy random field observations, choose the model which corresponds most closely in some sense to the data. When statistical models are available

for the textures and the measurements, this problem can be formulated as a likelihood ratio test, and that is the approach we take in this section. In particular, we will utilize GMRF models, and multiscale representations of these, as a basis for texture representation, and will examine the relative performance of a GMRF-based LRT, a multiscale model based LRT and a minimum-distance classifier approach developed in [23] to texture classification. Our analysis will be based on synthetic random field measurements which correspond to noisy realizations of either the multiscale or GMRF texture models. We wish to demonstrate several things:

1. The probability of error performances of the likelihood-based approaches generally differ by about 5%, with the superior method being dependent on whether the measurements are generated using multiscale or GMRF models. This provides quantitative evidence of the visual similarity noticed in Chapter 3, and suggests that in real applications in which neither model is correct, neither framework will provide performance which is substantially superior to the other.
2. The multiscale approach to texture classification is applicable in situations in which the GMRF-based LRT's are often too computationally demanding to be practical, e.g. in situations in which the measurement data are available over irregularly shaped regions or if only incomplete random field measurements are available.
3. The minimum-distance classifier approach, while providing generally the computationally most efficient approach to classification, provides probability of error performance which is substantially worse than that of the two likelihood based approaches. Moreover, like the GMRF-based approach, it cannot effectively take advantage of incomplete measurement data.

### 4.3.1 The Texture Discrimination Problem

The noisy random field measurements on which the hypothesis test is based are of the form:

$$y(i, j) = c(i, j)z(i, j) + v(i, j) \quad (4.29)$$

where  $v(i, j) \sim \mathcal{N}(0, r(i, j))$ ,  $c(i, j)$  is a spatially varying gain and  $z(i, j)$  is a realization of a random field generated using (3.160) or a multiscale model. The possibility of spatially varying measurement gain  $c(i, j)$  can be used to capture the possibility that measurements are available over only a subset  $\mathcal{D}$  of the image lattice. In this case one simply sets  $c(i, j) = 1, (i, j) \in \mathcal{D}$  and  $c(i, j) = 0$  otherwise. We focus here on a binary hypothesis testing problem and denote the parameters of the multiscale models as  $\theta_0^{mm}$  and  $\theta_1^{mm}$ , and the parameters of the GMRF models as  $\theta_0^{gmrf}$  and  $\theta_1^{gmrf}$ . As is well known, the likelihood ratio test (LRT) given by:

$$\begin{array}{ccc} & \text{Choose } \theta_0^{mm} & \\ \log \frac{p_{y|\theta_0^{mm}}(Y|\theta_0^{mm})}{p_{y|\theta_1^{mm}}(Y|\theta_1^{mm})} & \begin{array}{c} > \\ < \end{array} & \log \eta \\ & \text{Choose } \theta_1^{mm} & \end{array} \quad (4.30)$$

results in optimum performance for the discrimination problem when (4.29) corresponds to measurements of a realization of a multiscale texture model. A similar test, based on  $\theta_0^{gmrf}$  and  $\theta_1^{gmrf}$  is, of course, optimal when the measurements correspond to a GMRF. We refer below to the LRT based on the GMRF and multiscale models as the GMRF-based and multiscale model (MM)-based approaches to texture classification, respectively. The parameter  $\eta$  can be chosen either via a Neyman-Pearson type criterion, or based on choosing prior probabilities for, and a set of costs associated with making correct or incorrect decisions under, the two hypothesis. For the examples presented here, we simply used the maximum-likelihood philosophy, corresponding to a choice of  $\eta = 1$ .

The probability of error performance of the two approaches cannot be calculated

$h_{k,l}$	$\theta_0^{gmrf}$	$\theta_1^{gmrf}$	$h_{k,l}$	$\theta_0^{gmrf}$	$\theta_1^{gmrf}$
(1,0)	.3795	.5341	(0,2)	-.0494	-.0432
(0,1)	.4528	.4135	(-1,2)	-.0037	-.0120
(-1,1)	-.1117	-.1831	(1,2)	.0098	.0111
(1,1)	-.1548	-.2050	(-2,1)	.0086	.0362
(2,0)	-.0566	-.1229	(2,1)	.0233	.0442

Table 4.2: GMRF model coefficients. The model given by  $\theta_0^{gmrf}$  corresponds to pigskin, whereas  $\theta_1^{gmrf}$  corresponds to sand.

in closed form due to the size of the problem, and the fact that the likelihood is a quadratic function of the data. Hence, we will draw conclusions based on Monte-Carlo results for a number of model pairs, image lattice sizes and noise levels. With regard to the choice of model pair, a number of GMRF models corresponding to natural textures are proposed in [74]. As observed there and in [10], two of these generate realizations which are quite similar visually. The parameters of these two models, which correspond to pigskin and sand, are given in Table 4.2, and realizations are shown in Figure 4-7. The specific results in this paper correspond to these two GMRF's, and in fact to the family of models given by:

$$\theta_\omega^{gmrf} = (1 - \omega)\theta_0^{gmrf} + \omega\theta_1^{gmrf} \quad (4.31)$$

with  $0 \leq \omega \leq 1$  and to a complementary family of multiscale approximate representations of the GMRF family<sup>1</sup>. The motivation behind choosing a family of models is that we want to illustrate that the performance characteristics of the MM and GMRF based approaches are comparable as the contrast or distance between the model choices varies. For instance, if we are trying to distinguish between observations coming from  $\theta_\omega^{mm}$  and  $\theta_1^{mm}$ , this task is increasingly difficult as  $\omega \rightarrow 1$ . We will also present the results of experiments in which, for a given pair of models, the additive noise power is varied.

---

<sup>1</sup>We have not included the GMRF parameter  $\sigma^2$  in Table 4.2 since this parameter was changed as a function of  $\omega$  and the lattice size in order to equalize the mean square values of the random fields.

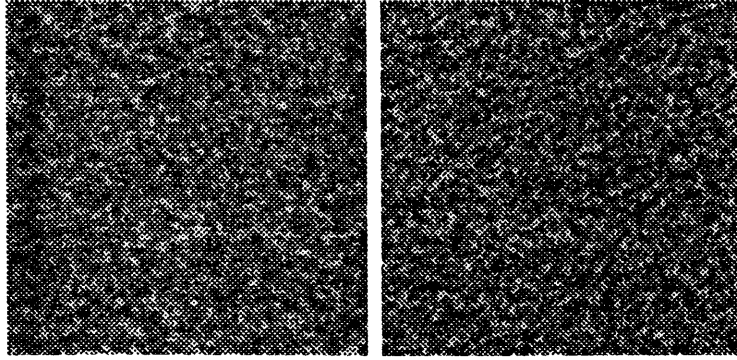


Figure 4-7: Sample paths of two GMRF texture models. The left image corresponds to the pigskin model, and the right to a model representing sand.

### 4.3.2 Performance on Rectangular Domains

To demonstrate that the GMRF-based and multiscale model-based approaches to texture discrimination result in similar performance, we first compare their performance in the case that  $r(i, j) \equiv r$ ,  $c(i, j) \equiv c$ , since in this case discrete Fourier transforms can be used to calculate the likelihoods required in (4.30). In particular, from (3.160) we can write:

$$Hz = e \quad (4.32)$$

where  $z$  and  $e$  are lexicographic orderings of the random field and driving noise values, respectively, and  $H$  is a matrix characterized by  $\theta$ . Note that  $e \sim \mathcal{N}(0, H)$  and hence  $z \sim \mathcal{N}(0, H^{-1})$  and, assuming that (4.29) corresponds to measurements of a GMRF,  $y \sim \mathcal{N}(0, (c^2 H^{-1} + rI)^{-1})$ . As discussed in [24, 23],  $H$  is a block circulant matrix, with circulant blocks and therefore can be diagonalized by the 2-D Discrete Fourier Transform (DFT) matrix  $F$ . Defining  $\bar{H} \equiv c^2 H^{-1} + rI$ , it is clear that  $\bar{H}$  also is diagonalized by the 2-D DFT, and we can write the likelihood function of  $y$  as:

$$\mathcal{L}(\theta) = \frac{-1}{2}(M_1 M_2 \log 2\pi + \log |\bar{H}| + Y^T \bar{H}^{-1} Y) \quad (4.33)$$

$$= \frac{-1}{2}(M_1 M_2 \log 2\pi + \log |F \bar{H} F^*| + Y^T F^* (F \bar{H} F^*)^{-1} F Y) \quad (4.34)$$

which can be used to efficiently carry out the GMRF-based LRT.

To implement the MM-based LRT in (4.30), we need to make a choice of *which* multiscale models to use. We choose below the simplest of the multiscale approximate models corresponding to a given GMRF, which is a zeroth-order Haar-transform based model. The state dimension  $n$  of this model is 16, and the measurement dimension at the finest scale is also 16 (in this application, the measurements are only available at the finest scale). Assuming that  $M_1 = M_2 = M$ , tedious but straightforward calculations show that in this case the algorithm of Section 4.2 requires approximately 8000 flops per pixel. The likelihood corresponding to the GMRF model can be computed using 2D-FFT approaches in  $\mathcal{O}(M^2 \log M)$  computations, which in this case leads to less computation than the MM-based approach.

The results of experiments in which we generated measurements according to (4.29), and then carried out three approaches to classification are given in Figures 4-8 and 4-10. In particular, in Figure 4-8, each data point corresponds to 1000 experiments in which we generated a random field corresponding to  $\theta_{\omega}^{gmrf}$  or  $\theta_1^{gmrf}$  (500 experiments each) for a  $32 \times 32$  lattice and signal-to-noise ratio of 0 dB, and then implemented the MM-based, GMRF-based and minimum-distance (MD)-classifier approaches to texture classification (our implementation of the MD-classifier is discussed in Appendix D.2). The percentages of correct classifications we have plotted are estimates of the probabilities of correct classification. We can characterize the error in these estimates by noting that since the experiments were independent, the probability we are trying to estimate is just the probability of success (correct classification)  $p$  in a Bernoulli test. In particular, let us define the sample mean as  $\hat{p} = N_c/N$ , where  $N_c$  is the number of correct classifications in  $N$  trials. Then a simple application of the central limit theorem allows us to show that the probability distribution for the difference between our estimate  $\hat{p}$  and the true probability of success  $p$  is normally distributed, with zero mean and variance  $p(1-p)/N$ . From this we can conclude that, for instance:

$$\Pr(|p - \hat{p}| < 1.96\sqrt{\frac{p(1-p)}{N}}) = 0.95 \quad (4.35)$$

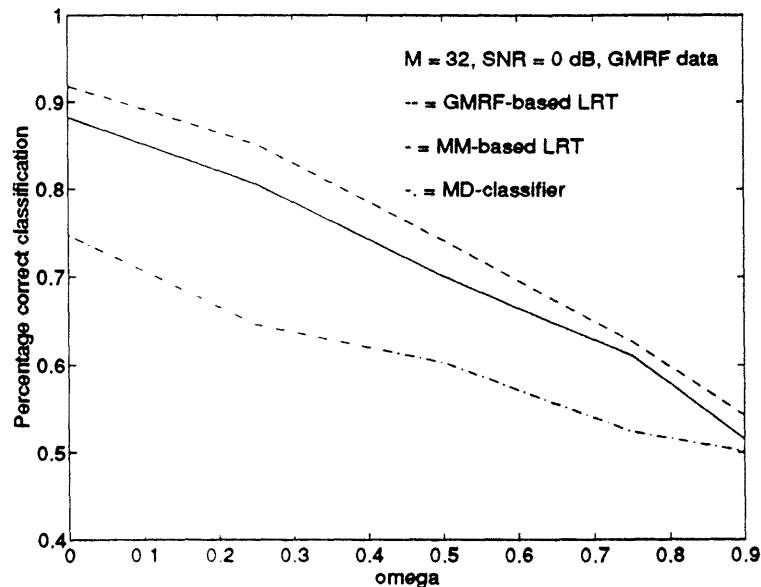


Figure 4-8: Comparison of the MM-based, GMRF-based and MD-classifier approaches to texture classification. The random field measurements correspond to GMRF texture models.

i.e., if  $p = 0.5$ , with 95 percent confidence the error in  $\hat{p}$  is less than 0.031.

Note that, as expected, as  $\omega \rightarrow 1$ , the percentage of correct classifications approaches 50 percent, reflecting the increasing similarity of the models. In Figure 4-8, the GMRF-based approach is superior to the MM-based approach, since in the experiments the measurements actually did correspond to a GMRF. The MM-based approach leads to approximately a 5% performance degradation in this case. By increasing the order of the approximate models, the performance results will become progressively closer to one another. For instance, we have performed experiments using the first and second order approximate representations discussed in Chapter 3, for  $\text{SNR} = 0$ , and  $M = 16$ . The results of these experiments (10,000 Monte-Carlo trials) are shown in Figure 4-9. The improvement in performance with increasing model order is apparent.

The performance results in Figure 4-10 are based on experiments in which the measurements correspond to multiscale texture models, and hence in this case the MM-based approach provides superior performance. The GMRF-based approach

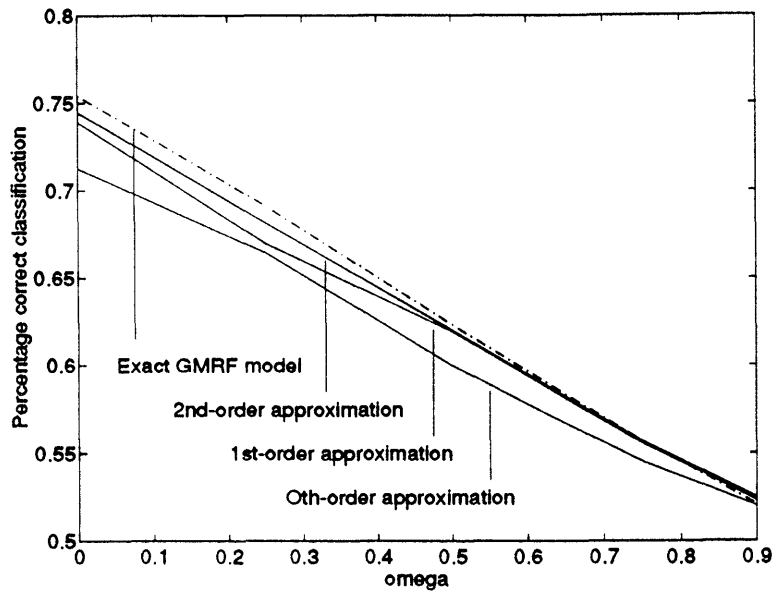


Figure 4-9: The improvement in performance as the order of the approximate GMRF representation is increased is shown.

led to a performance degradation of, on average, around 5%. Figures 4-8 and 4-10 both indicate that the MD-classifier, while requiring less computation, provides substantially poorer performance.

Our main conclusions based on Figures 4-8 and 4-10 are the following. First, the MM-based and GMRF-based approaches provide comparable performance when the data actually do correspond to a multiscale or GMRF texture model. Both of these models are idealizations, and we expect based on these results that when applied to real data, neither the MM-based or GMRF-based approach will provide a significant probability of error performance advantage. Second, we conclude that the two likelihood-based approaches provide substantially better performance than the MD-classifier approach. Further support for these conclusions is provided in Figures 4-11 to 4-16, in which we provide performance results complementing those in Figure 4-8 (the measurements correspond to realizations of the GMRF models) for a variety of SNR's and image lattice sizes. As in Figure 4-8, the figures show that the performance differential is generally three to seven percent, and that the likelihood approaches are generally superior to the MD-classifier approach.



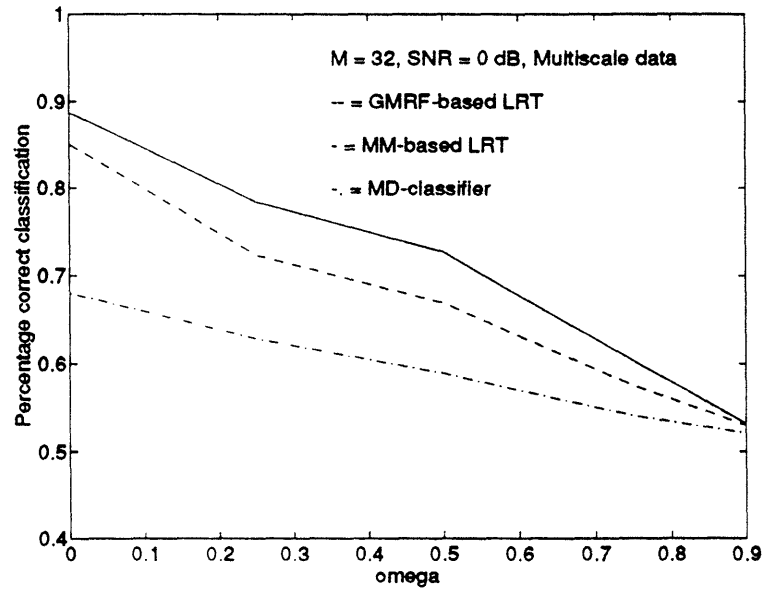


Figure 4-10: Comparison of the MM-based, GMRF-based and MD-classifier approaches to texture classification. The random field measurements correspond to multiscale texture models.

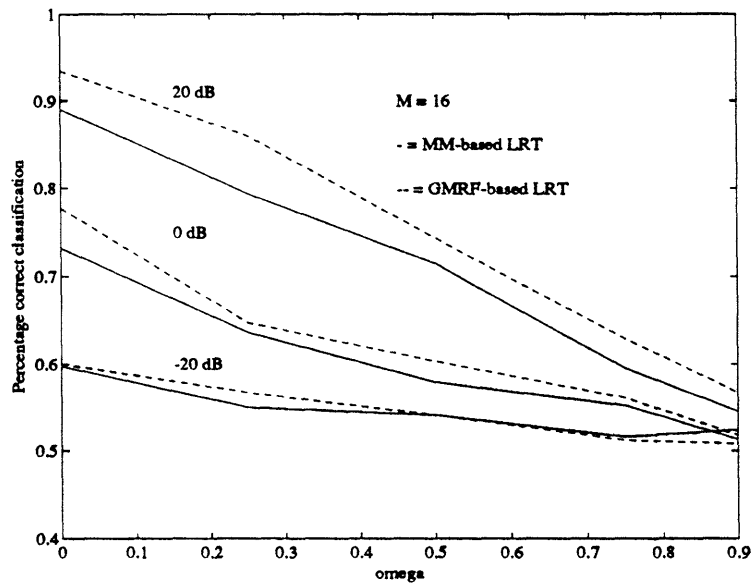


Figure 4-11: The data above and in Figures 4-13 and 4-15 further illustrate that the MM-based approach to texture classification provides performance close to that of the GMRF-based approach, which except in special cases is computationally infeasible.

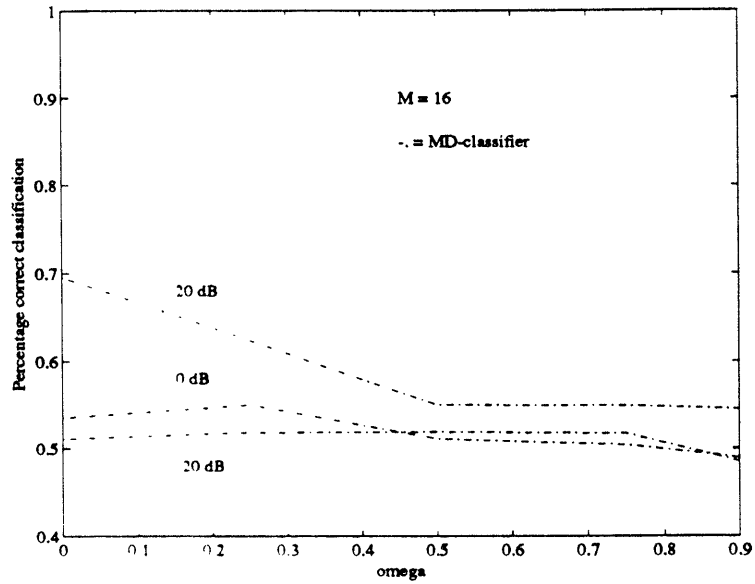


Figure 4-12: The data above and in Figures 4-14 and 4-16 illustrate the performance of the minimum-distance (MD) classifier approach to texture discrimination discussed in [23]. This approach requires only a small amount of computation, but has substantially poorer probability of error characteristics than the likelihood-based approaches.

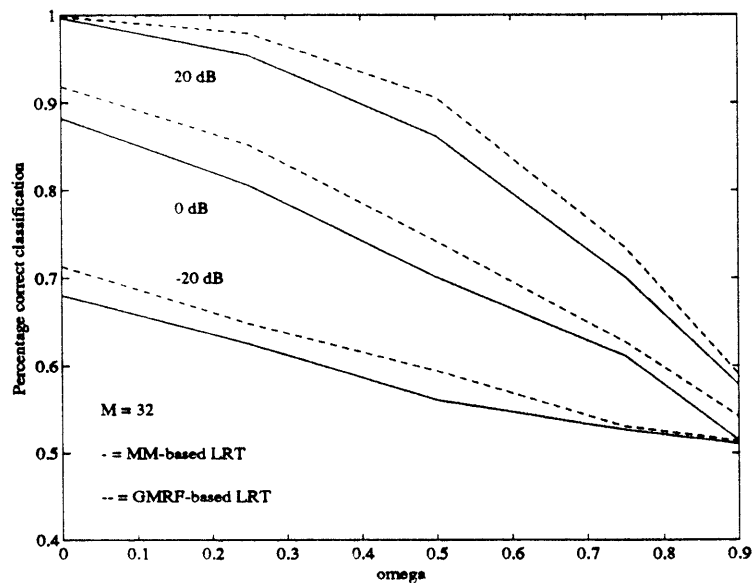


Figure 4-13: MM and GMRP-based LRT performance,  $M = 32$ .

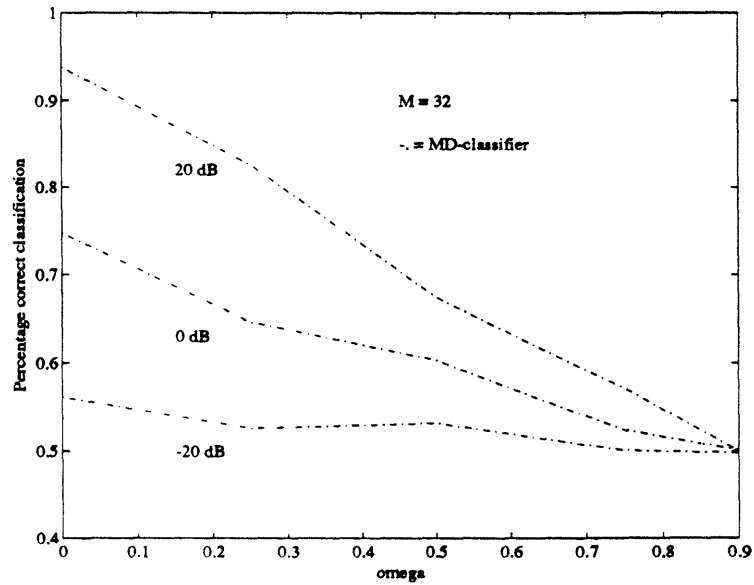


Figure 4-14: Minimum-distance classifier performance,  $M = 32$ .

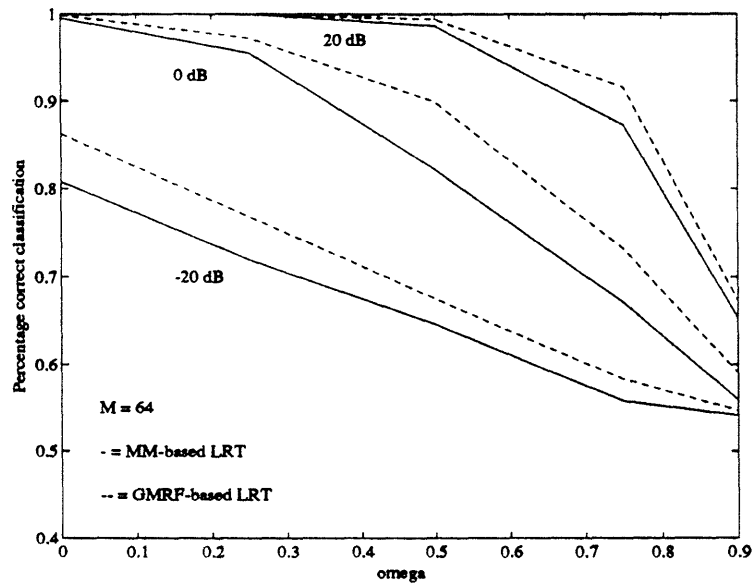


Figure 4-15: MM and GMRP-based LRT performance,  $M = 64$ .

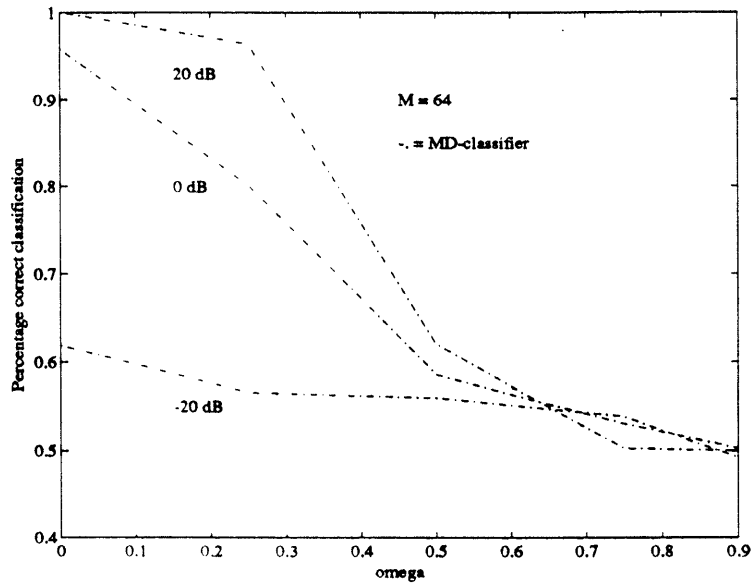


Figure 4-16: Minimum-distance classifier performance,  $M = 64$ .

### 4.3.3 Performance on Non-rectangular Regions

The results in the previous section provide substantial evidence that the MM-based and GMRF-based approaches to texture classification provide comparable performance under a variety of conditions. In this section, the results of experiments are presented which provide further evidence of this and allow us to demonstrate how our multiscale framework can be used to calculate likelihoods given measurements over only a subset of the image lattice.

In particular, note that the measurement matrix  $C(s)$  in (1.20) can vary as a function of node. In the approximate multiscale models, the values of the GMRF are represented as components of state vectors at the finest level of the tree, each value being represented in one state vector. Thus, setting  $C(s) = I$  if  $s$  is a node at the finest level corresponds to the case of *complete* measurements, i.e.  $c(i, j) = 1$  for all pairs  $(i, j)$ . Likewise, when not all measurement data are available, we can take this into account by eliminating the appropriate rows of the matrices  $C(s)$ . This is exactly what we have done in this section in which we used measurements over a circular subset of the domain, as shown in Figure 4-17, and over an incompletely

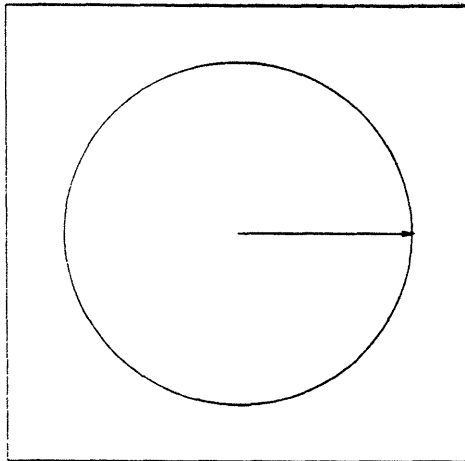


Figure 4-17: We demonstrate in Section 4.3.3 how the multiscale framework can be used to calculate likelihoods given measurements over non-rectangular regions, such as the circle above.

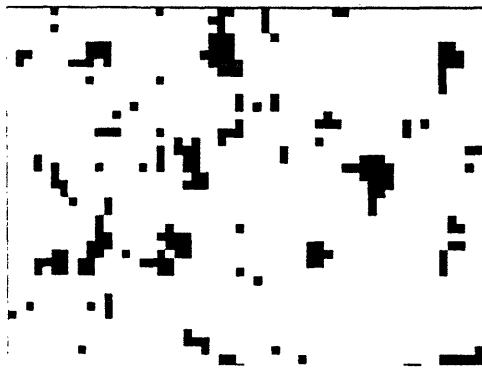


Figure 4-18: The multiscale framework can also be used to calculate likelihoods given incomplete measurements resulting from dropouts, or occluding objects. The dark areas in the figure correspond to pixels or regions over which measurements of the random field are unavailable.

sampled region as in Figure 4-18.

We begin with the experiments corresponding to the circular subset. It is assumed that the GMRF is defined over an  $M \times M$  lattice, and that the measurement gain is

an indicator function over a circle embedded in the lattice:

$$c(i, j) = \begin{cases} 1, & \text{if } (i - (M - 1)/2)^2 + (j - (M - 1)/2)^2 < \rho \\ 0, & \text{else} \end{cases} \quad (4.36)$$

Calculating the GMRF likelihood in this case is difficult since the 2-D DFT no longer diagonalizes the measurement covariance matrix. The covariance matrix must therefore be inverted directly. Using the approximation that the inversion of an  $M^2 \times M^2$  symmetric matrix requires  $(M^2)^3/3$  flops, the direct approach to the likelihood calculation requires  $M^4/3$  flops per pixel. As discussed in the previous section, the likelihood calculation algorithm developed in this chapter requires approximately 8000 flops per pixel. Hence, ignoring architectural issues, this suggests that for  $M > 12$  and non-rectangular geometries, our multiscale approach is computationally superior. We have computed the performance of the two approaches in the case that  $\rho = 8$ ,  $M = 16$  and several SNR's. The results are shown in Figures 4-19 and 4-20. The figures illustrate that the MM and GMRF-based approaches provide comparable performance, and performance which is superior to that of the MD-classifier.

Next, consider the case in which measurements are available over only a subset of the image lattice due to dropouts or objects blocking a portion of the measurement system field-of-view. Unavailable measurements correspond to black regions in Figure 4-18, and might be single pixels or groups of pixels of various sizes. This case is more difficult than the previous case from the GMRF-perspective. In particular, when measurements are available over a contiguous region as in the circular subset of Figure 4-17, one could imagine breaking up the region into a number of smaller rectangular regions, and adding up the likelihoods corresponding to these regions. On the other hand, for measurement regions such as that in Figure 4-18, this idea will be more difficult to implement since there are fewer large contiguous regions of data. Likewise, the MD-classifier approach will necessarily neglect a portion of the data since the correlation computations that are performed at each pixel require that measurement data be available at all neighbors. Hence, elimination of a single data point leads to loss of data over a region of size equal to the neighborhood of that data

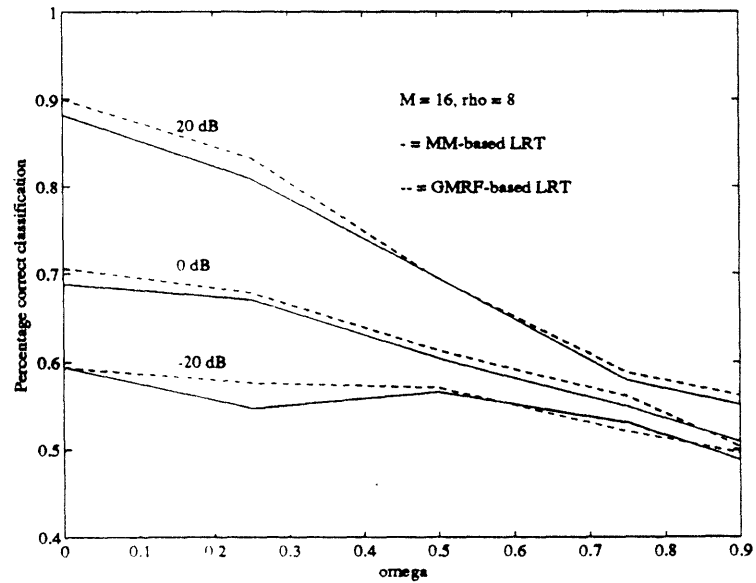


Figure 4-19: The data above illustrates the potential that the MM-based approach has for evaluating likelihoods based on data over arbitrarily shaped regions. In this case, we chose a circular region of small enough size so that the exact likelihood could also be evaluated. The MM-based approach provides performance comparable to that of the GMRF-based approach.

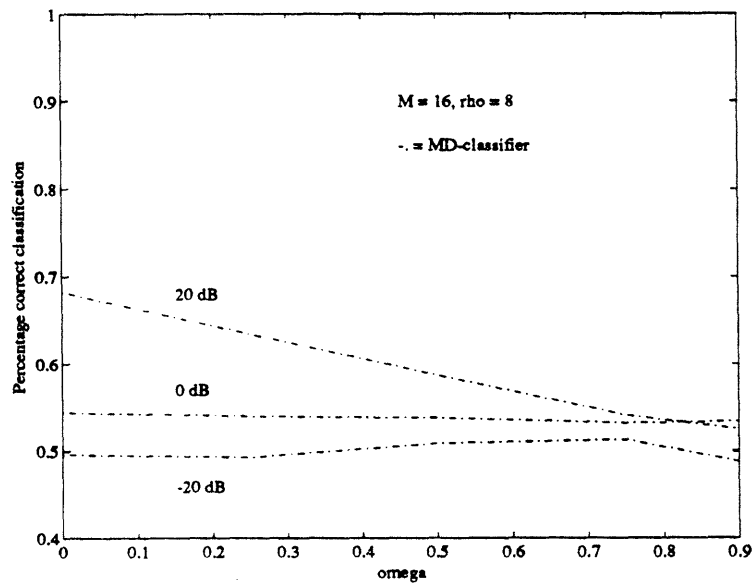


Figure 4-20: Minimum-distance classifier performance,  $\rho = 8$ .

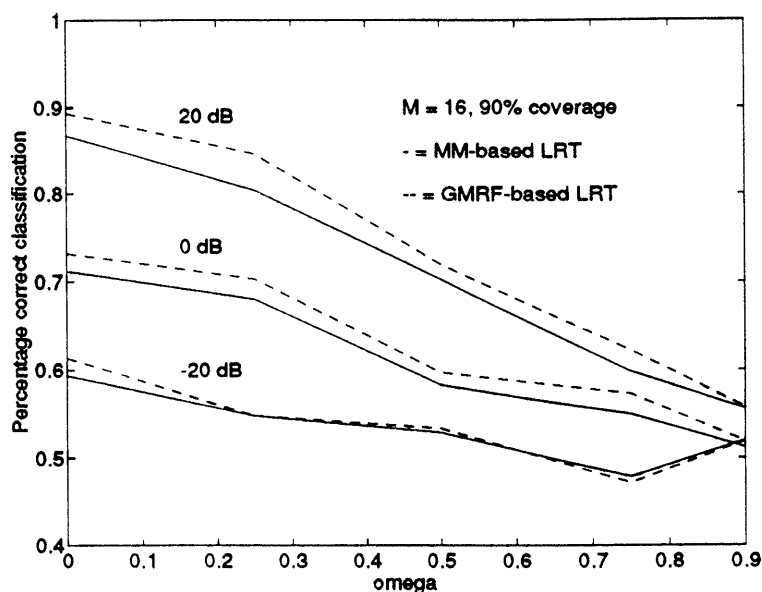


Figure 4-21: The data above illustrates the potential that the MM-based approach has for evaluating likelihoods based on incomplete measurement data.

point. This can be mitigated to some extent by interpolation, but to what extent will certainly be dependent on the specific interpolation scheme, and in any event, would be expected to only further deteriorate the already relatively poor performance of the MD-classifier.

We have computed the relative performance of the GMRF-based and MM-based approaches on domains small enough to do the exact calculations for the GMRF models. Measurements of a GMRF random field were made at 90% of the  $16 \times 16$  lattice sites, at SNR's of -20, 0 and 20 dB. The results are shown in Figure 4-21 and illustrate that the GMRF-based and MM-based approaches provide comparable performance. The MD-classifier was not applied to this example since, due to the incomplete availability of data, the correlation computations could not be carried out at an adequate number of pixels.



## 4.4 Summary

We have presented a likelihood calculation algorithm for a class of multiscale stochastic models. The algorithm exploits the structure of the tree on which the multiscale models are defined resulting in an efficient, highly parallelizable approach. In addition, we have discussed one possible application of the algorithm to texture discrimination and demonstrated that likelihood-based methods using our algorithm and the results in Chapter 3 for modeling GMRF's have substantially better probability of error characteristics than well-known least-squares methods, and achieve performance comparable to that of GMRF-based techniques, which in general are prohibitively complex computationally.

# Chapter 5

## Thesis Contributions and Recommendations for Future Research

### 5.1 Thesis Contributions

In Chapter 2, we presented a new approach to the regularization of ill-posed inverse problems, and demonstrated its potential through its application to the problem of computing optical flow. The approach starts from the “fractal prior” interpretation of the smoothness constraint introduced by Horn and Schunck to motivate regularization based on a multiscale stochastic model. This new formulation leads to an extremely efficient, non-iterative, scale-recursive solution based on the smoothing algorithm discussed in Section 1.3, yielding substantial savings over the iterative algorithms required for the smoothness constraint solution. In particular for  $256 \times 256$  or  $512 \times 512$  images, the algorithm leads to computational savings on the order of a factor of 10 to 100. Indeed, since the iterative approaches associated with the smoothness constraint solution take longer to converge as the image grows, whereas the per pixel computation associated with the MR algorithm is independent of image size, even larger savings can be realized for larger image domains. In addition to these computational advantages, we showed how the multiscale framework allows one to

select in a rational way the preferred resolution at which to represent the optical flow field.

In Chapter 3, we have shown how to represent reciprocal and Markov processes in one dimension and Markov random fields in two dimensions with a generalized multiscale model class introduced in Section 3.2. The representations in 1-D rely on a generalization of the mid-point deflection construction of Brownian motion. In 2-D, we introduced a “mid-line” construction which leads to a class of models with scale-varying dimension. Since for reasonable size fields this state dimension may be prohibitively large, we also introduced a class of multiscale *approximate* MRF representations based on 1-D wavelet transforms of the MRF along 1-D boundaries of multiresolution partitionings of the 2-D domain of interest. This family of models allows one to tradeoff complexity and accuracy of the representations. Examples demonstrated that for relatively low-order models, an approximate representation which retains most of the qualitative and statistical features of the original MRF can be obtained.

In Chapter 4, we presented a likelihood calculation algorithm for Gaussian multiscale models defined on  $q^{\text{th}}$ -order trees. We exploited the structure of the tree to obtain an efficient, highly parallelizable approach. In addition, we have discussed one possible application of the algorithm to texture discrimination and demonstrated that likelihood-based methods using our algorithm and the approximate GMRF models defined in Chapter 3 have vastly better probability of error characteristics than well-known least-squares methods, and achieve performance comparable to GMRF-based techniques, which in general are prohibitively complex computationally.

Finally, in Appendix A we derived a multiscale model for the error process associated with two-sweep smoothing algorithm. This model allows one to obtain the complete correlation structure of the smoothing error process. Applications of the result are discussed at the end of the next section.

## 5.2 Recommendations for Future Research

There are many promising avenues for further research. In the context of the chapter dealing with multiscale regularization and the computation of optical flow:

1. In situations in which the underlying optical flow field is expected to be discontinuous (or, more generally, in situations where the Brownian motion prior does not adequately represent the prior statistics) our approach may fail to give acceptable results. It is of interest to develop a framework which can accurately estimate the flow fields which arise in the presence of occlusion, transparency, etc. There are several approaches one can imagine taking. First, if there is some segmentation information available, then this could be used to modify the model on the tree in a way such that smoothness would not be enforced across borders. Secondly, one could imagine feeding back residual information, such as that illustrated in Figure 2-29, and using this in an adaptive procedure to detect and compensate for discontinuities. In particular, the measurement residual field represents a high-pass (in time) version of the observed data which accentuates the effects of motion discontinuities and removes other features corresponding to smoothly varying parts of the flow field. For time series, such residuals provide the basis for extremely effective methods for the detection of discontinuities, and the development of corresponding methods in our multiscale, image processing framework represents a promising direction for the future. Indeed, this suggests a number of additional directions for extending time-series methods to the imaging context such as adaptive estimation of the multiscale parameters  $b$  and  $\mu$  in order to adaptively adjust the level and nature of the regularization imposed on different image regions. While such adaptive methods are certainly not unknown in image processing, our scale-recursive framework not only leads to an extremely efficient framework for the realization and provides the error covariance information needed for the development of statistically optimal methods but the use of a pyramidal framework provides enormous flexibility in adaptation. For example, in the time series case, the use

of a very large value for the noise parameter corresponding to  $b$  at some point in time essentially decouples the processing before and after that point (since no smoothness at that point is expected). In our framework a large value for  $b$  at some node decouples the processing within the region, corresponding to the subtree of pixels beneath that node, from processing outside that region, exactly what would be needed to deal with a region corresponding to motion discontinuity relative to the background.

2. The multiscale solution was found to be an excellent initial guess for iterative methods employing a smoothness constraint type regularization, and we suggested a two-step procedure for obtaining that smoothness constraint solution. Another approach to obtaining the smoothness constraint solution quickly might be based on a “pre-conditioned” iterative approach. In particular, the normal equations that must be solved to obtain the smoothness constraint solution are of the form:

$$T_{sc}\hat{x} = b \quad (5.1)$$

where  $T_{sc}$  is the matrix resulting from the smoothness type regularization and  $b$  constitutes the “measurements”, i.e. from (2.10):

$$T_{sc} \equiv \mathbf{C}^T \mathbf{R}^{-1} \mathbf{C} + \mathbf{L}^T \mathbf{L} \quad (5.2)$$

$$b \equiv \mathbf{C}^T \mathbf{R}^{-1} \mathbf{y} \quad (5.3)$$

One approach to solving (5.1) is by the following iterative procedure:

$$\hat{x}_k = \hat{x}_{k-1} - T_{mr}^{-1}(T_{sc}\hat{x}_{k-1} - b) \quad (5.4)$$

where  $T_{mr}$  is the counterpart of  $T_{sc}$  arising from the multiscale regularization and  $\hat{x}_1 = T_{mr}^{-1}b$  is the MR solution. This iteration will converge if the operator  $I - T_{mr}^{-1}T_{sc}$  has all of its eigenvalues inside the unit circle, since the error  $e_k =$

$\hat{x}_k - \hat{x}$  is given by:

$$e_k = (I - T_{mr}^{-1}T_{sc})e_{k-1} \quad (5.5)$$

Some experiments of this sort were done, and it was found that in fact this operator *does not* in general have all of its eigenvalues inside the unit circle. One possible explanation for this is the following. We know from experiments in Chapter 2 that the initial error  $e_1$  contains mostly high frequency components. This energy is amplified by the differential component of  $T_{sc}$ , and further amplified by  $T_{mr}^{-1}$ . Hence, the high frequency errors tend to accumulate as the iteration proceeds.

There are several possible solutions to this problem. First, we might introduce a parameter  $\omega$  which reduces the gain:

$$\hat{x}_k = \hat{x}_{k-1} - \omega T_{mr}^{-1}(T_{sc}\hat{x}_{k-1} - b) \quad (5.6)$$

The drawback of this approach is that while the eigenvalues outside the unit circle move inside, some of the eigenvalues inside move closer to the unit circle. Indeed, as  $\omega \rightarrow 0$ , the operator becomes an identity.

Second, the problem of the high frequency error components could be addressed directly by lowpass filtering the correction term with  $T_{lp}$ :

$$\hat{x}_k = \hat{x}_{k-1} - T_{lp}T_{mr}^{-1}(T_{sc}\hat{x}_{k-1} - b) \quad (5.7)$$

Again, there is a tradeoff here which is similar to that faced in the previous example — since at least part of the error that must be eliminated is high frequency, lowpass filtering the correction term necessarily reduces the rate of convergence. Finally, a third approach would be to sequentially modify the operator corresponding to the multiscale regularization, i.e. to have an iteration

of the form:

$$\hat{x}_k = \hat{x}_{k-1} - T_{mr,k}^{-1}(T_{sc}\hat{x}_{k-1} - b) \quad (5.8)$$

where the multiscale regularization operator is a function of  $k$ . Intuitively, we should be enforcing less and less smoothness as the number of iterations increases, since the residual term is no longer expected to be locally smooth. Hence, one might choose a sequence of operators  $T_{mr,k}$  corresponding to models which require successively less smoothness. The challenging part of such an approach would be finding a sequence of models which both allows computational advantages over the two-step approach discussed in the thesis, and which also leads to a stable iterative procedure.

3. The multiresolution philosophy used in Chapter 2 may offer a promising approach to motion-compensated image sequence coding. In particular, although we used the coding metric of reconstruction error as the basis for the comparison of the SC and MR approaches, the methods presented in Chapter 2 would not be the method of choice in a coding context. In particular, motion-compensated coding algorithms designed specifically to minimize an image sequence coding metric [4, 100, 143] will generally outperform the SC and MR approaches (which are not designed for that express purpose). However, the computationally efficient MR algorithm can be used as an initial preconditioning step for such coding algorithms. In addition, one can also imagine a second way in which the multiscale framework could be used in this context. In particular, one of the problems with the SC and MR based methods is the differential form of the brightness constraint which, given the discrete nature of spatial and temporal sampling, is only valid for relatively small interframe displacements. In contrast, methods such as [4, 100, 143] use a direct displaced frame matching metric, which is nothing but the integrated version of the brightness constraint. A common approach to dealing with larger displacements with the differential brightness constraint is to spatially blur the image sequences, i.e. to consider

lower resolution versions of the image to estimate larger displacements [49, 60]. What this suggests is a multiscale approach based on (1.16) in which we not only have a multiresolution model for optical flow but also multiresolution measurements.

4. Finally, while we have focused in Chapter 2 on a particular image processing problem, the computation of optical flow, we believe that the multiscale stochastic modeling approach can be more generally useful. In particular, it may provide a computationally attractive alternative to standard approaches to the broad class of estimation problems in which the underlying field to be estimated is modeled as a Gaussian Markov random field or as the solution of noise driven partial differential equations, or in which a “smoothness constraint” type regularization is employed. Viewing the multiscale model as an alternative underlying model should lead to significant computational savings for such problems and should also have the other benefits discussed in Chapter 2, e.g. multiresolution estimates and error covariances.

In the area of Markov random field modeling via multiscale models:

1. With regard to the approximate GMRF models developed in Section 3.4, there are strong motivations for the consideration of alternatives to wavelet transforms for the approximate representations used in our multiscale models. For instance, it is certainly possible to consider non-orthogonal multiresolution approximations to the values of MRF’s along 1-D boundaries. Indeed, one possibility is to use linear or higher-order polynomial interpolation — i.e. to perform generalized mid-point deflection along each of the 1-D boundaries. Also, and perhaps more importantly, wavelet packet basis functions [144] may provide lower-dimensional approximations for some GMRF’s, such as the “wood” texture discussed in Section 3.5, in which the random field has bandpass (i.e. oscillatory) rather than low-pass characteristics in one or more directions. Indeed, techniques such as in [36], suggest the idea of choosing the “optimal” set of basis functions within some class, where optimality might be measured in terms of the compactness of



the representation, i.e. in terms of the order of the approximate model required to achieve an acceptable representation.

2. While the results in Chapter 3 demonstrate that MRF models can be represented within the multiscale model class, this alone is not enough of a reason for choosing the multiscale framework as a basis for the development of statistical signal processing algorithms and applications. In particular, although the results demonstrate conclusively that the multiscale modeling framework is quite rich, to obtain substantial computational gains over standard MRF approaches, models more parsimonious than the exact MRF representations are likely to be required. Our development and use of approximate models for GMRF's in Chapters 3 and 4 suggests that the exact MRF representations can be used to *guide* the effort to obtain such models. While we have concentrated on the Gaussian case, in our opinion the multiscale framework also holds substantial potential for the development of other, non-Gaussian models, and associated optimal signal processing algorithms, and the results of Chapter 3 provide one starting point for that effort.

In the area of likelihood calculations:

1. The texture classification application in Chapter 4 might be just the first step in an effort aimed at the development of supervised, and ultimately unsupervised, image segmentation algorithms. The richness of the multiscale model class suggests that it may be possible to develop a framework applicable to a broad range of problems. Moreover, many of the current approaches to segmentation utilize not only a stochastic model for the random field measurements given the segmentation structure (the "label field"), but also a stochastic model for the segmentation structure itself. What this suggests is a multiscale framework for segmentation based on multiscale texture models *and* multiscale label field models.
2. One important aspect of the multiscale framework yet to be developed is a set of tools for building multiscale models directly from data. The likelihood cal-

culation framework is an integral part of this set, but there are of course many others. For instance, it is quite important to identify rich subclasses of the basic model (1.16) which are in fact identifiable. That is, as is well known in the case of standard time series models, not all parameterizations lead to a unique solution of a maximum likelihood formulation of the identification problem [83]. For instance, given some set of model parameters, scaling the driving noise gain matrix  $B(s)$  by a constant factor, and scaling the measurement matrix  $C(s)$  by the inverse of that factor, leads to a different set of model parameters with an identical measurement covariance structure. In fact, there are actually many other interesting issues that arise in the context of system identification theory for multiscale models, that do not arise in the standard time-series case. For instance, the identifiability of multiscale model classes will depend on whether measurements of the process are available at several scales, or just at the finest scale. A thorough understanding of such issues will allow for much wider application of the multiscale modeling framework.

Finally, there are a number of possible applications of the smoothing error models discussed in Appendix A:

1. One of the applications of smoothing error models for the standard time series model (1.19) is in the area of updating and combining smoothed estimates as new information becomes available [11]. An entirely analogous application can be developed for our multiscale models. A problem related to updating of estimates is that of developing time-recursive multiscale models and algorithms. One may be interested in, for instance, a filtering problem in which at each point in time measurements of a multiscale random field are available and are used to *update* estimates of the random field available from previous measurements. When the time evolution of the random field is a trivial mapping, i.e. the process is assumed to be static, then this is really just a multiscale version of the updating problem mentioned above. However, when non-trivial dynamics are involved the problem becomes much more interesting. For instance, dynamics

will necessarily change the correlation structure of the multiscale process so that it may no longer correspond to our standard model (1.16). If the multiscale structure is preserved, then the estimation problem at each time step can be solved with the multiscale framework already available. Thus, it would be of interest to identify classes of dynamics which do preserve the multiscale structure of the process, or which change the dynamics in such a way that they can be well approximated within the multiscale framework.

2. The development in Chapter 2 of an approach to finding the optimal level at which to represent the optical flow field was based on the smoothing error covariances  $P^s(s)$ . Other approaches to this same problem could be developed based on the information about the correlation structure of the error process that the smoothing error model provides. For instance, one might look at the error  $\tilde{w}(s)$  in estimating the driving noise terms  $w(s)$ . By projecting both sides of the basic modeling equation (1.16) onto the space spanned by the measurements, and then subtracting this from (1.16), one obtains:

$$\tilde{x}^s(s) = A(s)\tilde{x}^s(s\bar{\gamma}) + B(s)\tilde{w}(s) \quad (5.9)$$

This can be used to obtain the error covariance of  $\tilde{w}(s)$  in terms of the error covariance matrix associated with the pair  $(\tilde{x}^s(s), \tilde{x}^s(s\bar{\gamma}))$ . Intuitively, if the error in estimating  $w(s)$  is comparable to the a-priori covariance of  $w(s)$ , then the data provide essentially no information about the scale-to-scale evolution from  $x(s\bar{\gamma})$  to  $x(s)$ , and this suggests that, for the sake of simplicity, it would be preferable to represent the optical flow field at the coarser level.

# Appendix A

## Smoothing Error Models

We derive here a multiscale model for the error process associated with the smoothed estimates of  $\hat{x}^s(s)$  (1.16). Smoothing error models have been previously derived for the smoothing error process associated with the time-recursive Gauss-Markov model (1.19) and applied to the problem of combining smoothed estimates based on independent measurements [11]. This application, as well as several discussed in the conclusions of this thesis, provide motivation for the results presented here.

Given the model (1.16) and the measurements (1.20), the two-sweep algorithm described in Chapter 1 computes, at each node  $s$  of the tree, the smoothed estimate  $\hat{x}^s(s)$ , i.e. the best estimate of  $x(s)$  based on all of the data. What we show here is that the associated error,  $\tilde{x}^s(s) \equiv x(s) - \hat{x}^s(s)$ , satisfies a multiscale difference equation of the form (1.16). We begin by noting that the error  $\tilde{x}(s|s) \equiv x(s) - \hat{x}(s|s)$  in the estimate of  $x(s)$ , based on measurements at  $s$  and in the subtree below, satisfies a certain conditional independence property. In particular, let  $\mathcal{T}_s$  be defined as the set of nodes which includes  $s$  and nodes in the subtree below  $s$ , and  $\mathcal{T}_s^c$  its complement. Then:

$$\mathbf{E}\{\tilde{x}(s|s)|\tilde{x}(\sigma|s), \sigma \in \mathcal{T}_s^c\} = \mathbf{E}\{\tilde{x}(s|s)|\tilde{x}(s\bar{\gamma}|s)\} \quad (\text{A.1})$$

where  $\tilde{x}(\sigma|s) \equiv \mathbf{E}\{x(\sigma)|Y_s\}$ . To see this, note that using the upward and downward dynamics (1.16), (1.21), we can represent  $x(\sigma), \sigma \in \mathcal{T}_s^c$  as a linear function of  $x(s\bar{\gamma})$

plus a linear function of driving noise terms. For instance:

$$x(s\bar{\gamma}^2) = F(s\bar{\gamma})x(s\bar{\gamma}) + \bar{w}(s\bar{\gamma}) \quad (\text{A.2})$$

$$x(s\bar{\gamma}^2\alpha_i) = A(s\bar{\gamma}^2\alpha_i)F(s\bar{\gamma})x(s\bar{\gamma}) + A(s\bar{\gamma}^2\alpha_i)\bar{w}(s\bar{\gamma}) + B(s\bar{\gamma}^2\alpha_i)w(s\bar{\gamma}^2\alpha_i) \quad (\text{A.3})$$

In general, we have that:

$$x(\sigma) = \Phi(\sigma, s\bar{\gamma})x(s\bar{\gamma}) + \varphi(\sigma) \quad (\text{A.4})$$

where  $\varphi(\sigma)$  is orthogonal to  $x(s\bar{\gamma})$ , the set of states  $\{x(\varsigma)|\varsigma \in \mathcal{T}_s\}$ , and the corresponding set of measurements  $\{y(\varsigma)|\varsigma \in \mathcal{T}_s\}$ . It follows then that:

$$\begin{aligned} \mathbf{E}\{\tilde{x}(s|s)|\tilde{x}(\sigma|s), \sigma \in \mathcal{T}_s^c\} &= \mathbf{E}\{\tilde{x}(s|s)|\tilde{x}(s\bar{\gamma}|s), \varphi(\varsigma), \varsigma \in \{\mathcal{T}_s^c \setminus s\bar{\gamma}\}\} \\ &= \mathbf{E}\{\tilde{x}(s|s)|\tilde{x}(s\bar{\gamma}|s)\} + \mathbf{E}\{\tilde{x}(s|s)|\varphi(\varsigma), \varsigma \in \{\mathcal{T}_s^c \setminus s\bar{\gamma}\}\} \\ &= \mathbf{E}\{\tilde{x}(s|s)|\tilde{x}(s\bar{\gamma}|s)\} \end{aligned} \quad (\text{A.5})$$

Thus, using (A.1), the upward dynamics (1.21) and the upward sweep prediction equation (1.37) we can write:

$$\tilde{x}(s|s) = P(s|s)F^T(s)P^{-1}(s\bar{\gamma}|s)\tilde{x}(s\bar{\gamma}|s) + \check{w}(s) \quad (\text{A.6})$$

where  $\check{w}(s)$  is *independent* of  $\tilde{x}(\sigma|s)$ ,  $\sigma \in \mathcal{T}_s^c$ , and has covariance:

$$P(s|s) - P(s|s)F^T(s)P^{-1}(s\bar{\gamma}|s)F(s)P(s|s) \quad (\text{A.7})$$

Next, note that the independence of  $\check{w}(s)$  and  $\tilde{x}(\sigma|s)$  for  $\sigma \in \mathcal{T}_s^c$  implies that  $\check{w}(s)$  is also independent of the *residual* information about  $x(s)$  which is contained in the set of all available measurements  $Y_0$ , but *not* contained in  $Y_s$ . In particular, at each node in  $\mathcal{T}_s^c$  a residual component  $\nu(\sigma|s)$  which is orthogonal to the measurements in

the set  $Y_s$  can be defined as:

$$\nu(\sigma|s) = y(\sigma) - \mathbf{E}\{y(\sigma)|Y_s\} \quad (\text{A.8})$$

$$= y(\sigma) - C(\sigma)\hat{x}(\sigma|s) \quad (\text{A.9})$$

Denoting  $\nu_s \equiv \{\nu(\sigma|s), \sigma \in \mathcal{T}_s^c\}$ , it is clear that

$$\text{span } Y_0 = \text{span } \{Y_s, \nu_s\} \quad (\text{A.10})$$

and that  $\nu_s \perp Y_s$ . In addition, since  $\nu(\sigma|s) = C(s)\tilde{x}(\sigma|s) + v(s)$ , we have that  $\tilde{w}(s) \perp \nu(\sigma|s)$  for all  $\sigma \in \mathcal{T}_s^c$ . Taking the expected value of both sides of (A.6), conditioned on knowledge of  $\nu(\sigma|s), \sigma \in \mathcal{T}_s^c$ , and using the orthogonality of  $\tilde{w}(s)$  and the residuals  $\nu_s$ , we obtain:

$$\mathbf{E}\{\tilde{x}(s|s)|\nu_s\} = P(s|s)F^T(s)P^{-1}(s\bar{\gamma}|s)\mathbf{E}\{\tilde{x}(s\bar{\gamma}|s)|\nu_s\} \quad (\text{A.11})$$

Subtracting (A.11) from (A.6) and using (A.10) we obtain:

$$\tilde{x}^s(s) = P(s|s)F^T(s)P^{-1}(s\bar{\gamma}|s)\tilde{x}^s(s\bar{\gamma}) + \tilde{w}(s) \quad (\text{A.12})$$

which is a multiscale model for the smoothing error of precisely the same form as (1.16). Furthermore, using the Lyapunov equation for (A.12) and defining  $J(s) \equiv P(s|s)F^T(s)P^{-1}(s\bar{\gamma}|s)$  as in (1.45), we obtain:

$$\begin{aligned} P_s(s) &= J(s)P_s(s\bar{\gamma})J^T(s) + P(s|s) - P(s|s)F^T(s)P^{-1}(s\bar{\gamma}|s)F(s)P(s|s) \\ &= P(s|s) + J(s)[P^s(s\bar{\gamma}) - P(s\bar{\gamma}|s)]J^T(s) \end{aligned} \quad (\text{A.13})$$

which agrees with the recursion in (1.44).

# Appendix B

## Appendices to Chapter 2

### B.1 Non-homogeneous Tree Structures

We made the assumption at the beginning of Section 2.2 that the image lattice is square, and that the number of rows is equal to a power of two. The reason we have done this is because of the fact that the multiscale model describing the optical flow is defined on a quadtree structure. There are at least two ways to relax the assumption. First, we could simply zero pad  $C(s)$  on the image lattice to make it fit the quadtree structure. This corresponds assuming no information is available about the (non-existent) optical flow in that region. A second, slightly more elegant approach, would be to change the modeling structure to accommodate the lattice. In particular, we would like to have a structure which gives us the proper number of nodes on the finest level. The quadtree structure is homogeneous in the sense that each parent has four offspring; what we are proposing are non-homogeneous tree structures in which different parents may have different numbers of offspring. For example, suppose one had a  $6 \times 9$  lattice. Figure B-1 illustrates a sequence of grids that one might use to model a random field defined this lattice. In the first level, the root node has six offspring, two in the row direction and three in the column direction. At the second level, each node has nine offspring, three in the row direction and three in the column direction. Thus, at the finest level there is a  $6 \times 9$  lattice. This example illustrates only one simple suggestion. More complicated tree structures could be derived, and

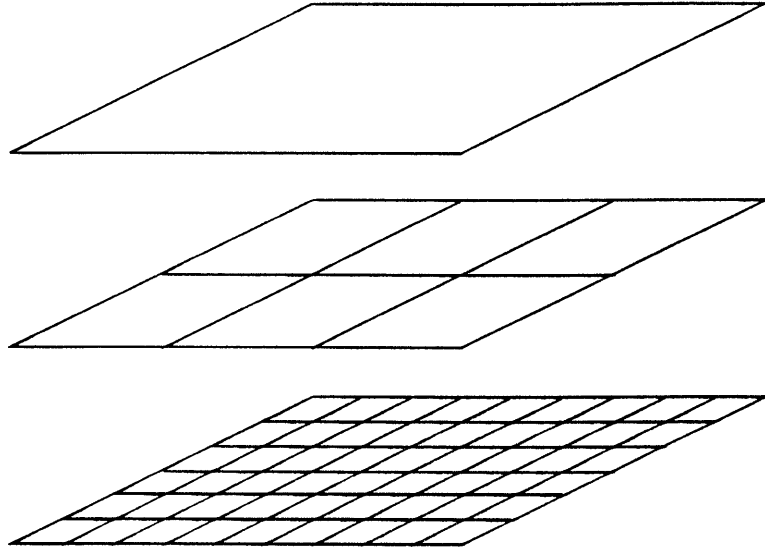


Figure B-1: Non-homogeneous tree structure for lattices which are not square. The grid structure is a simple extension of the quadtree structure in that it allows for varying numbers of “offspring” from each parent. The figure illustrates a hierarchy of grids for a  $6 \times 9$  lattice.

certainly the idea could be combined with zero padding.

## B.2 Iterative Approaches to Computation of the Smoothness Constraint Solution

We discuss in this section two iterative algorithms that can be used to compute the solution to Horn and Schunck’s smoothness constraint formulation of the optical flow problem. One can show using the calculus of variations that the solution to (2.4) satisfies the following set of coupled partial differential equations [62]:

$$\nabla^2 x_1 = R^{-1} E_{z_1} (E_t + \nabla E \cdot x) \tag{B.1}$$

$$\nabla^2 x_2 = R^{-1} E_{z_2} (E_t + \nabla E \cdot x) \tag{B.2}$$

where:

$$E_{z_1} = \frac{\partial}{\partial z_1} E(z_1, z_2, t) \tag{B.3}$$



$$E_{z_2} = \frac{\partial}{\partial z_2} E(z_1, z_2, t) \quad (\text{B.4})$$

$$E_t = \frac{\partial}{\partial t} E(z_1, z_2, t) \quad (\text{B.5})$$

$$\mathbf{x} = \begin{bmatrix} \frac{\partial z_1}{\partial t} & \frac{\partial z_2}{\partial t} \end{bmatrix}^T \quad (\text{B.6})$$

and where  $\nabla^2$  is the Laplacian operator,  $x_1$  and  $x_2$  are the first and second components of the vector  $\mathbf{x}$ , and  $R$  is the parameter controlling the tradeoff between the smoothness and data dependent terms in (2.4). Denote  $\mathbf{x}(i, j) \equiv \mathbf{x}(ih, jh, t)$  where  $h$  is the grid spacing. Discretizing (B.1), (B.2) on a uniform grid  $\{(i, j) | i \in \{1, \dots, Z_1\}, j \in \{1, \dots, Z_2\}\}$  leads to the following equations at each point:

$$\bar{x}_1 - 4x_{1,i,j} = h^2 R^{-1} E_{z_1} (E_{z_1} x_{1,i,j} + E_{z_2} x_{2,i,j} + E_t) \quad (\text{B.7})$$

$$\bar{x}_2 - 4x_{2,i,j} = h^2 R^{-1} E_{z_2} (E_{z_1} x_{1,i,j} + E_{z_2} x_{2,i,j} + E_t) \quad (\text{B.8})$$

where:

$$\begin{bmatrix} x_{1,i,j} \\ x_{2,i,j} \end{bmatrix} \equiv \mathbf{x}(i, j) \quad (\text{B.9})$$

$$\bar{x}_1 = x_{1,i-1,j} + x_{1,i+1,j} + x_{1,i,j-1} + x_{1,i,j+1} \quad (\text{B.10})$$

$$\bar{x}_2 = x_{2,i-1,j} + x_{2,i+1,j} + x_{2,i,j-1} + x_{2,i,j+1} \quad (\text{B.11})$$

The GS and SOR algorithms separate the image grid into two sets of points. These are generally referred to as the *Red points* ( $i + j$  is even) and the *Black points* ( $i + j$  is odd). The Gauss-Seidel iterations can be derived by solving (B.7) and (B.8) for  $x_{1,i,j}$  and  $x_{2,i,j}$ :

### GS Red Points

$$x_{1,i,j}^{n+1} = (\bar{x}_1^n - h^2 R^{-1} E_{z_1} (E_{z_2} x_{2,i,j}^n + E_t)) / d_{1,i,j} \quad (\text{B.12})$$

$$x_{2,i,j}^{n+1} = (\bar{x}_2^n - h^2 R^{-1} E_{z_2} (E_{z_1} x_{1,i,j}^{n+1} + E_t)) / d_{2,i,j} \quad (\text{B.13})$$

## GS Black Points

$$x_{1,i,j}^{n+1} = (\bar{x}_1^{n+1} - h^2 R^{-1} E_{z_1} (E_{z_2} x_{2,i,j}^n + E_t)) / d_{1,i,j} \quad (\text{B.14})$$

$$x_{2,i,j}^{n+1} = (\bar{x}_2^{n+1} - h^2 R^{-1} E_{z_2} (E_{z_1} x_{1,i,j}^{n+1} + E_t)) / d_{2,i,j} \quad (\text{B.15})$$

where:

$$d_{1,i,j} = 4 + h^2 R^{-1} E_{z_1}^2 \quad (\text{B.16})$$

$$d_{2,i,j} = 4 + h^2 R^{-1} E_{z_2}^2 \quad (\text{B.17})$$

The SOR algorithm is very similar to the GS algorithm, except that certain *relaxation parameters* are introduced to increase the convergence rate. The SOR iterations are given by:

## SOR Red Points

$$x_{1,i,j}^{n+1} = (1 - w_{1,i,j}) x_{1,i,j}^n + w_{1,i,j} (\bar{x}_1^{n+1} - h^2 R^{-1} E_{z_1} (E_{z_2} x_{2,i,j}^n + E_t)) / d_{1,i,j} \quad (\text{B.18})$$

$$x_{2,i,j}^{n+1} = (1 - w_{2,i,j}) x_{2,i,j}^n + w_{2,i,j} (\bar{x}_2^{n+1} - h^2 R^{-1} E_{z_2} (E_{z_1} x_{1,i,j}^{n+1} + E_t)) / d_{2,i,j} \quad (\text{B.19})$$

## SOR Black Points

$$x_{1,i,j}^{n+1} = (1 - w_{1,i,j}) x_{1,i,j}^n + w_{1,i,j} (\bar{x}_1^{n+1} - h^2 R^{-1} E_{z_1} (E_{z_2} x_{2,i,j}^n + E_t)) / d_{1,i,j} \quad (\text{B.20})$$

$$x_{2,i,j}^{n+1} = (1 - w_{2,i,j}) x_{2,i,j}^n + w_{2,i,j} (\bar{x}_2^{n+1} - h^2 R^{-1} E_{z_2} (E_{z_1} x_{1,i,j}^{n+1} + E_t)) / d_{2,i,j} \quad (\text{B.21})$$

where:

$$w_{1,i,j} = \frac{2}{1 + (1 - \rho_{1,i,j}^2)^{\frac{1}{2}}} \quad (\text{B.22})$$

$$w_{2,i,j} = \frac{2}{1 + (1 - \rho_{2,i,j}^2)^{\frac{1}{2}}} \quad (\text{B.23})$$

$$\rho_{1,i,j} = \frac{2}{d_{1,i,j}} \left( \cos \frac{\pi}{Z_1 + 1} + \cos \frac{\pi}{Z_2 + 1} \right) \quad (\text{B.24})$$

$$\rho_{2,i,j} = \frac{2}{d_{2,i,j}} \left( \cos \frac{\pi}{Z_1 + 1} + \cos \frac{\pi}{Z_2 + 1} \right) \quad (\text{B.25})$$

From (B.12), we see that each GS iteration requires 10 adds and 4 multiplies per pixel per iteration. From (B.18), we see that each SOR iteration requires 12 adds and 6 multiplies per pixel per iteration<sup>1</sup>. Thus, GS and SOR require 14 and 18 flops per pixel per iteration respectively.

## B.3 MR Algorithm Computational Complexity Analysis

In this section we analyze the computational complexity of the MR algorithm. The analysis applies to the specific model used in the optical flow application. The model is repeated here for convenience:

$$\mathbf{x}(s) = \mathbf{x}(s\bar{\gamma}) + (b4^{\frac{-\mu m(s)}{2}})w(s) \quad (\text{B.26})$$

$$\mathbf{y}(s) = C(s)\mathbf{x}(s) + v(s) \quad (\text{B.27})$$

$$w(s) \sim \mathcal{N}(0, I) \quad (\text{B.28})$$

$$v(s) \sim \mathcal{N}(0, R(s)) \quad (\text{B.29})$$

$$\mathbf{x}_0 \sim \mathcal{N}(0, pI) \quad (\text{B.30})$$

where  $R(s) = \max(\|C(s)\|^2, 10)$ . The analysis below takes into account all floating point adds, multiplies and divides.

Consider first the update step given by (1.33) – (1.36).  $P(s|s+)$  is initialized with  $pI$ . Computation of  $V^{-1}(s)$  requires 6 floating point operations (the inverse requires 1 divide since  $V(s)$  is a scalar and the comparison required to compute  $R(s)$  is not counted). Computation of  $K(s)$  requires 3 flops. Computation of  $P(s|s)$  requires 7 flops (Perform the  $C(s)P(s|s+)$  first, and use the fact that  $P(s|s)$  must be symmetric).

---

<sup>1</sup>All additions and multiplications which are redundant from iteration to iteration have been ignored. For instance,  $1 - w_{1,i,j}$  does not count as an add in (B.18) since one could compute this once at the start instead of every iteration.

Initialize  $\hat{x}(s|s+)$  with zero. Computation of  $\hat{x}(s|s)$  then requires 2 flops. The update step is required only at the finest level, since this is the only place we have data for in the optical flow problem. Thus, the total computation associated with this step is  $18 \times 4^l$  flops ( $l$  is defined to be the number of levels in the quadtree. There are  $4^l$  points at the finest level.)

Next, consider the prediction step, (1.37) – (1.38). Computation of  $P(s|\alpha_i)$  requires 5 flops (note that  $F(s)$  and  $Q(s)$  are diagonal multiples of the identity). Computation of the predicted estimate  $\hat{x}(s|\alpha_i)$  requires 2 flops. These computations must be done at levels 1 through  $l$ . Thus, the total computation associated with this step is approximately  $7 \times 4/3 \times 4^l$  flops.

Next, consider the merge step, (1.39) – (1.40). Computation of  $P(s|s+)$  requires 44 flops (there are five  $2 \times 2$  inverses requiring 6 flops apiece, and the computation of  $(1 - q)P_s^{-1}$  is negligible since it only varies with scale. The inverses require only 6 flops because the matrices involved are  $2 \times 2$  and symmetric.) Computation of  $\hat{x}(s|s+)$  requires 36 flops. The merge step must be done at levels 0 through  $l - 1$ . Thus, the total computation associated with this step is  $80 \times 1/3 \times 4^l$  flops.

Finally, consider the steps in the downward sweep, (1.43) – (1.45). Computation of  $J(s)$  requires 12 flops (the matrix  $P(s\bar{\gamma}|s)$  has already been inverted in (1.40),  $F(s)$  is a multiple of the identity and  $J(s)$  is symmetric.) Computation of  $P^s(s)$  is not required, unless one is explicitly interested in the error covariance of the smoothed estimate. Computation of  $\hat{x}^s(s)$  requires 10 flops. The smoothing step must be done at levels 1 through  $l$ . Thus, the total computation associated with this step is  $22 \times 4^l$  flops.

We can now add up all of the computations associated with the MR algorithm. There are  $4^l$  pixels in the problem domain, and thus the algorithm requires  $18 + 28/3 + 80/3 + 22 = 76$  flops per pixel. We note that this is a lower bound on the number of flops per pixel in any implementation of the algorithm and that the implementation with the lowest number of flops per pixel may not be the best. The reason is simply that there may not be enough memory available to keep all intermediate calculations around (such as the inverses computed in (1.40) and reused in (1.45)). We compute

the complexity of the GS and SOR algorithms in the same way (i.e. all intermediate results are assumed to be available), and thus the computational comparison we make between these algorithms is based on optimal (in terms of the number of flops) implementations. Suboptimal implementation of the MR algorithm will lower its computational advantage, but any reasonable implementation (for instance one which saves just  $\hat{x}(s|s)$ ,  $P(s|s)$  and the measurement data) will still provide a significant savings over the SOR and GS algorithms.

# Appendix C

## Correlation Computations for Multiscale GMRF Models

The correlations required in a  $p^{\text{th}}$ -order multiscale GMRF model require the computation of  $2^p$  1-D Fast Fourier Transforms (FFT's) per level. To see this, recall that the state  $x(s)$  of the multiscale model at any given node  $s$  is just a linear function of the values  $z(i, j)$  of the Gauss-Markov random field:

$$x(s) = Lz \tag{C.1}$$

where  $z$  is a lexicographic ordering of the GMRF values and  $L$  is a matrix. As in Chapter 3, we denote the correlation matrix of  $z$  as  $R_{zz}$  and the 2-D Fourier transform operator by  $F$ . Recalling that the 2-D Fourier transform diagonalizes  $R_{zz}$ :

$$FR_{zz}F^* = \Lambda \tag{C.2}$$

where  $\Lambda$  is a diagonal matrix containing the eigenvalues of  $R_{zz}$ , we have that:

$$P_s = (LF^*)\Lambda(FL^T) \tag{C.3}$$

For a  $p^{\text{th}}$ -order model describing an  $N \times N$  random field,  $L$  is a  $(16 \times 2^p) \times N^2$  matrix, with each row corresponding to one component of the state  $x(s)$ . Each component of

the state corresponds to the inner product of a dilated and translated mother wavelet with an appropriate subset of the state. Consider the first component of the state and the corresponding dilated and translated wavelet. Associated with that same dilated and translated wavelet are 15 other components of the state. The subsets of GMRF values corresponding to these other 15 states are just “shifted” versions of the subset of values corresponding to the first state (e.g., in Figure 3-13, consider the two states corresponding to “upward” and “downward” pointing triangles,  $\triangle$  and  $\nabla$ ) and hence, using the shifting properties of the 2-D Fourier transform, it is clear that one really only needs to compute one 2-D Fourier transform in order to calculate the components of  $FL^T$  corresponding to the first component of  $x(s)$  and its 15 counterparts which are based on the same translation and dilation of the mother wavelet. With a bit more thought, it is clear that these 2-D Fourier transforms can actually be computed with 1-D Fourier transforms, because of the fact that the components of  $x(s)$  correspond to horizontal and vertical lines in the random field lattice, which implies that each row of  $L$  can be thought of as the lexicographic ordering of a separable 2-D function defined on the image lattice. As a simple example of these points, suppose we had a random field defined on  $\{0, 1, 2, 3\} \times \{0, 1, 2, 3\}$  and that the first component of the state  $x(s)$  corresponded to the average of  $z(0, 0)$  and  $z(0, 1)$ , i.e., the first row of  $L$  was given by:

$$L_1 = [1, 1, 0, \dots, 0] \quad (\text{C.4})$$

which is just a lexicographic ordering of:

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (\text{C.5})$$

It is clear then that the 2-D Fourier transform of this array can be computed using only 1-D Fourier transforms. Likewise, if the second component of the state,  $L_2z$ ,

corresponded to the average of  $z(1,0)$  and  $z(1,1)$ , which in turn corresponds to:

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (\text{C.6})$$

then clearly the shifting properties of the 2-D FFT can be used to calculate  $FL_2^T$  directly from  $FL_1^T$ .

The situation for the first component of the state as described above is identical to that of the other  $16 \times 2^p - 1$  components of the state, hence, computation of the correlation structure of the entire state, as given by (C.3) requires  $2^p$  FFT operations. For stationary random fields, these computations are the same for states at the same level, and hence  $2^p$  FFT's are required per-level.



# Appendix D

## Appendices to Chapter 4

### D.1 Likelihood Calculations for Singular $P_s$ and

$$P(s|Y)$$

In this appendix, we discuss a generalization of the likelihood calculation algorithm which allows for singular state and state estimation error covariance matrices  $P_s$  and  $P(s|Y)$ . The need for such a generalization arises from the fact that the states of the multiscale approximate GMRF models introduced in Chapter 3 may contain redundant information about the MRF values they describe. Hence, the state covariance matrix  $P_s$  may be singular, and this will lead to singularity in the error covariance matrices  $P(s|Y)$  as well.

There are actually two ways to deal with this singularity problem. Not surprisingly, both generalizations involve the use of pseudo-inverse matrices [22]. Given a symmetric matrix  $P$  of rank  $r$ , we can decompose it as:

$$P = U\Lambda U^T \tag{D.1}$$

where  $\Lambda$  is an  $r \times r$  diagonal matrix containing the eigenvalues of the matrix  $P$ , and  $U$  is a matrix made up of the eigenvectors of  $P$  corresponding to non-zero eigenvalues.

Then, the so-called *Moore-Penrose* pseudo-inverse is given by<sup>1</sup>:

$$P^\dagger = U\Lambda^{-1}U^T \quad (\text{D.2})$$

The first, and most obvious, approach to dealing with singular  $P_s$  is to just eliminate the redundant state information so that the inverse of  $P_s$  is well defined. A decomposition of the state covariance at each node as above can be used to do this. In particular, given a decomposition:

$$P_s = U_s\Lambda_sU_s^T \quad (\text{D.3})$$

as above, define:

$$\bar{x}(s) = U_s^T x(s) \quad (\text{D.4})$$

Note that the covariance matrix of  $\bar{x}(s)$  is full rank and that, because  $x(s)$  must be an element of the column space of  $P_s$ ,  $x(s) = U_sU_s^T x(s)$ . Hence, the dynamics of the reduced order state and corresponding measurements are given by:

$$\begin{aligned} \bar{x}(s) &= U_s^T A(s)x(s\bar{\gamma}) + U_s^T B(s)w(s) \\ &= U_s^T A(s)U_{s,\bar{\gamma}}\bar{x}(s\bar{\gamma}) + U_s^T B(s)w(s) \\ &= \bar{A}(s)\bar{x}(s\bar{\gamma}) + \bar{B}(s)w(s) \end{aligned} \quad (\text{D.5})$$

$$\begin{aligned} y(s) &= C(s)U_sU_s^T x(s) + v(s) \\ &= \bar{C}(s)\bar{x}(s) + v(s) \end{aligned} \quad (\text{D.6})$$

Likewise, the second-order statistics of the state are governed by the Lyapunov equation:

$$\mathbf{E}\{\bar{x}(s)\bar{x}(s)^T\} \equiv \bar{P}_s \quad (\text{D.7})$$

---

<sup>1</sup>This definition of the pseudo-inverse applies only to symmetric matrices, which is sufficient for our purposes.

$$= \bar{A}(s)\bar{P}_{s\bar{\gamma}}\bar{A}^T(s) + \bar{B}(s)\bar{B}^T(s) \quad (\text{D.8})$$

Now, to compute the likelihood of a set of observations  $\{y(s)\}$ , the algorithm of Section 4.2.2 can be applied directly.

A second, more direct approach to dealing with singular covariance matrices, is to allow for their presence by modifying the equations describing the likelihood calculation algorithm. The first place that we run into inverses of covariance matrices is in the definition of the upwards model for the multiscale process (1.21) – (1.25), which involves  $P_s^{-1}$ . A more general backward model which allows for non-invertibility of  $P_s$  is:

$$x(s\bar{\gamma}) = \bar{F}(s)x(s) + \tilde{w}(s) \quad (\text{D.9})$$

$$y(s) = C(s)x(s) + v(s) \quad (\text{D.10})$$

where:

$$F(s) = P_{s\bar{\gamma}}A^T(s)P_s^\dagger \quad (\text{D.11})$$

$$\mathbf{E}[\tilde{w}(s)\tilde{w}^T(s)] = P_{s\bar{\gamma}} - P_{s\bar{\gamma}}A^T(s)P_s^\dagger A(s)P_{s\bar{\gamma}} \quad (\text{D.12})$$

$$\equiv Q(s) \quad (\text{D.13})$$

where  $\tilde{w}(s) \perp x(\sigma)$  if  $\sigma = s$  or  $\sigma$  is a descendant of  $s$ .

Singular covariance matrices also will be a problem in the merging formulas (4.16) – (4.17), (4.24) – (4.25). As we show below, these equations can be replaced by:

$$\begin{aligned} \hat{x}(s|Y_s^{\alpha_i}) &= P(s|Y_s^{\alpha_i}) \sum_{j=1}^i P^\dagger(s|Y_{s\alpha_j}) \hat{x}(s|Y_{s\alpha_j}) \\ &= P(s|Y_s^{\alpha_i}) [P^\dagger(s|Y_s^{\alpha_{i-1}}) \hat{x}(s|Y_s^{\alpha_{i-1}}) + P^\dagger(s|Y_{s\alpha_i}) \hat{x}(s|Y_{s\alpha_i})] \end{aligned} \quad (\text{D.14})$$

$$\begin{aligned} P(s|Y_s^{\alpha_i}) &= [(1-i)P_s^\dagger + \sum_{j=1}^i P^\dagger(s|Y_{s\alpha_j})]^\dagger \\ &= [P^\dagger(s|Y_s^{\alpha_{i-1}}) + P^\dagger(s|Y_{s\alpha_i}) - P_s^\dagger]^\dagger \end{aligned} \quad (\text{D.15})$$

and

$$\hat{\boldsymbol{x}}(s|Y_s^{\alpha_i}, \bar{Y}_s) = P(s|Y_s^{\alpha_i}, \bar{Y}_s)[P^\dagger(s|Y_s^{\alpha_i})\hat{\boldsymbol{x}}(s|Y_s^{\alpha_i}) + P^\dagger(s|\bar{Y}_s)\hat{\boldsymbol{x}}(s|\bar{Y}_s)] \quad (\text{D.16})$$

$$P(s|Y_s^{\alpha_i}, \bar{Y}_s) = [P^\dagger(s|Y_s^{\alpha_i}) + P^\dagger(s|\bar{Y}_s) - P_s^\dagger]^\dagger \quad (\text{D.17})$$

To see this, note that the subspaces spanned by  $Y_s^{\alpha_{i-1}}$  and  $Y_{s\alpha_i}$  are orthogonal conditioned on knowledge of  $x(s)$ . Hence, grouping the measurements which span these subspaces into vectors:

$$\boldsymbol{y}_s^{\alpha_{i-1}} = \text{vec}\{\boldsymbol{y}(\sigma)|\boldsymbol{y}(\sigma) \in Y_s^{\alpha_{i-1}}\} \quad (\text{D.18})$$

$$\boldsymbol{y}_{s\alpha_i} = \text{vec}\{\boldsymbol{y}(\sigma)|\boldsymbol{y}(\sigma) \in Y_{s\alpha_i}\} \quad (\text{D.19})$$

we have:

$$\begin{aligned} \boldsymbol{y}_s^{\alpha_{i-1}} &= H_s^{\alpha_{i-1}} U_s U_s^T \boldsymbol{x}(s) + \boldsymbol{v}_s^{\alpha_{i-1}} \\ &= \bar{H}_s^{\alpha_{i-1}} \bar{\boldsymbol{x}}(s) + \boldsymbol{v}_s^{\alpha_{i-1}} \end{aligned} \quad (\text{D.20})$$

$$\begin{aligned} \boldsymbol{y}_{s\alpha_i} &= H_{s\alpha_i} U_s U_s^T \boldsymbol{x}(s) + \boldsymbol{v}_{s\alpha_i} \\ &= \bar{H}_{s\alpha_i} \bar{\boldsymbol{x}}(s) + \boldsymbol{v}_{s\alpha_i} \end{aligned} \quad (\text{D.21})$$

where the additive noise terms are orthogonal,  $\boldsymbol{v}_s^{\alpha_{i-1}} \perp \boldsymbol{v}_{s\alpha_i}$ , and where, given a decomposition of  $P_s$  as in (D.3), we have defined the reduced order state as (D.4). Also, we define the covariances of the noise terms above as  $R_s^{\alpha_{i-1}}$  and  $R_{s\alpha_i}$ , respectively. Using standard formulas for the estimate of  $\bar{\boldsymbol{x}}(s)$  based on measurements of the form (D.21), we can compute:

$$\begin{aligned} \hat{\boldsymbol{x}}(s|Y_{s\alpha_i}) &= U_s \hat{\bar{\boldsymbol{x}}}(s|Y_{s\alpha_i}) \\ &= U_s (\bar{P}_s^{-1} + \bar{H}_{s\alpha_i}^T R_{s\alpha_i}^{-1} \bar{H}_{s\alpha_i}) \bar{H}_{s\alpha_i}^T R_{s\alpha_i}^{-1} \boldsymbol{y}_{s\alpha_i} \\ &= (P_s^\dagger + U_s U_s^T H_{s\alpha_i}^T R_{s\alpha_i}^{-1} H_{s\alpha_i} U_s U_s^T) H_{s\alpha_i}^T R_{s\alpha_i}^{-1} \boldsymbol{y}_{s\alpha_i} \end{aligned} \quad (\text{D.22})$$

with corresponding error covariance given by:

$$\begin{aligned}
P(s|Y_{s\alpha_i}) &= U_s \bar{P}(s|Y_{s\alpha_i}) U_s^T \\
&= U_s [U_s^T U_s (\bar{P}_s^{-1} + \bar{H}_{s\alpha_i}^T R_{s\alpha_i}^{-1} \bar{H}_{s\alpha_i}) U_s^T U_s]^{-1} U_s^T \\
&= U_s [U_s^T (P_s^\dagger + U_s U_s^T H_{s\alpha_i}^T R_{s\alpha_i}^{-1} H_{s\alpha_i} U_s U_s^T) U_s]^{-1} U_s^T \\
&= (P_s^\dagger + U_s U_s^T H_{s\alpha_i}^T R_{s\alpha_i}^{-1} H_{s\alpha_i} U_s U_s^T)^\dagger \tag{D.23}
\end{aligned}$$

Inverses of covariance matrices are often interpreted as *information* matrices, and the formula above can be interpreted in the same way. In particular, what (D.23) says is that the total information about  $x(s)$ , given by  $P(s|Y_{s\alpha_i})^\dagger$ , is equal to the prior information  $P_s^\dagger$ , plus the information in the measurements,  $U_s U_s^T H_{s\alpha_i}^T R_{s\alpha_i}^{-1} H_{s\alpha_i} U_s U_s^T$ . Note that this latter term involves the matrix  $U_s U_s^T$ , which is a projection matrix onto the column space of  $P_s$ . The interpretation of this is that, since  $x(s)$  is constrained to lie in a certain subspace, only measurements in certain directions provide new information about it.

Similar analysis can be used to show that:

$$\begin{aligned}
\hat{x}(s|Y_s^{\alpha_{i-1}}) &= \\
&= (P_s^\dagger + U_s U_s^T (H_s^{\alpha_{i-1}})^T (R_s^{\alpha_{i-1}})^{-1} H_s^{\alpha_{i-1}} U_s U_s^T) (H_s^{\alpha_{i-1}})^T (R_s^{\alpha_{i-1}})^{-1} y_s^{\alpha_{i-1}} \tag{D.24}
\end{aligned}$$

$$\begin{aligned}
P(s|Y_s^{\alpha_{i-1}}) &= \\
&= (P_s^\dagger + U_s U_s^T (H_s^{\alpha_{i-1}})^T (R_s^{\alpha_{i-1}})^{-1} H_s^{\alpha_{i-1}} U_s U_s^T)^\dagger \tag{D.25}
\end{aligned}$$

Then, using the formulae,

$$\hat{\bar{x}}(s|Y_s^{\alpha_i}) = \bar{P}(s|Y_s^{\alpha_i}) [\bar{P}^{-1}(s|Y_s^{\alpha_{i-1}}) \hat{\bar{x}}(s|Y_s^{\alpha_{i-1}}) + \bar{P}^{-1}(s|Y_{s\alpha_i}) \hat{\bar{x}}(s|Y_{s\alpha_i})] \tag{D.26}$$

$$\bar{P}(s|Y_s^{\alpha_i}) = [\bar{P}^{-1}(s|Y_s^{\alpha_{i-1}}) + \bar{P}^{-1}(s|Y_{s\alpha_i}) - \bar{P}^{-1}]^{-1} \tag{D.27}$$

and the fact that  $x(s) = U_s \bar{x}(s)$ , (D.14) – (D.15) follow. With a similar analysis (D.16) – (D.17) can also be shown to hold.

## D.2 A Minimum Distance Classifier

We discuss in this section a minimum distance classifier approach to texture discrimination, which is based on that in [23]. Suppose noise free measurements  $z(i, j)$  of a random field are available, and that we wish to choose which among a number of competing GMRF models, characterized by parameters  $\theta_p, p = 1, 2, \dots, P$ , best represents the field. Suppose further that we can estimate the model parameters directly from the data and that we call this estimate  $\hat{\theta}$ . Then the minimum distance classifier classifies the texture by choosing the model which is closest in parameter space to  $\hat{\theta}$ :

$$\text{Model class} = \arg \min_p \|\hat{\theta} - \theta_p\|_W \quad (\text{D.28})$$

where  $W$  is a diagonal weighting matrix.

The key component of such a classifier is clearly the algorithm for parameter estimation. In [24], a consistent estimation scheme is proposed for the class of GMRF models (3.160). In particular, define  $\hat{h}$  as a vector of estimates of the coefficients  $h_{k,l}$  in (3.160). Since it is required that  $h_{k,l} = h_{-k,-l}$ ,  $\hat{h}$  will have half as many elements as any lattice site has neighbors. In particular, set:

$$\hat{h} = \left[ \sum Q(i, j) Q^T(i, j) \right]^{-1} \left[ \sum Q(i, j) z(i, j) \right] \quad (\text{D.29})$$

$$\hat{\sigma}^2 = \frac{1}{M_1 M_2} \sum [z(i, j) - \hat{h}^T Q(i, j)] \quad (\text{D.30})$$

where:

$$Q(i, j) = \text{vec}[z(i - k, j - l) + z(i + k, j + l), \{(-k, -l), (k, l) \in D\}] \quad (\text{D.31})$$

and define  $\hat{\theta} = \{\hat{h}, \hat{\sigma}^2\}$ . As shown in [23], these parameter estimates are consistent, although they are not generally efficient or unbiased.

The weighting matrix  $W$  and the model class parameters  $\theta_p$  are generated during a training phase which uses the estimation procedure above. In particular, given a set of sample paths, all of which are assumed to correspond to a single GMRF model, we

estimate for each sample path the model parameters. For the  $p^{\text{th}}$  model, let us denote these as  $\theta_p^i, i = 1, 2, \dots, N$ , where  $N$  is the number of available testing samples, and also let  $\theta_p^{i,j}$  denote the  $j^{\text{th}}$  component of  $\theta^i$ . Then the  $j^{\text{th}}$  component of the “trained” parameter set  $\theta_p$  used in (D.28) is given by the sample mean  $(1/N) \sum_i \theta^{i,j}$ , and the  $j^{\text{th}}$  weight is given by the inverse of the corresponding sample covariance. In our experiments in Section 4.3, we used 1000 training experiments for each test (i.e., each set of 1000 trials corresponding to a specific lattice size, measurement domain, GMRF pair and SNR). The MD-classifier was trained on the noisy data, *not* on noise free samples of the random field. In fact, the MD-classifier does *better* when trained on noisy data, than when trained on noise free data or when given the actual parameters. This is because the MD-classifier does not take into account the presence of noise. Hence, if the comparison (D.28) is done with respect to the actual random field parameters, as the SNR falls the MD-classifier chooses the candidate which is closest to white noise, since the estimates of  $h_{k,l}$  approach zero. This effect can be offset to some extent (and better performance is thereby obtained) by training on the noisy data in the first place, which is what we have done.

# Bibliography

- [1] K. Abend, T. Harley, and L. Kanal. "Classification of binary random patterns". *IEEE Transactions on Information Theory*, 11:538–544, 1965.
- [2] J. Aggarwal and N. Nandhakumar. "On the computation of motion from sequences of images – a review". *Proceedings of the IEEE*, 76:917–935, 1988.
- [3] M. Athans and P. Falb. *Optimal Control*. McGraw-Hill, 1966.
- [4] N. Baaziz and C. Labit. "Multigrid motion estimation on pyramidal representations for image sequence coding". Technical Report 572, IRISA, February 1991.
- [5] J. Barron, D. Fleet, S. Beauchemin, and T. Burkitt. "Performance of optical flow techniques". In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, pages 236–242, 1992.
- [6] M. Basseville and A. Benveniste. *Detection of Abrupt Changes in Signals and Dynamical Systems*, volume 77 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, 1986.
- [7] M. Basseville, A. Benveniste, K. Chou, S. Golden, R. Nikoukhah, and A. Willsky. "Modeling and estimation of multiresolution stochastic processes". *IEEE Transactions on Information Theory*, 38:766–784, 1992.
- [8] M. Basseville, A. Benveniste, and A. Willsky. "Multiscale autoregressive processes, Parts 1 and 2". *IEEE Transactions on Signal Processing*, 40:1915–1954, 1992.



- [9] R. Baxter. *Exactly Solved Models in Statistical Mechanics*. Academic Press, 1982.
- [10] M. Bello. "A combined Markov random field and wave-packet transform based approach for image segmentation". Technical Report CSDL-P-3202, Draper Laboratories, 1992. Submitted to *IEEE Transactions on Signal Processing*.
- [11] M. Bello, A. Willsky, and B. Levy. "Construction and applications of discrete-time smoothing error models". *International Journal of Control*, 50:203–223, 1989.
- [12] M. Bertero, T. Poggio, and V. Torre. "Ill posed problems in early vision". *Proceedings of the IEEE*, 76:869–889, 1988.
- [13] J. Besag. "Spatial interaction and the statistical analysis of lattice systems". *Journal of the Royal Statistical Society B*, 36:192–225, 1974.
- [14] J. Besag. "On the statistical analysis of dirty pictures". *Journal of the Royal Statistical Society B*, 48:259–302, 1986.
- [15] G. Beylkin, R. Coifman, and V. Rokhlin. "Fast wavelet transforms and numerical algorithms I". *Communications on Pure and Applied Mathematics*, 44:141–183, 1991.
- [16] A. T. Bharucha-Reid. *Elements of the Theory of Markov Processes and Their Applications*. McGraw-Hill, 1960.
- [17] C. Bouman. "A multiscale image model for Bayesian image segmentation". Technical Report TR-EE 91-53, School of Electrical Engineering, Purdue University, 1991.
- [18] C. Bouman and B. Liu. "Multiresolution segmentation of textured images". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:99–113, 1991.

- [19] C. Bouman and M. Shapiro. "Multispectral image segmentation using a multiscale model". In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 565–568, 1992.
- [20] C. Bouman and M. Shapiro. "A multiscale random field model for Bayesian image segmentation". *IEEE Transactions on Image Processing*, 1993. To appear.
- [21] P. Burt and E. Adelson. "The Laplacian pyramid as a compact image code". *IEEE Transactions on Communications*, 31:482–540, 1983.
- [22] S. Campbell and C. Meyer. *Generalized Inverses of Linear Transformations*. Pitman, London, 1979.
- [23] R. Chellappa and S. Chatterjee. "Classification of textures using Gaussian Markov random fields". *IEEE Transactions on Acoustics, Speech and Signal Processing*, 33:959–963, 1985.
- [24] R. Chellappa and R. Kashyap. "Digital image restoration using spatial interaction models". *IEEE Transactions on Acoustics, Speech and Signal Processing*, 30:461–472, 1982.
- [25] T. Chin. *Dynamic Estimation in Computational Vision*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 1991.
- [26] T. M. Chin, W. Karl, and A. Willsky. "Sequential optical flow estimation using temporal coherence". Technical Report LIDS-P-2122, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems, 1992. Submitted to *IEEE Transactions on Image Processing*.
- [27] K. C. Chou. *A Stochastic Modeling Approach to Multiscale Signal Processing*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, May 1991.

- [28] K. C. Chou, S. Golden, and A. S. Willsky. "Multiresolution stochastic models, data fusion, and wavelet transforms". Technical Report LIDS-P-2110, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems, 1992. To appear in *Signal Processing*.
- [29] K. C. Chou, A. S. Willsky, and A. Benveniste. "Multiscale recursive estimation, data fusion and regularization". Technical Report LIDS-P-2085, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems, 1991. To appear in *IEEE Transactions on Automatic Control*.
- [30] K. C. Chou, A. S. Willsky, A. Benveniste, and M. Basseville. "Recursive and iterative estimation algorithms for multiresolution stochastic processes". In *Proceedings of the IEEE Conference on Decision and Control*, 1989.
- [31] K. C. Chou, A. S. Willsky, and R. Nikoukhah. "Multiscale systems, Kalman filters and Riccati equations". Technical Report LIDS-P-2152, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems, 1992. To appear in *IEEE Transactions on Automatic Control*.
- [32] S. C. Clippingdale and R. G. Wilson. "Least squares image estimation on a multiresolution pyramid". In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 1989.
- [33] F. Cohen. "Modeling of ultrasound speckle with application in flaw detection in metals". *IEEE Transactions on Signal Processing*, 40:624-632, 1992.
- [34] F. Cohen, Z. Fan, and S. Attali. "Automated inspection of textile fabrics using textural models". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:803-808, 1991.
- [35] F. Cohen, Z. Fan, and M. Patel. "Classification of rotated and scaled textured images using Gaussian Markov random field models". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:192-202, 1991.

- [36] R. Coifman and M. Wickerhauser. "Best-adapted wave packet bases". Technical report, Numerical Algorithms Research Group, Department of Math., Yale University, 1990.
- [37] R. Coifman and V. Wickerhauser. "Entropy based algorithms for best basis selection". *IEEE Transactions on Information Theory*, 38:713-719, 1992.
- [38] J. Crowley and A. Parker. "A representation for shape based on peaks and ridges in the difference of low-pass transform". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2:156-170, 1984.
- [39] I. Daubechies. "Orthonormal bases of compactly supported wavelets". *Communications on Pure and Applied Mathematics*, 91:909-996, 1988.
- [40] I. Daubechies. "The wavelet transform, time-frequency localization and signal analysis". *IEEE Transactions on Information Theory*, 36:961-1005, 1990.
- [41] M. H. A. Davis. *Linear Estimation and Stochastic Control*. John Wiley & Sons, New York, 1977.
- [42] H. Derin, H. Elliot, R. Cristi, and D. Geman. "Bayes smoothing algorithms for segmentation of binary images modeled by Markov random fields". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:707-720, 1984.
- [43] H. Derin and P. Kelly. "Discrete index Markov type random processes". *Proceedings of the IEEE*, 77:1485-1509, 1989.
- [44] R. DeVore, B. Jawerth, and B. Lucier. "Image compression through wavelet transform coding". *IEEE Transactions on Information Theory*, 38:719-747, 1992.
- [45] V. Digalakis and K. Chou. "Maximum likelihood identification of multiscale stochastic models using the wavelet transform and the EM algorithm". In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 1993.

- [46] R. Dijkerman, V. Badrinath, and R. Mazumdar. "Multiscale representation of stochastic processes using compactly supported wavelets". In *Proceedings of the IEEE International Symposium on Time-frequency and Time-scale Analysis*, 1992.
- [47] P. Dobruschin. "The description of a random field by means of conditional probabilities and conditions of its regularity". *Theory of Probability and its Applications*, 13:197-224, 1968.
- [48] P. Doerschuk. "Bayesian signal reconstruction, Markov random fields and X-ray crystallography". Technical Report CICS-P-252, Massachusetts Institute of Technology, Center for Intelligent Control Systems, 1990.
- [49] W. Enkelmann. "Investigations of multigrid algorithms for the estimation of optical flow fields in image sequences". *International Journal of Computer Vision*, 43:150-177, 1990.
- [50] P. Flandrin. "On the spectrum of fractional Brownian motions". *IEEE Transactions of Information Theory*, 35:197-199, 1989.
- [51] P. Flandrin. "Wavelet analysis and synthesis of fractional Brownian motion". *IEEE Transactions on Information Theory*, 38:910-917, 1992.
- [52] D. Fleet and A. Jepson. "Computation of component image velocity from local phase information". *International Journal of Computer Vision*, 5:77-104, 1990.
- [53] S. Geman and D. Geman. "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721-741, 1984.
- [54] B. Gidas. "A renormalization group approach to image processing problems". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:164-180, 1989.

- [55] S. Golden. *Identifying Multiscale Statistical Models Using the Wavelet Transform*. Master's thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 1991.
- [56] A. Grossman and J. Morlet. "Decomposition of Hardy functions into square integrable wavelets of constant shape". *SIAM Journal on Mathematical Analysis*, 15:726–736, 1984.
- [57] D. Gustafson, A. Willsky, J. Wang, M. Lancaster, and J. Triebwasser. "ECG/VCG rhythm diagnosis using statistical signal analysis, parts 1 and 2". *IEEE Transactions on Biomedical Engineering*, 25:344–361, 1978.
- [58] N. Haddadi and C. C. Jay Kuo. "Computation of dense optical flow fields with a parametric smoothness model". Technical Report USC SIPI 233, USC Signal and Image Processing Institute, 1992.
- [59] D. Heeger and E. Simoncelli. "Sequential motion analysis". In *Proceedings of the AAAI Robot Navigation Symposium*, March 1989.
- [60] F. Heitz and P. Bouthemy. "Multimodal estimation of discontinuous optical flow using Markov random fields". Technical Report 561, Programme 6, IRISA, January 1991.
- [61] F. Heitz, P. Perez, and P. Bouthemy. "Parallel visual motion analysis using multiscale Markov random fields". In *Proceedings of the IEEE Workshop on Visual Motion*, Princeton, NJ, October 1991.
- [62] B. K. P. Horn and B. Schunck. "Determining optical flow". *Artificial Intelligence*, 17:185–203, 1981.
- [63] E. Ising. "Beitrag zur theorie des ferromagnetismus". *Zeitschrift für Physik*, 31:253–258, 1925.
- [64] A. Jain and J. Jain. "Partial differential equations and finite difference methods in image processing — Part 2: Image restoration". *IEEE Transactions on Automatic Control*, 23:817–834, 1978.

- [65] A. K. Jain. "Advances in mathematical models for image processing". *Proceedings of the IEEE*, 69:502-528, 1981.
- [66] A. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, New York, 1970.
- [67] F. Jeng and J. Woods. "On the relationship of the Markov mesh to the NSHP Markov chain". *Pattern Recognition Letters*, 5:273-279, 1986.
- [68] T. Kailath. "Fast time-invariant implementation of Gaussian signal detectors". *IEEE Transactions on Information Theory*, 24, 1978.
- [69] T. Kailath. *Linear Systems*. Prentice-Hall, Englewood Cliffs, NJ, 1980.
- [70] T. Kailath. *Lectures on Wiener and Kalman Filtering*, volume 140 of *International Centre for Mechanical Sciences, Courses and Lectures*. Springer-Verlag, 1981.
- [71] R. Kashyap and R. Chellappa. "Estimation and choice of neighbors in spatial interaction models of images". *IEEE Transactions on Information Theory*, 29:60-72, 1983.
- [72] R. Kashyap, R. Chellappa, and A. Khotanzad. "Texture classification using features derived from random field models". *Pattern Recognition Letters*, 1:43-50, 1982.
- [73] J. Konrad and E. Dubois. "Estimation of image motion fields: Bayesian formulation and stochastic solution". In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1072-1074, 1988.
- [74] B. Kosko, editor. *Neural Networks for Signal Processing*, pages 37-61. Prentice-Hall, 1992.
- [75] C.-C. J. Kuo, B. C. Levy, and B. R. Musicus. "A local relaxation method for solving elliptic pde's on mesh connected arrays". *SIAM J. Sci. Stat. Comput*, 8:550-573, 1987.

- [76] S. Lakshmanan and H. Derin. "Simultaneous parameter estimation and segmentation of Gibb's random fields using simulated annealing". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:799–813, 1989.
- [77] S. Lakshmanan and H. Derin. "Gaussian Markov random fields at multiple resolutions". In A. Jain and R. Chellappa, editors, *Markov Random Fields: Theory and Application*. Academic Press, 1993.
- [78] M. Laurentev, V. Romanov, and S. Shishat-skii. *Ill-Posed Problems of Mathematical Physics and Analysis*. American Mathematical Society, 1986.
- [79] B. C. Levy. "Non-causal estimation for Markov random fields". In M. A. Kaachhoek, J. H. van Schuppen, and A. Ran, editors, *Proceedings of the International Symposium MTNS-89, Vol. 1: Realization and Modeling in System Theory*, Basel, 1990. Birkhauser-Verlag.
- [80] B. C. Levy. "Regular and reciprocal multivariate stationary Gaussian reciprocal processes over  $\mathcal{Z}$  are necessarily Markov". Technical report, Univ. of California Davis, Department of Electrical Engineering and Computer Science, 1991.
- [81] B. C. Levy, R. Frezza, and A. J. Krener. "Modeling and estimation of discrete time Gaussian reciprocal processes". *IEEE Transactions on Automatic Control*, 35:1013–1023, 1990.
- [82] P. Lévy. "Le mouvement Brownien". *Mem. Sc. Math., fasc. 126*, pages 1–81, 1954.
- [83] L. Ljung. *System Identification*. Prentice-Hall, 1987.
- [84] L. Ljung and T. Söderström. *Theory and Practice of Recursive Identification*. MIT Press, 1983.
- [85] S. Mallat. "A theory for multiresolution signal decomposition: The wavelet transform". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:674–693, 1989.



- [86] S. Mallat. "Multi-frequency channel decomposition of images and wavelet models". *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37:2091–2110, 1989.
- [87] S. Mallat and W. Hwang. "Singularity detection and processing with wavelets". *IEEE Transactions on Information Theory*, 38:617–643, 1992.
- [88] H. S. Malvar. *Signal Processing with Lapped Transforms*. Artech House, 1992.
- [89] B. Mandelbrot and H. Van Ness. "Fractional Brownian motions, fractional noises and applications". *SIAM Review*, 10:422–436, October 1968.
- [90] B. S. Manjunath and R. Chellappa. "Unsupervised texture segmentation using Markov random field models". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:478–482, 1991.
- [91] B. S. Manjunath, T. Simchony, and R. Chellappa. "Stochastic and deterministic networks for texture segmentation". *IEEE Transactions on Acoustics, Speech and Signal Processing*, 38:1039–1049, 1990.
- [92] A. Mariano. "Contour analysis: A new approach for melding geophysical fields". *Journal of Atmospheric and Oceanic Technology*, 7:287–295, 1990.
- [93] D. Marr and E. Hildreth. "Theory of edge detection". *Proceedings of the Royal Society of London B*, pages 187–217, 1980.
- [94] D. Marr and T. Poggio. "A computational theory of human vision". *Proceedings of the Royal Society of London, B*, pages 301–328, 1979.
- [95] J. Marroquin. "Optimal Bayesian estimators for image segmentation and surface reconstruction". Technical Report LIDS-P-1456, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems, 1985.
- [96] B. McCoy and T. Wu. *The Two-Dimensional Ising Model*. Harvard University Press, Cambridge, MA, 1973.

- [97] D. Mumford and J. Shah. "Optimal approximation by piecewise smooth functions and associated variational problems". Technical Report CICS-P-88, Center for Intelligent Control Systems, 1988.
- [98] D. Murray and B. Buxton. "Scene segmentation from visual motion using global optimization". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:220-228, 1987.
- [99] H. Nagel and W. Enkelmann. "An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:565-593, 1986.
- [100] A. Netravali and J. Robbins. "Motion-compensated television coding: Part 1". *Bell System Technical Journal*, 58:631 - 670, 1979.
- [101] R. Nikoukhah. *A Deterministic and Stochastic Theory for Two Point Boundary Value Descriptor Systems*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 1988.
- [102] L. Onsager. "Crystal statistics 1: A two-dimensional model with an order-disorder transition". *Physical Review*, 65:117 - 149, 1944.
- [103] T. Pappas. "An adaptive clustering algorithm for image segmentation". *IEEE Transactions on Signal Processing*, 40:901-914, 1992.
- [104] R. Pool. "Making 3-D movies of the heart". *Science*, 251:28-30, 1991.
- [105] B. Potamianos and J. Goutsias. "Stochastic simulation techniques for partition function approximation of Gibb's random field images". Technical Report JHU/ECE 91-02, Johns Hopkins University Department of Electrical and Computer Engineering, 1991. Also submitted to *IEEE Transactions on Information Theory*.
- [106] P. Prenter. *Splines and Variational Methods*. Wiley, 1975.

- [107] J. Prince and E. McVeigh. "Motion estimation from tagged MR image sequences". *IEEE Transactions on Medical Imaging*, 11:238–249, 1992.
- [108] L. Rabiner and R. Schafer. *Digital Processing of Speech Signals*. Prentice-Hall, Englewood Cliffs, NJ, 1978.
- [109] R. Ramanathan and O. Zeitouni. "On the wavelet transform of fractional Brownian motion". *IEEE Transactions on Information Theory*, 37:1156–1158, 1991.
- [110] S. Ranganeth. "Image filtering using a multiresolution representation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:426–438, 1991.
- [111] K. Rao and P. Yip. *Discrete Cosine Transforms – Algorithms, Advantages and Applications*. Academic Press, New York, 1990.
- [112] D. Raviv. "A quantitative approach to camera fixation". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Maui, Hawaii, 1991.
- [113] Y. Rosanov. "On Gaussian fields with given conditional distributions". *Theory of Probability and its Applications*, 12:381–391, 1967.
- [114] A. Rosenfeld and M. Thurston. "Edge and curve detection for visual scene analysis". *IEEE Transactions on Computing*, 20:512–569, 1971.
- [115] A. Rosenfeld and B. Vanderbrug. "Coarse-fine template matching". *IEEE Transactions on Systems, Man and Cybernetics*, 7:104–107, 1977.
- [116] A. Rougée, B. Levy, and A. S. Willsky. "An estimation-based approach to the reconstruction of optical flow". Technical Report LIDS-P-1663, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems, 1987.
- [117] A. Rougée, B. Levy, and A. S. Willsky. "Reconstruction of two-dimensional velocity fields as a linear estimation problem". In *Proceedings of the 1st International Conference on Computer Vision*, pages 646–650, 1987.

- [118] G. Ruckebush. "Biorthogonal wavelet decomposition of fractional Brownian motion". (Unpublished).
- [119] L. Scharf. *Statistical Signal Processing*. Addison-Wesley, 1991.
- [120] J. Schroter and C. Wunsch. "Solution of nonlinear finite difference ocean models by optimization methods with sensitivity and observational strategy analysis". *Journal of Physical Oceanography*, 16:1855–1874, 1986.
- [121] F. Scheppe. "Evaluation of Likelihood functions of Gaussian signals". *IEEE Transactions on Information Theory*, 11:61–70, 1965.
- [122] A. Shio and J. Sklansky. "Segmentation of people in motion". In *Proceedings of the IEEE Workshop on Visual Motion*, pages 325 – 332, Princeton NJ, 1991.
- [123] E. Simoncelli, E. Adelson, and D. Heeger. "Probability distributions of optical flow". In *IEEE Conference on Computer Vision and Pattern Recognition*, Maui, Hawaii, June 1991.
- [124] A. Singh. "Incremental estimation of image flow using a Kalman filter". In *Proceedings of the IEEE Workshop on Visual Motion*, pages 36–43, 1991.
- [125] M. J. T. Smith and T. P. Barnwell. "Exact reconstruction techniques for tree structured subband coders". *IEEE Transactions on Acoustics, Speech and Signal Processing*, 34:434–441, 1986.
- [126] F. Spitzer. "Markov random fields and Gibbs ensembles". *American Mathematical Monthly*, 78:142–154, 1971.
- [127] G. Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, Wellesley MA, 1986.
- [128] G. Strang. *Linear Algebra and its Applications*. Harcourt-Brace-Jovanovich, 1988.

- [129] J. Stuller and G. Krishnamurthy. "Kalman filter formulation of low-level television motion estimation". *Computer Vision, Graphics and Image Processing*, 21:169–204, 1983.
- [130] V. Sundareswaren. "Egomotion from global flow field data". In *Proceedings of the IEEE Workshop on Visual Motion*, pages 140–145, Princeton NJ, 1991.
- [131] R. Szeliski. *Bayesian Modeling of Uncertainty in Low-level Vision*. Kluwer Academic, 1989.
- [132] D. Taylor. *Parallel Estimation on One and Two Dimensional Systems*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 1992.
- [133] R. Tenney and A. Willsky. "Multi-resolution estimation for image processing and fusion". In *Proceedings of the AFIT Workshop on Wavelets and Signal Processing, TP-329*, 1992.
- [134] D. Terzopoulos. "Image analysis using multigrid relaxation methods". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:129–139, 1986.
- [135] A. H. Tewfik and M. Kim. "Correlation structure of the discrete wavelet coefficients of fractional Brownian motion". *IEEE Transactions on Information Theory*, 38:904–910, 1992.
- [136] A. Tikhonov and V. Arsenin. *Solutions of Ill-Posed Problems*. Halstead Press, John Wiley and Sons, 1977.
- [137] S. Turkoz. *Variational Procedures for  $\phi^4$  Scalar Field Theory*. PhD thesis, Massachusetts Institute of Technology, Department of Physics, 1990.
- [138] P. Vaidyanathan. "Theory and design of M-channel maximally decimated quadrature mirror filters with arbitrary M, having the perfect reconstruction property". *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35:476–492, 1987.

- [139] H. L. VanTrees. *Detection, Estimation and Modulation Theory: Part 1*. Wiley, New York, NY, 1968.
- [140] M. Vetterli. "Filter banks allowing perfect reconstruction". *Signal Processing*, 10:219–244, 1986.
- [141] M. Vetterli. "A theory of multirate filter banks". *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35:356–372, 1987.
- [142] J. Vlontzos and D. Geiger. "A MRF approach to optical flow estimation". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 853–856, 1992.
- [143] D. Walker and K. Rao. "Improved pel-recursive motion compensation". *IEEE Transactions on Communications*, 32:1128–1134, 1984.
- [144] M. V. Wickerhauser. "Lectures on wavelet packet algorithms". Technical report, Washington University Department of Mathematics, 1992.
- [145] A. S. Willsky. "A survey of design methods for failure detection in dynamic systems". *Automatica*, pages 601–611, November 1976.
- [146] A. S. Willsky. "Opportunities and challenges in signal processing and analysis". In *Proceedings of the International Conference on Computer Science and Control*, Paris, 1992.
- [147] R. Wilson and A. Bhalerao. "Kernel design for efficient multiresolution edge detection and orientation estimation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:384–390, 1992.
- [148] J. Woods. "Two-dimensional discrete Markovian fields". *IEEE Transactions on Information Theory*, 18:232–240, 1972.
- [149] J. Woods and C. Radewan. "Kalman filtering in two dimensions". *IEEE Transactions on Information Theory*, 23:473–482, 1977.

- [150] G. Wornell. "A Karhunen-Loève like expansion for  $1/f$  processes". *IEEE Transactions on Information Theory*, 36:859–861, 1990.
- [151] G. Wornell. *Synthesis, Analysis and Processing of Fractal Signals*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 1991.
- [152] G. Wornell. "Wavelet-based representations for the  $1/f$  family of fractal processes". *Proceedings of the IEEE*, September 1993.
- [153] G. Wornell and A. Oppenheim. "Estimation of fractal signals from noisy measurements". *IEEE Transactions on Signal Processing*, 40:611–623, 1992.