June 1996

LIDS-TH-2334

Research Supported By:

National Science Foundation
Graduate Research Fellowship

Air Force Office of Scientific Research
Grant AFOSR F49620-95-1-0083

Advanced Research Project Agency
Grant ARPA F49620-93-1-0604

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

Multiscale Methods for the Segmentation of Images

Michael K. Schneider

June 1996

LIDS-TH-2334

Sponsor Acknowledgments

National Science Foundation
Graduate Research Fellowship

Air Force Office of Scientific Research
Grant AFOSR F49620-95-1-0083

Advanced Research Project Agency
Grant ARPA F49620-93-1-0604

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

Multiscale Methods for the Segmentation of Images

Michael K. Schneider

This report is based on the unaltered thesis of Michael K. Schneider submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of Master of Science at the Massachusetts Institute of Technology in June 1996.

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

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

Multiscale Methods for the Segmentation of Images

by

Michael K. Schneider

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

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1996

Author .... Michael Schneider ......
Department of Electrical Engineering and Computer Science
May 17, 1996

Certified by ....
Alan S. Willsky
Professor of Electrical Engineering
Thesis Supervisor

Accepted by ....
F.R. Morgenthaler
Chairman, Department Committee on Graduate Students

Multiscale Methods for the Segmentation of Images

by

Michael K. Schneider

Submitted to the Department of Electrical Engineering and Computer Science
on May 17, 1996, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

This thesis addresses the problem of segmenting an image into homogeneous regions bounded by curves on which the image intensity changes abruptly. Motivated by large problems arising in certain scientific applications, such as remote sensing, two objectives for an image segmentation algorithm are laid out: it should be computationally efficient and capable of generating statistics for the errors in the estimates of the homogeneous regions and boundary locations.

The starting point for the development of a suitable algorithm is some previous work on variational approaches to image segmentation. Such approaches are deterministic in nature and so provide an inappropriate context for discussing error statistics. However, many variational problems lend themselves to Bayesian statistical interpretations. This thesis develops a precise statistical interpretation of a one-dimensional version of a variational approach to image segmentation. The one-dimensional segmentation algorithm that arises as a result of this analysis is computationally efficient and capable of generating error statistics. This motivates an extension of the algorithm to two dimensions.

A straightforward extension would incorporate recursive procedures for computing estimates of arbitrary Markov random fields. Such procedures require an unacceptably large number of multiplications. To meet the objective of developing a computationally efficient algorithm, the use of recently developed multiscale statistical methods for segmentation is investigated. This results in the development of segmentation algorithms which are not only computationally efficient but also capable of generating error statistics, as desired.

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

Acknowledgments

First and foremost, I'd like to express my gratitude for the help and encouragement I received from my advisor Alan Willsky. I am thankful to have had his input from the very beginning, when he suggested my studying the problem of image segmentation, to the very end, when he read draft after draft of my thesis as I simultaneously wrote up my research and learned how to write up research.

I would also like to express my thanks to Clem Karl. He has helped me greatly in focusing my investigation. In particular, he brought the work of Shah to my attention and suggested it as an appropriate starting point.

Paul Werner Fieguth deserves many thanks for helping me on my thesis. He has spent a lot of time answering my pesky questions and has simplified my programming tasks enormously by providing me with code. Furthermore, he has been a great person to have around. I would like to thank him for the many insightful conversations about society we have had and for looking just like Theodore Kaczynski.

I am also indebted to the other members of the Stochastic Systems Group, past and present, for the numerous discussions, technical and otherwise, that have helped me mature as a researcher and a person as well as buoyed my spirits during my first two years of graduate school: Hamid "I'm so proud to have a kid" Krim, Mike "I watch a movie every day" Daniel, Charlie "The Big Dog" Fosgate, Austin "The Go Master" Frakt, Ben "Jamin" Halpern, Bill "This is not vacuous posturing" Irving, Seema "I'm hungry" Jaggi, Andrew "Studious" Kim, The learned Dr. Learned, Cedric "Will it ever end?" Logan, Terrence "No!" Ho, and Ilya "Wavelet Ave: Dead End" Polyak.

Next to last, I'd like to express my sincere gratitude to Dana Buske. Her loving support and encouragement has sustained my very life for the past few years.

Finally, I would like to thank my loving parents, who brought me up so well.

This material is based upon work supported under a National Science Foundation Graduate Research Fellowship. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the author and do not necessarily reflect the views of the National Science Foundation.

Contents

1 **Introduction** 13

1.1 Contributions 14

1.2 Organization 15

2 **Background** 17

2.1 Variational Approaches to Image Segmentation 18

2.2 Multiscale Estimation Framework 21

2.3 Concluding Remarks 27

3 **Segmentation in One Dimension** 28

3.1 The Estimation Problem 28

3.1.1 Continuous version 29

3.1.2 Discrete version 31

3.2 Results 34

3.2.1 Typical Examples 35

3.2.2 Error Statistics 38

3.2.3 Edge Localization 42

3.2.4 The Noise Parameter 44

3.2.5 Concluding Remarks 51

3.3 Analysis 52

3.3.1 Parameter choices 52

3.3.2 Convexity 55

3.4 Summary of One Dimensional Results 57

4 Image Segmentation **58**

4.1 Segmentation with 1/f-Like Models 60

4.1.1 Derivation of the Algorithm 60

4.1.2 Results 65

4.1.3 Analysis 71

4.2 Segmentation with Thin Plate Models 73

4.2.1 Derivation of the Algorithm 73

4.2.2 Results 76

4.2.3 Analysis 87

4.3 Conclusion 88

5 Conclusions and Extensions **89**

5.1 Brief Summary 89

5.2 Extensions 90

A Multiscale Thin Plate Model **92**

List of Figures

2-1 An example of coordinate descent on a simple surface. 21

2-2 In the notation of this thesis, if ν is the index of some node, $\nu\bar{\gamma}$ denotes the parent of that node. 23

2-3 An example of using Paul's multiscale models to perform surface reconstruction. 26

3-1 An example of the sampling grids that would be used for the piecewise smooth function f and edge function s. 32

3-2 The one dimensional segmentation algorithm. 35

3-3 A synthetic segmentation example. 37

3-4 A step edge segmentation example. 39

3-5 These plots depict how the value of the functional converges for the examples of Figures 3-3 and 3-4. 40

3-6 The edge function used to generate the realizations in the Monte Carlo runs of Figures 3-7 and 3-12 through 3-14. 41

3-7 A comparison of various error statistics compiled using Monte Carlo techniques for segmenting synthetic data. 42

3-8 Step edges with differing amounts of measurement noise. 44

3-9 The average value of W for the step edge example of Figure 3-4 but with different levels of measurement noise added. 45

3-10 Average value of W for the step edge example of Figure 3-4 but for different values of the parameter λ. 46

3-11 Average value of W for the step edge example of Figure 3-4 but for different values of the parameter ρ. 47

3-12 A comparison of various error statistics for $P_v = 5$, $r = 1$. 48

3-13 A comparison of various error statistics for $P_v = 5$, $r = 5$. 49

3-14 A comparison of various error statistics for $P_v = 1$, $r = 5$. 50

3-15 The average value of W for the step edge example of Figure 3-4 but with different levels of measurement noise added and r kept constant. 51

4-1 General structure of the two-dimensional algorithms. 59

4-2 Progression of recursion in one-dimensional (a) and quad tree models (b). 60

4-3 Even in the absence of edges, two process values on the finest scale of a quad tree, such as f_{ν_1} and f_{ν_2}, can be physically close together but have a small correlation. 63

4-4 Using overlapping tree framework to compute estimates $\hat{f}$ with error covariance P from data g with measurement noise R. 64

4-5 Multiscale segmentation algorithm formed by using overlapping projection operations just once. 65

4-6 A circle segmentation example. 67

4-7 Mesh plots of the circle segmentation example. 68

4-8 AVHRR data of the North Atlantic on June 2, 1995 at 5:45:01 GMT. 69

4-9 An AVHRR data segmentation example. 71

4-10 Location of data drop outs for the example in Figure 4-9. 72

4-11 How discontinuities are incorporated into the thin plate multiscale model. 75

4-12 A circle segmentation example computed using the multiscale thin plate approach. 79

4-13 Mesh plots of the circle segmentation example computed using multiscale thin plate models. 80

4-14 Surfaces comprising the data for the example in Figure 4-15. 81

4-15 Segmentation of a semicircular steps surface using the multiscale thin plate approach. 82

4-16 A circle segmentation example with data drop outs. 83

4-17 Locations of data drop outs for the circle example in Figure 4-16. . . 83

4-18 Segmentation of AVHRR data using the multiscale thin plate approach. 85

4-19 Segmentation of AVHRR data containing a warm core ring. 86

4-20 Locations of data drop outs for the AVHRR data example in Figure 4-19. 86

A-1 How nodes are numbered in the thin plate model. 93

List of Tables

3.1 Parameter values for the synthetic example in Figure 3-3. 36

3.2 Parameter values for the step edge example in Figure 3-4. 38

3.3 Values of P_v and r used to characterize the error statistics. 45

4.1 Parameter values for the synthetic example in Figures 4-6 and 4-7. . . 66

4.2 Parameter values for the AVHRR data example in Figure 4-9. 70

4.3 Parameter values for the synthetic examples in Figures 4-12 and 4-13, 4-15, and 4-16. 78

4.4 Parameter values for the AVHRR segmentation in Figures 4-18. . . . 84

4.5 Parameter values for the AVHRR segmentation in Figure 4-19. 84

Chapter 1

Introduction

Many imaging applications require segmenting an image, often corrupted by noise, into homogeneous regions bounded by curves on which the image intensity changes abruptly. In some cases, the principal interest is in obtaining an estimate of the boundaries. In others, the primary goal is to obtain an estimate of the homogeneous regions, which provides one with an estimate of the underlying phenomenon being imaged without undesirable smoothing across edges. Currently, the process of marking the boundaries is often done painstakingly by hand. To reduce the tedium of this component of image analysis, one would like to develop an algorithm that would allow a computer to automatically generate estimates of the boundaries and of the homogeneous regions. For many scientific applications, such as remote sensing, one is interested in obtaining not only the estimates but also statistics for the errors in the estimates.

There have been many approaches to the problems of edge detection and segmentation [9, 11, 13, 14, 17, 18]. Of interest here are methods based on models that lend themselves to relatively simple statistical interpretations. In particular, the starting point for the work in this thesis is a variational formulation of the segmentation problem. A variational approach to a general problem poses it as the minimization of a functional over a class of admissible functions. In a segmentation formulation, the class of admissible functions are segmentations (boundaries and homogeneous regions), and the functionals incorporate some criteria for what constitutes a good segmentation. One can prove that minimizers of the functionals posed for segmentation exist and have many nice mathematical properties that are intuitively appropriate for a segmentation. However, this deterministic approach to segmentation does not lend itself to a discussion of error statistics. Problems formulated in the variational context can often be rewritten into equivalent Bayesian estimation problems. Such problems are specified by a prior model and a measurement equation. In this context, it is natural to discuss the statistics of the errors in the estimates, as one would like to do in the segmentation problem.

The Bayesian estimation problems that arise from a study of variational problems typically involve the use of a particular class of priors, Markov random field priors. Unfortunately, the use of such priors leads to estimation algorithms that require an unacceptably large number of computations to generate estimates and error statistics. Previous work has addressed the use of another class of prior models, multiscale prior models, to formulate problems which are often posed in a variational context [6, 8, 15, 16]. The use of multiscale priors leads to computationally efficient estimation algorithms that generate good results. This thesis investigates the use of multiscale models to develop a segmentation algorithm that is

- computationally efficient (constant computational complexity per pixel)
- and capable of generating error statistics.

1.1 Contributions

Shah has proposed a particular variational formulation of the segmentation problem [20] whose structure is amenable to a statistical interpretation. One of the contributions of this thesis is providing a precise statistical interpretation of the one-dimensional version of Shah's variational approach to segmentation. This, in turn, leads to an exploration of various statistical properties of the segmentation formulation. Developing statistical interpretations of variational problems is not new. However, the particular variational approach to segmentation addressed in this thesis has not been previously cast into a statistical framework.

The other principal contribution of this thesis is the incorporation of multiscale models into an image segmentation algorithm. A variety of multiscale models have been created for other problems in computer vision such as surface reconstruction [6, 8]. Two such models are used to formulate two different image segmentation algorithms. Although neither of the models are developed from scratch, one of them requires some modifications so as to be appropriate for use as a model in a segmentation algorithm. Both multiscale models are proven useful for segmentation.

1.2 Organization

Chapter 2 provides some background for the subsequent chapters. It introduces the use of variational methods for general problems in computer vision and presents Shah's variational approach to segmentation that forms the basis for most of the work in this thesis. This is followed by a discussion of the relationship between variational and statistical approaches to problems in computer vision. Finally, an overview of multiscale models is presented in the context of statistical approaches to computer vision problems.

Chapter 3 develops a statistical interpretation of the one-dimensional version of the variational approach to segmentation presented in Chapter 2. A set of estimation problems associated with this variational approach to segmentation is derived. It forms the core of a one-dimensional segmentation algorithm. Some numerical results are presented in order to analyze the performance of this algorithm. These include some typical examples and some Monte Carlo simulations designed to characterize the parameters of the algorithm and the statistical properties of the estimates of the edges and homogeneous regions of an image. The chapter concludes with some simple calculations which further develop a physical understanding of the parameters in the algorithm as well as examine how the parameters affect the convexity of the functional in Shah's variational approach to segmentation.

Chapter 4 derives two multiscale image segmentation algorithms. The algorithms are similar in that the structure of both is motivated by that of the one-dimensional segmentation algorithm derived in Chapter 3. However, the two multiscale image segmentation algorithms are also significantly different. They have slightly different structures and make use of different multiscale models. As already mentioned, the models used are simple extensions of ones developed for use in the problem of surface reconstruction [6, 8]. Numerical results are shown for the algorithms segmenting both synthetic images and satellite imagery of the Gulf Stream.

Chapter 5 summarizes the results and discusses possible avenues of further research.

Chapter 2

Background

This chapter presents some of the background material assumed in subsequent chapters. First, a brief introduction to variational calculus is presented through a discussion of the thin membrane problem in mathematical physics. It is noted that the variational formulation of the thin membrane problem can be used to low-pass filter an image. This leads to a discussion of variational approaches to problems in computer vision. In particular, Shah's variational formulation of image segmentation is presented. It is important because it underlies all of the work on segmentation appearing in later chapters. The section on variational methods leads into a discussion of the relationship between statistical and variational approaches. The thin membrane variational problem is revisited in the context of computer vision, and a statistical interpretation is presented. This motivates statistical approaches to computer vision problems. Details for a specific statistical framework for addressing computer vision problems, the multiscale modeling and estimation framework, are then discussed. Finally, there is a brief overview of previous work which successfully demonstrated the utility of the multiscale framework for addressing problems in computer vision. It is this work which has strongly motivated the investigation of this thesis into the use of multiscale statistical methods for image segmentation.

2.1 Variational Approaches to Image Segmentation

Variational calculus is a collection of mathematical tools for setting up and solving an optimization problem over a function space. The centerpiece of such a problem is a cost functional, a scalar mapping from a set of candidate functions. The functional captures the essence of a problem and, essentially, orders the candidates by the criteria implied by the form of the functional [17]. A classic example is the thin membrane problem in physics. In this case, one desires to describe the shape of the interior of a thin membrane that is pinned down on the boundary into a particular shape. One can derive a functional

$$E(f) = \int |\nabla f|^2 dxdy \tag{2.1}$$

which has the physical interpretation of representing the total potential energy of the membrane whose surface is given by the scalar function f over $\mathbf{R}^2$ [5]. Since natural phenomena tend towards states of lowest potential energy, the candidate function which minimizes (2.1) will describe the shape of the membrane very well. Notice that the integral in (2.1) incorporates the physical characteristics of the membrane and hence penalizes large gradients in the surface.

Thus, a related functional can be used to process noisy images. If one defines

$$\begin{aligned} E(f) &= \int (r^{-1}(g-f)^2 + \lambda|\nabla f|^2) dxdy \\ &= r^{-1}||g-f||^2 + \lambda||\nabla f||^2 \end{aligned} \tag{2.2}$$

where g is the raw image intensity data, r and λ are positive constants, and $||\cdot||$ denotes the standard L^2-norm of functions, then the candidate function that minimizes (2.2) is a smoothed version of the raw image. The interpretation of the terms in this functional are as follows: the first term penalizes deviations from the original image, and the second term ensures that the minimum of (2.2) is smooth. The degree of smoothing depends on the relative weighting of the data and gradient terms. The larger λ/r^{-1}, the smoother the result. Processing an image in this way is equivalent to acting on it with a low-pass filter. The result is a good method to remove noise from an image, but the drawback is that edges are blurred. One can modify the membrane problem to avoid this by introducing edge terms which prevent smoothing near the edge. Such functionals can be used to produce a segmentation.

For precisely this purpose, Mumford and Shah [18] proposed the following functional:

$$E(f,B) = \int\int_{\Omega} r^{-1}(g-f)^2 dxdy + \lambda \int\int_{\Omega - B} |\nabla f|^2 dxdy + \nu |B| \qquad (2.3)$$

where Ω is the domain of the image, $f : \Omega \rightarrow \mathbf{R}$ is a piecewise smooth approximating surface, B is the union of segment boundaries, and $|B|$ is the length of B. The edge term, B, appears in two critical places. It prevents smoothing at edges by its introduction into the domain of the second term, but to counteract this, there is also a third term, which places a penalty on the amount of edginess in the image. The constants r^{-1}, λ and ν control the degree of interaction between the terms and ultimately determine the edginess of the final segmentation. The functional (2.3) has many nice mathematical and psychovisual properties, some of which are discussed in [17]. The disadvantage of using this functional for segmentation is that actually computing minimizers is very difficult, in large part because of the discrete nature of the edge term.

Variants of this work have been proposed by Ambrosio and Tortorelli [1], [2]. They attempt to solve some of the computational difficulties associated with computing minimizers of (2.3) by constructing a family of simpler functionals whose minimum points converge to a minimum point of (2.3). One such family of functionals is at the core of an image segmentation algorithm developed by Shah [20], which has been extended and implemented by Pien and Gauch [19] among others. For this particular family of functionals, the computational difficulties associated with an edge set term are circumvented by introducing a continuous-valued edge function instead. A member of this family of functionals, parameterized by ρ, is of the form

$$E(f,s) = \int\int_{\Omega} (r^{-1}(g-f)^2 + \lambda |\nabla f|^2 (1-s)^2 + \frac{\nu}{2}(\rho |\nabla s|^2 + \frac{s^2}{\rho}))dxdy, \qquad (2.4)$$

where $f : \Omega \rightarrow \mathbf{R}$ is a piecewise smooth approximating function, and $s : \Omega \rightarrow [0,1]$ is an edge function, indicating the presence of an edge where it takes values close to one. The terms in a functional of this form have intuitive interpretations, similar to those in (2.3). The first and second terms constrain the approximating surface f to match the data as best as possible and also to be smooth in those places where s is close to zero, indicating the lack of an edge. The third term places constraints on the amount of edginess in the image and is strongly related to the penalty on the length of the boundaries appearing in (2.3). The form of the edginess penalty on s is based on work done by Modica and Mortola and is described by Ambrosio and Tortorelli [2]. The main property of (2.4) that Ambrosio and Tortorelli prove is that minimum points of (2.4) converge to a minimum point of (2.3) as $\rho \rightarrow 0$.

The general approach Shah and Pien use to minimize (2.4) is coordinate descent. A coordinate descent algorithm consists of alternating between fixing one coordinate while minimizing over the other and vice versa. An example of using this technique to find the minimum of a simple surface is diagrammed in Figure 2-1, in which asterisks mark the path of the algorithm. Notice that at each iteration, the algorithm moves to a location strictly better than the previous one. Another important characteristic of such coordinate descent algorithms is that, in most cases, they will converge to a local minimum. Now, coordinate descent of (2.4) consists of alternating between fixing s and minimizing

$$E_s = \int\int_{\Omega} (r^{-1}(g-f)^2 + \lambda |\nabla f|^2 (1-s)^2) dxdy \tag{2.5}$$

over possible f and fixing f and minimizing

$$E_f = \int\int_{\Omega} (\lambda |\nabla f|^2 (1-s)^2 + \frac{\nu}{2}(\rho |\nabla s|^2 + \frac{s^2}{\rho})) dxdy. \tag{2.6}$$

over possible s. The intuition behind these two problems is as follows. One would like to obtain an edge map by minimizing E_f over possible edge functions s, for a fixed function f. One can't estimate the edge function s directly from g because it is typically noisy and needs to be smoothed. To smooth g, one minimizes E_s

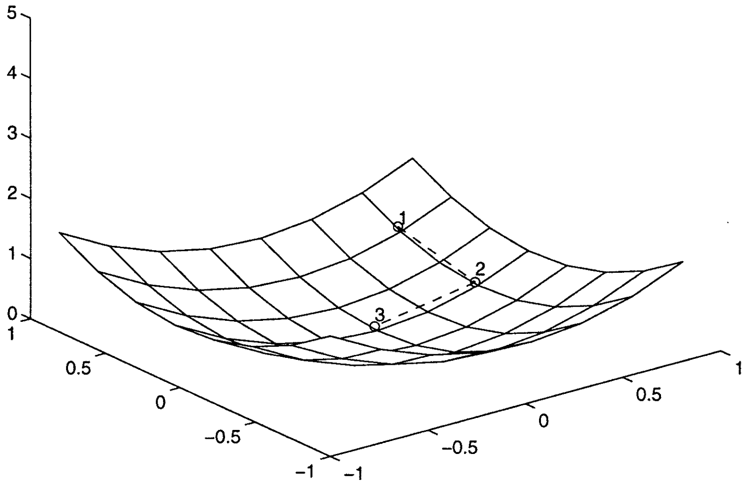


Figure 2-1: An example of coordinate descent on a simple surface.

with respect to f for a fixed current estimate of s, and then, one uses the resulting smoothed approximation f to arrive at a new estimate of s by minimizing E_f. Based on empirical evidence, Shah [20], and Pien and Gauch [19] have noted that this coordinate descent scheme converges to a reasonable solution and that the results are not significantly affected by the initial condition or whether one starts by estimating f or s. Unfortunately there are, as of yet, no mathematical results concerning the convergence of this method, but the indication is that it, at the very least, converges to a local minimum which serves as a good segmentation.

2.2 Multiscale Estimation Framework

Many functionals that are interesting for computer vision purposes are related to statistical estimation problems. For instance, consider the discrete form of (2.2),

$$E(f) = r^{-1}||g - f||^2 + \lambda||Lf||^2, \tag{2.7}$$

where f and g are elements in a finite dimensional real vector space, $||\cdot||$ represents the Euclidean norm, and L is a linear operator. Notice that this framework includes the case where f and g are vectors consisting of a lexicographic ordering of pixels in an image and L is a matrix that operates by taking first differences of nearest neighbors as an approximation of a derivative. In general, the function $\hat{f}$ that minimizes (2.7) is also the solution to finding the Bayes least-squares estimate of a process f whose measurement equation is

$$g = f + \sqrt{r}v \tag{2.8}$$

and whose prior probabilistic model is given by

$$\sqrt{\lambda}Lf = w, \tag{2.9}$$

where v and w are independent white Gaussian random vectors with identity covariance. Thus, one can view the problem at hand from the viewpoint of optimization or of statistical estimation.

The main advantage of the statistical formulation is that it casts the problem into a probabilistic framework in which it is natural to ask questions concerning the average quality of the results. This is especially relevant in many scientific applications such as remote sensing, in which one is interested in estimating the size of the errors in the results as compared to some underlying truth. Estimating the magnitude of the errors is natural in the Bayesian statistical framework. For example, if one were interested in forming a smoothed version of a noisy image, one could compute a Bayes least-squares estimate of the image using the model equations (2.8), (2.9). Then, the variance of the error for the estimate can be used to assess the quality of the smoothed image. The merits of a statistical formulation are not restricted to error statistics. Viewing a problem in a probabilistic context can also help guide one's choice of a specific operator L that leads to a prior model (2.9) which is appropriate for a specific application.

One particular class of operators are those associated with multiscale tree models [4]. Rather than describe the structure of the matrix associated with these operators, one often describes them more simply by writing down recursive equations that define the prior probabilistic model. The recursive equations for the stochastic process f are

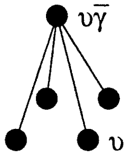


Figure 2-2: In the notation of this thesis, if ν is the index of some node, $\nu\bar{\gamma}$ denotes the parent of that node.

written in terms of a tree. Each node can have an arbitrary number of children, but for applications considered in this thesis, the number of children is constant throughout the tree and is either two or four per node. An abstract index ν is used to specify a particular node on the tree, and the notation $\nu\bar{\gamma}$ is used to refer to the parent of node ν (see Figure 2-2). The process that lives on the tree has a state variable f_ν at every node and is defined by the root-to-leaf recursion

$$f_\nu = A_\nu f_{\nu\bar{\gamma}} + B_\nu w_\nu \tag{2.10}$$

where the w_ν and the state f_0 at the root node are a collection of independent zero-mean Gaussian random variables, the w's with identity covariance and f_0 with some prior covariance. The A and B matrices are deterministic quantities which define the statistics of the process on the tree. Observations g_ν of the state variables have the form

$$g_\nu = C_\nu f_\nu + v_\nu \tag{2.11}$$

where the v_ν are an independent collection of Gaussian random variables, and the matrices C_ν are deterministic quantities which specify what is being observed.

A rich class of processes can be modeled within this framework. In particular, Luettgen demonstrates that given any one-dimensional Gauss-Markov process, there exists a multiscale one on a binary tree where the finest-scale nodes have the same statistics as the specified Gauss-Markov process [15]. The same is true for two-dimensional Gaussian Markov random fields but using quad trees instead. Such Markov processes are important because they include those processes that arise when L in (2.9) has the structure of a local difference operator, as would be natural to use when approximating a differential operator in a continuous formulation. Thus, the class of multiscale models encompass a wide variety processes that arise from using local difference operators in the plane to approximate derivatives, but the class is even larger.

Now, given a prior for a stochastic process and some data, one of the key tasks one would like to accomplish is to compute the Bayes least-squares estimate of the process. For the case in which the prior model is multiscale, Chou has derived a recursive estimation algorithm that is computationally efficient [4]. The number of multiplications required to compute the estimates and error variances is proportional to

$$\sum_{\nu \in \mathcal{T}} \delta_\nu^3 \tag{2.12}$$

where $\mathcal{T}$ is the set of nodes in the tree and δ_ν is the state dimension at node ν. In the case of N-point one-dimensional Gauss-Markov processes, the corresponding exact multiscale tree models developed by Luettgen have a fixed state dimension $\delta_\nu = 3$ for all $\nu \in \mathcal{T}$ for all $N \geq 3$. Thus, the multiscale estimation algorithm performs a number of multiplications proportional to

$$\begin{aligned} \sum_{\nu \in \mathcal{T}} \delta_\nu^3 &= 27 \sum_{\nu \in \mathcal{T}} && (2.13) \\ &= 27(N-1). && (2.14) \end{aligned}$$

So, the algorithm is essentially $O(N)$, which is fantastic.

For an $N \times N$ two-dimensional Gauss-Markov process, the corresponding multiscale tree model has a state dimension $\delta_\nu = \alpha 2^{M-m(\nu)-1} - \beta$ where $M = \log_2 N$, α and β are constants, and $m(\nu)$ is the scale corresponding to node ν with zero being the coarsest and $M-1$ being the finest. The multiscale estimation algorithm performs a number of multiplications proportional to

$$\begin{aligned} \sum_{\nu \in \mathcal{T}} \delta_\nu^3 &= \sum_{m=0}^{M-1} (\alpha 2^{M-m-1} - \beta)^3 2^{2m} && (2.15) \\ &= \frac{\alpha^3}{4}(N^3 - N^2) - \frac{\alpha^2 \beta}{4} N^2 \log_2 N + \frac{\alpha \beta^2}{2}(N^2 - N) - \frac{\beta^3}{3}(N^2 - 1) && (2.16) \end{aligned}$$

Hence, the multiscale estimation algorithm requires $O(N^3)$ multiplications to compute estimates and error variances of a two-dimensional Gauss-Markov process. As it turns out, the multiscale estimation algorithm computes estimates for such processes with optimally few multiplications in the order of magnitude sense. Recursive estimation algorithms, such as the multiscale estimation algorithm, solve a system of equations essentially by Gaussian elimination and back substitution. For the type of equations that arise when estimating an $N \times N$ Markov random field, one can not solve the system by such methods with fewer than $O(N^3)$ multiplications [10]. This is precisely the order of the number of multiplications required by the multiscale algorithm.

For problems in computer vision, one would like to do better than $O(N^3)$. Specifically, one would like to develop algorithms that are $O(N^2)$ so that they have constant computational complexity per pixel. Now, the $O(N^3)$ lower bound is applicable only when applying recursive algorithms to estimation problems involving arbitrary Gauss-Markov random field priors. In order to circumvent the lower bound, it is quite common to use iterative algorithms to compute estimates. The advantage of such techniques is that they usually calculate estimates more efficiently than recursive techniques. The disadvantage of iterative algorithms is that they generally do not compute error variances as well. Thus, one can not use iterative methods in a segmentation algorithm and expect to meet the objective of computing error statistics set out in the introduction. However, there also has been much work investigating the use of other prior models and employing recursive algorithms to compute estimates and error variances efficiently for problems in computer vision.

Recall that the Gauss-Markov random field model arose as a result of using simple difference schemes to approximate the derivatives in the continuous variational formulations. However there is no particular reason why this is the right thing to do. One may be able to find other models which lead to less computationally intensive recursive estimation algorithms and have characteristics appropriate for a computer vision problem. In particular, there recently has been much work on the use of multiscale prior models with bounded state dimension ($\delta_\nu \leq \delta \ \forall \nu, \forall N$, for some fixed δ). From (2.12), one notes that in this case the multiscale estimation algorithm can

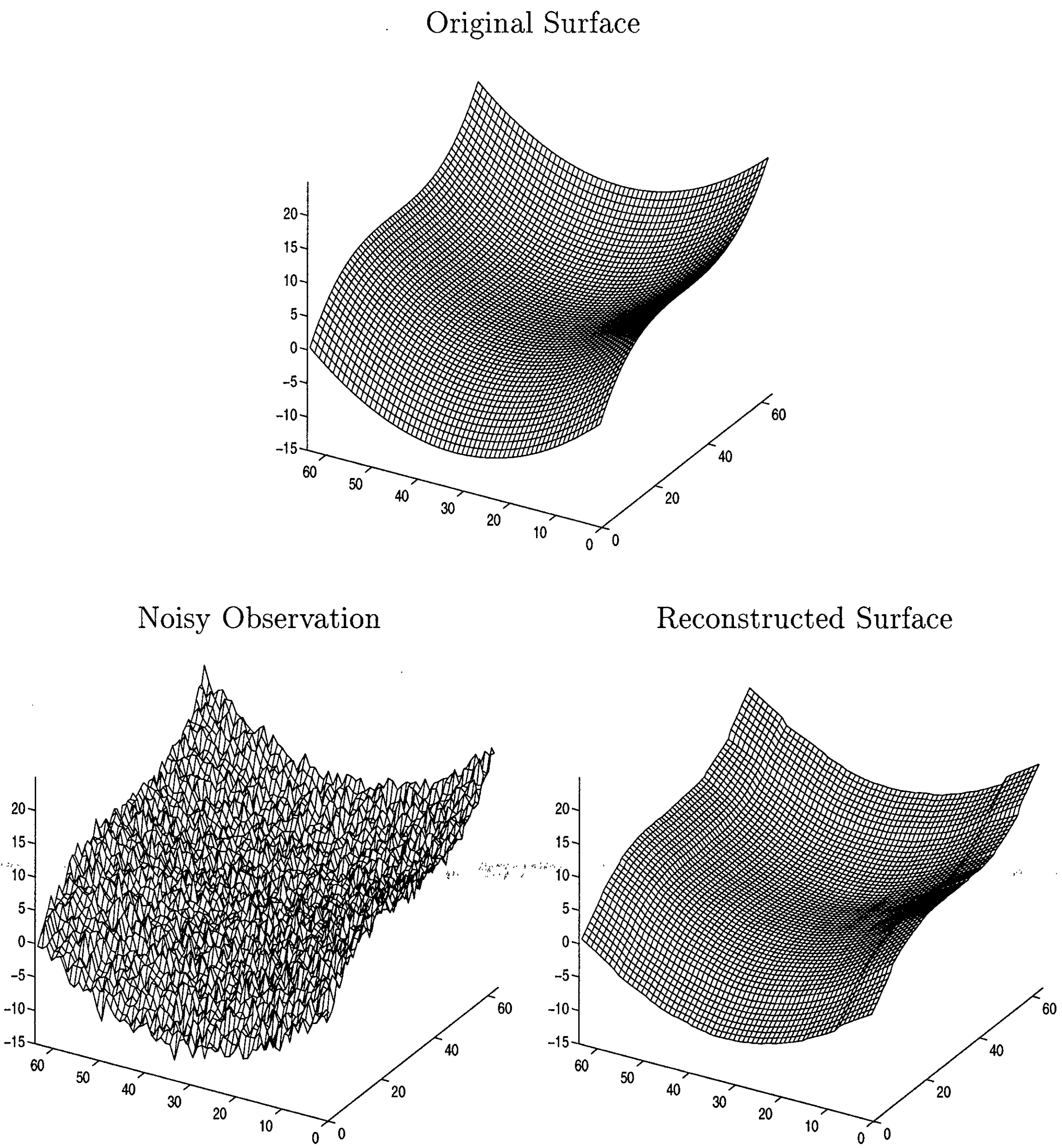


Figure 2-3: An example of using Paul's multiscale models to perform surface reconstruction.

calculate the estimates and error statistics with $O(N^2)$ multiplications. The question that remains is whether one can find such models which are appropriate for computer vision problems.

Luettgen and Fieguth have addressed this issue and concluded that there are multiscale priors which can be used to obtain results similar to the ones obtained by minimizing functionals associated with the optical flow [16] and surface reconstruction problems [6]. As an example of this approach, consider the problem of reconstructing a surface from noisy measurements. The functional typically associated with this problem is

$$E(f) = \int \Big((g - f)^2 + \alpha(p_x^2 + p_y^2 + q_x^2 + q_y^2) + \beta(|\nabla f|^2)\Big) \tag{2.17}$$

where g is a data term, $p = \partial f/\partial x$, $q = \partial f/\partial y$, and the subscripts denote second partial derivatives with respect to the indicated variable. Notice that the difference between (2.17) and (2.2) is the introduction of second derivatives. The second derivative term represents the potential energy of a deformed thin plate, as opposed to a thin membrane. Fieguth developed a multiscale prior with constant state dimension which can be used in place of a Markov random field involving local difference operators that mimic first and second derivatives. An impressive result using this model to reconstruct a smooth surface from measurements which contain unit intensity white Gaussian noise is illustrated in Figure 2-3.

2.3 Concluding Remarks

The background material presented in this chapter motivates the flow of the remainder of the thesis. The next chapter starts with Shah's variational approach to image segmentation and develops a statistical interpretation of it. This interpretation is then used to guide the development in Chapter 4 of an image segmentation algorithm which calculates efficiently both edge and piecewise smooth image estimates and also the error variances for those estimates. The algorithm is formulated in a multiscale framework according to the precedent set by Luettgen and Fieguth. Based on their promising results, one expects the final image segmentation algorithm to perform well, and this is what one observes.

Chapter 3

Segmentation in One Dimension

The problem of deriving a multiscale algorithm to segment images is a difficult one and needs to be simplified as a first step. A natural approach is to reduce the complexity of the problem by reducing the dimension of the domain involved. Rather than segmenting images, one could try segmenting one-dimensional signals. This problem is considerably easier for many reasons. First of all, one desires that the edge sets of images are connected in some way, but the edge sets of signals do not need to satisfy this restriction. Thus, segmenting a signal is a fundamentally easier problem. In addition, computation is not as much of an issue. Many algorithms including multiscale ones can do the relevant computation and have a complexity that is proportional only to the number of points in the signal. Even though this one-dimensional segmentation problem is much easier than the image segmentation problem, there is much that can be learned by considering this simplification, both from deriving the algorithm and analyzing its results and properties.

3.1 The Estimation Problem

In order to use multiscale statistical methods to perform segmentation based on the variational formulation of Shah introduced in the last chapter, one must first derive related estimation problems. Since these problems are formulated in a continuous space, one should first analyze the estimation problems for functions defined on the real line. This will give one an idea for the form of the statistical formulation and the intuition behind it. In order to implement the algorithm using standard multiscale techniques, one needs a problem for which one estimates sampled functions, however. The discretization of the signal and edge functions must be done carefully for a variety of reasons. The discussion of both the continuous and discrete problems in one-dimension yields considerable insight into the original segmentation problem and forms the basis for a one dimensional segmentation algorithm.

3.1.1 Continuous version

In one-dimension, equation (2.5) becomes

$$E_s(f) = \int_{\Omega} (r^{-1}(f-g)^2 + \lambda|\frac{df(x)}{dx}|^2(1-s)^2 dx. \tag{3.1}$$

The problem of minimizing this functional with respect to f, for a fixed function s that is less than one, is equivalent to the problem of estimating f given a prior model described by

$$\begin{aligned} g(x) &= f(x) + \sqrt{r}v^f(x) && (3.2)\\ \frac{df(x)}{dx} &= \frac{1}{(1-s(x))\sqrt{\lambda}}w^f(x) && (3.3) \end{aligned}$$

where $v^f(x)$ and $w^f(x)$ are independent Gaussian white-noise processes with unit intensity. Equation (3.3) admits a very nice intuitive explanation. Where the edge function $s \approx 1$, the multiplier of the process noise $1/(1-s)$ is very large. Thus, at these locations, the prior model for the underlying function f allows for increased variability, and a least-squares estimator will allow large jumps to occur in the estimate of the function f. This is exactly what one wants the estimator to do at edge locations.

Casting the minimization over the edge function s for a fixed piecewise smooth function f into a statistical framework is more difficult because of the constraint placed on the edge process that it lie between zero and one. If one removes this constraint, the new unconstrained problem is easier and not much different from the constrained one. Minimizing (2.6) is equivalent to minimizing

$$E_f(s) = \int_\Omega (\lambda|\frac{df}{dx}|^2(1-s)^2 + \frac{\nu}{2}(\rho|\frac{ds}{dx}|^2 + \frac{s^2}{\rho}))dx \tag{3.4}$$

with respect to s. By completing the square, one can rewrite the integrand as

$$(a+b)(\gamma - \gamma^2 + (s-\gamma)^2) + c|\frac{ds(x)}{dx}|^2 \tag{3.5}$$

where $a(x) = \lambda|df/dx|^2$, $c = \nu\rho/2$, $b = \nu/2\rho$, and $\gamma(x) = a(x)/(a(x)+b)$. Thus, the problem of finding the minimum over s of (3.4) is the same as finding the minimum of

$$E_f(s) = \int_\Omega (a+b)(s-\gamma)^2 + c|\frac{ds(x)}{dx}|^2 dx. \tag{3.6}$$

This leads one to an estimation theoretic problem described by the pair of equations

$$\begin{aligned} \gamma(x) &= s(x) + \frac{1}{\sqrt{a(x)+b}} v^s(x) && (3.7) \\ \frac{ds(x)}{dx} &= \frac{1}{\sqrt{c}} w^s(x) && (3.8) \end{aligned}$$

where $w^s(x)$ and $v^s(x)$ are independent Gaussian white-noise processes with unit intensity. Notice that γ plays the role of an observation of the edge function. This function takes on values close to one where the derivative of the smooth function f is large and zero where the derivative is small. Since one declares edges at locations where the edge function $s \approx 1$, the form of γ makes intuitive sense. Observe also that the range of γ lies within $[0,1)$. As a consequence, the first term in (3.6) provides an increased penalty for functions s that do not stay within $[0,1]$. This is desirable because a solution to the unconstrained minimization of (3.6) that lies within $[0,1]$ is an optimal solution of the constrained problem. As it turns out, this is often the case, as discussed in Section 3.2, which presents results for the segmentation algorithm that is formed by alternately solving discrete versions of the estimation problems specified by (3.2), (3.3) and (3.7), (3.8).

3.1.2 Discrete version

The first attempt at discretizing (3.2), (3.3) and (3.7), (3.8) was to discretize each estimation problem separately without much regard as to how they interacted. This approach did not yield the desired results. To understand why, recall that provided these estimation problems yield estimates that satisfy the constraint $s \in [0,1]$, they are equivalent to the variational minimization problems that result from using coordinate descent to minimize the functional (2.4). Thus, the process of alternately finding estimates of the piecewise smooth function f and the edge function s should converge to a local minimum. The discretized estimation problems that were formed, however, did not. The cause of this problem was the fact that the pair of discrete estimation problems were not equivalent to estimation problems associated with the coordinate descent of a discrete functional. In order to ensure convergence, one merely has to be very careful about how one defines the sampling grids and the difference operators used to approximate derivatives on those grids.

The natural approach is to define the samples to be regularly spaced in a closed interval on the real line and to use a first order difference operator to replace the derivative. This can be written most simply by using the notation of real vector spaces. For example, a collection of n regularly spaced samples of the function $f(x)$ is written as a vector $f \in \mathbf{R}^n$. By defining the $(n-1) \times n$ matrix

$$L = \begin{pmatrix} -1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & 0 & \cdots & 0 & 0 \\ & & & & \ddots & & \\ 0 & 0 & 0 & 0 & \cdots & -1 & 1 \end{pmatrix}. \tag{3.9}$$

the first difference of $f \in \mathbf{R}^n$ can be written as Lf. Notice that if $f \in \mathbf{R}^n$, then $Lf \in \mathbf{R}^{n-1}$. Since the edges are defined in terms of this difference operator, one can not use the same number of sampling points for the piecewise smooth approximating function f and the edge function s. The solution is to mesh the sampling grids for these two functions: samples of the edge function occur only between samples of the

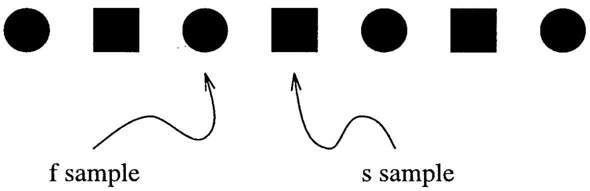


Figure 3-1: An example of the sampling grids that would be used for the piecewise smooth function f and edge function s.

piecewise smooth approximating function. This is diagrammed in Figure 3-1. This adds a little bit of complexity, but it is necessary in order to maintain consistency.

Keeping this sampling framework in mind, one can rewrite the functional (2.4) in discretized form as

$$E(f,s) = r^{-1}\sum_{i=1}^{n}(f_i - g_i)^2 + \lambda\sum_{i=1}^{n-1}(1-s_i)^2(f_{i+1}-f_i)^2 + \frac{\nu}{2}(\rho\sum_{i=1}^{n-2}(s_{i+1}-s_i)^2 + \frac{1}{\rho}\sum_{i=1}^{n-1}s_i^2). \tag{3.10}$$

Now, the problem of fixing s and finding the f that minimizes (3.10) is equivalent to finding the f that minimizes the discrete functional

$$E_s(f) = r^{-1}\sum_{i=1}^{n}(f_i - g_i)^2 + \sum_{i=1}^{n-1}\lambda(1-s_i)^2(f_{i+1}-f_i)^2. \tag{3.11}$$

A slightly more compact form can be written by using the notation $||x||^2 = x^T x$ and $||x||_W^2 = x^T W x$ for vectors $x \in \mathbf{R}^n$ and matrices $W \in \mathbf{R}^{n\times n}$. Now, for an edge process $s \in \mathbf{R}^{n-1}$, define the diagonal matrix

$$S = \begin{pmatrix} (1-s_1) & & \\ & \ddots & \\ & & (1-s_{n-1}) \end{pmatrix}. \tag{3.12}$$

Then, (3.11) simplifies to

$$E_s(f) = ||f-g||_{r^{-1}I}^2 + \lambda||Lf||_{S^TS}^2. \tag{3.13}$$

Finding the minimum of E_s for fixed invertible S is equivalent to finding the least-squares estimate of f assuming the following measurement and prior model:

$$\begin{aligned} g &= f + \sqrt{r} v^f & (3.14) \\ Lf &= \frac{1}{\sqrt{\lambda}} S^{-1} w^f & (3.15) \end{aligned}$$

where v^f and w^f are independent Gaussian random variables with covariance I.

Likewise, the problem of finding the s that minimizes (3.10) for fixed f is the same as finding the s that minimizes the discrete functional that can be written as

$$E_f(s) = \lambda||Lf||^2_{S^TS} + \frac{\nu}{2}(\rho||Ls||^2 + \frac{1}{\rho}||s||^2) \qquad (3.16)$$

with only a slight abuse of notation that occurs since the matrix L has a different dimension in each of the two terms in which it appears in (3.16) (since s is of dimension one less then f). As in the continuous case, one can arrive at a standard estimation problem if one removes the constraints placed on the edge function s. Now, make the substitutions $c = \nu\rho/2$, $b = \nu/2\rho$, and define the diagonal matrix

$$A = \begin{pmatrix} \sqrt{\lambda(Lf)_1 + b} & & \\ & \ddots & \\ & & \sqrt{\lambda(Lf)_{(n-1)} + b} \end{pmatrix}, \qquad (3.17)$$

and the vector

$$\gamma = \begin{pmatrix} \frac{\lambda(Lf)_1^2}{\lambda(Lf)_1^2 + b} \\ \vdots \\ \frac{\lambda(Lf)_{(n-1)}^2}{\lambda(Lf)_{(n-1)}^2 + b} \end{pmatrix}. \qquad (3.18)$$

This leads to the the problem of estimating s given the following measurement and prior model:

$$\begin{aligned} \gamma &= s + A^{-1} v^s & (3.19) \\ Ls &= \frac{1}{\sqrt{c}} w^s & (3.20) \end{aligned}$$

where v^s and w^s are again independent Gaussian random variables with covariance I. Combining this estimation problem with that described by (3.14), (3.15) yields an algorithm that can segment well.

3.2 Results

Almost all of the pieces of the segmentation algorithm are in place, and it remains only to put them together and discuss some of the implementation details. Using the estimation problem formulations (3.14), (3.15) and (3.19), (3.20), one can directly apply the results of [15] to obtain nearly equivalent multiscale recursive models. The only difference in the models is that the standard multiscale recursive form requires the specification of prior covariances P_0^f and P_0^s on the first samples of the piecewise smooth function f and edge function s. However, the precise interpretation of the variational formulation as an estimation problem corresponds to viewing the initial value as unknown, which is equivalent to a maximum-likelihood problem. One can closely approximate the solution to this problem in the standard multiscale framework by setting the prior covariances P_0^f and P_0^s to large numbers. Given the resulting multiscale model, one can use a recursive estimation algorithm [4] to estimate a signal of length n with $O(n)$ multiplications. Using this multiscale estimation engine, the segmentation algorithm involves alternately estimating the edge function s and the piecewise smooth approximating function f. Enforcing the constraint on the range of s has been ignored up till now, but one must incorporate the constraint in the algorithm because estimating f requires that $\frac{1}{(1-s)^2}$ be well-behaved. A simple solution that proves adequate is to clip each estimate of the edge function so that for some small ϵ, $s \in [0, 1-\epsilon]$. With the introduction of the clipping step, one iteration of the segmentation algorithm, as diagrammed in Figure 3-2, is determined. The only remaining two items to specify are how to start and when to stop. For all of the examples in this section, the algorithm starts by estimating s using the data as an initial estimate of the smooth function f, and the algorithm stops when the percent change of the functional (3.10) falls below some threshold Δ. This threshold is the

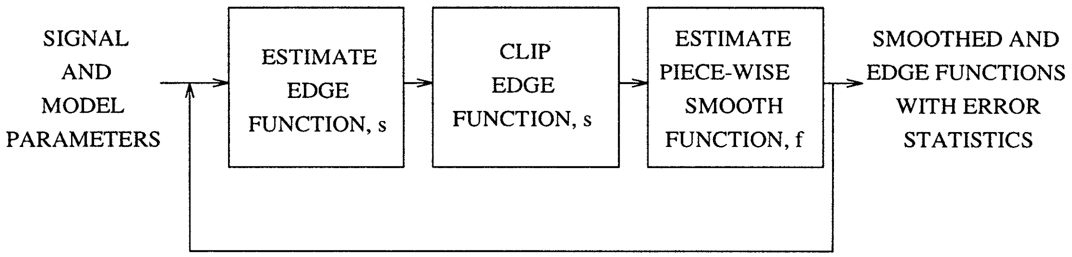

SIGNAL AND MODEL PARAMETERS
ESTIMATE EDGE FUNCTION, s
CLIP EDGE FUNCTION, s
ESTIMATE PIECE-WISE SMOOTH FUNCTION, f
SMOOTHED AND EDGE FUNCTIONS WITH ERROR STATISTICS

Figure 3-2: The one dimensional segmentation algorithm.

last of the four new parameters P_0^f, P_0^s, ϵ, and Δ that have been added to the list of parameters λ, ν, ρ and r, which are input to the one dimensional segmentation algorithm. To illustrate the operation of the algorithm, some typical examples follow. These, in turn, are followed by some Monte Carlo experiments designed to assess quantitatively the performance of the algorithm.

3.2.1 Typical Examples

Figure 3-3 illustrates a segmentation for a synthetic example, using the parameters in Table 3.1. The data consists of a synthetic signal to which white Gaussian noise with unit intensity has been added. The synthetic signal is a a realization of a Gaussian process described by (3.14), (3.15) for a small initial covariance of 0.001 and for an exponential edge function s_0 that is portrayed in the figure. The particular function s_0 used is natural in the following sense. If f were fixed as a step function with a single large jump, then the estimates resulting from the estimation problem posed in (3.19), (3.20) would be precisely s_0. Now, recall that where the edge function is approximately one, the variance of the process noise for the model of f increases. Thus, a realization is more likely to have jumps at such locations, but not all realizations will have jumps where the edge function is close to one. The particular realization used in the example displayed in Figure 3-3 was chosen for having a noticeable jump in the vicinity of the edge function's peak. The results shown are for four iterations, at which point, the values of the functional (3.10) were changing by less than $\Delta = 1\%$. No clipping was necessary during the course of the run, and thus, the results are true to Shah's variational formulation. The final estimates yield a good segmentation.

Parameter	*Value*
λ	1
b	10
c	100
r	1
P_0^f	100
P_0^s	100
ϵ	1.0×10^{-4}
Δ	1%

Table 3.1: Parameter values for the synthetic example in Figure 3-3.

The piecewise smooth function is a smoother version of the data, but the edge has not been smoothed away, and the edge function has a strong peak at the location of the edge. The results for the synthetic signal are good, but the signal used in this example is matched to the algorithm by its construction.

A simpler more prototypical example is a step edge. Figure 3-4 displays results for a noisy observation of a unit step, using the parameters in Table 3.2. The estimates are shown after six iterations, at which point the percentage change in the functional (3.10) has dropped below $\Delta = 1\%$. Once again, no clipping was necessary in the iterative process. The final results are very impressive. The piecewise smooth function is almost flat everywhere except at the location of the original step, and the edge function marks the location of that step well. When considering the effect of noise on the algorithm, the most useful quantity to consider is the ratio of standard deviation of the observation noise to magnitude of the discontinuity. For this unit step example, the ratio is a moderately large 0.2. How the algorithm performs with respect to other noise values is explored in Section 3.2.3. However, just from this one example, it appears that the performance of the algorithm is quite good.

The iterative scheme that generates these results converges quite nicely. Figure 3-5 displays values of the functional after each iteration for the examples already presented. One can observe not only the monotone behavior one expects from a coordinate descent minimization but also a rapid convergence to a local minimum. A few test runs have indicated that only three iterations of the algorithm are necessary to get reasonably close to a minimum and that the local minimum to which one converges

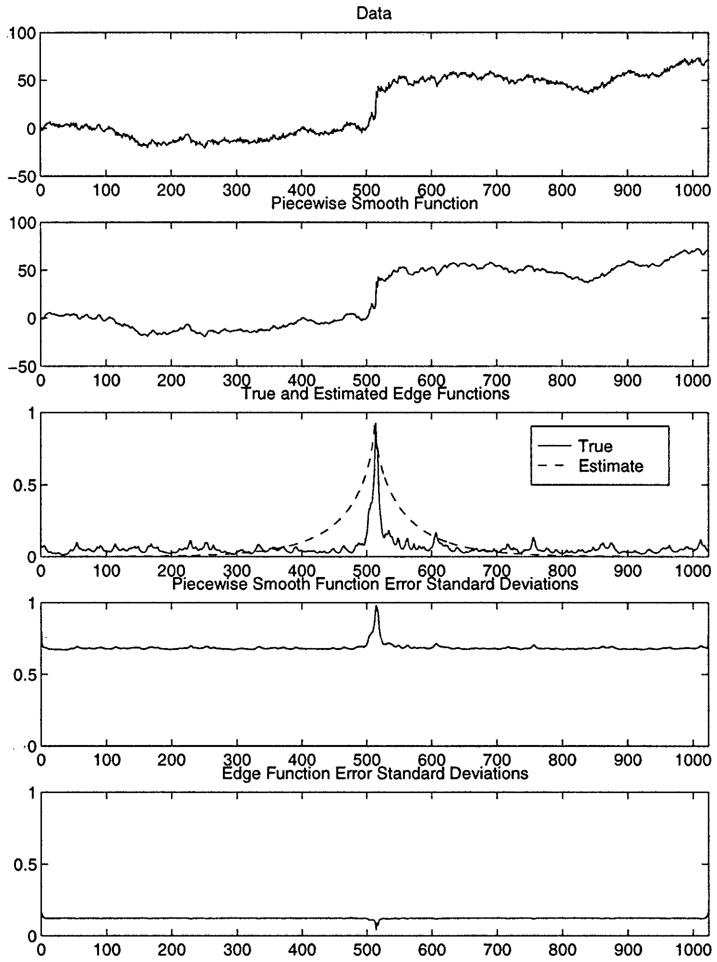


Figure 3-3: A synthetic segmentation example.

is independent of initial location in the piecewise smooth–edge function coordinate space. Shah remarks on the apparent uniqueness of the result [20], but nobody has formally demonstrated that this coordinate scheme has a unique stationary point. One can conclude from the empirical evidence, however, that for many functions of interest, convergence is fast and robust to perturbations in the initial conditions.

This fast segmentation algorithm not only computes the impressive estimates already discussed but also error standard deviations. They are graphed along with the estimates in Figures 3-3 and 3-4. Recall first of all that the plots are not truly of error standard deviations but of error standard deviations conditioned on assuming some extra knowledge. In the case of the piecewise smooth function, one assumes knowledge of the edges. Thus, the error standard deviations are higher in the locations

Parameter	*Value*
λ	2.5×10^3
b	25
c	25
r	0.04
P_0^f	100
P_0^s	100
ϵ	1.0×10^{-4}
Δ	1%

Table 3.2: Parameter values for the step edge example in Figure 3-4.

of the edges, where the process noise variance in the statistical model (3.14), (3.15) is higher. This is very intuitive because the behavior of the function at edges is more difficult to estimate from noisy data. One can also form an intuitive understanding of the edge function error standard deviations. These are computed assuming knowledge of the piecewise smooth function. Where it has an abrupt change, one can be fairly certain that there is an edge present in the original data. Thus, the error variance for estimating the edge function decreases at these points. At locations where the piecewise smooth function is smooth, the presence or absence of an edge in the original data is indeterminate because the process of estimating the smooth function may have smoothed over discontinuities. Hence, the error variance will be larger at these locations than where there is an edge. The amount of uncertainty in these locations is determined by how smooth one expects the original function to be, as specified by the parameter λ. Thus, the segmentation algorithm generates error standard deviations that match one's intuition and are useful in interpreting the final estimates.

3.2.2 Error Statistics

By performing a variety of Monte Carlo experiments, one can obtain a quantitative assessment of the algorithm that deepens one's understanding of it beyond the intuitive level of the last section. Figure 3-7 gives results for some Monte Carlo simulations that characterize the errors in the estimates generated by the segmentation algorithm and how they relate to the error variances calculated by the algorithm itself. For each run of the experiment, one generates a realization f of the process de-

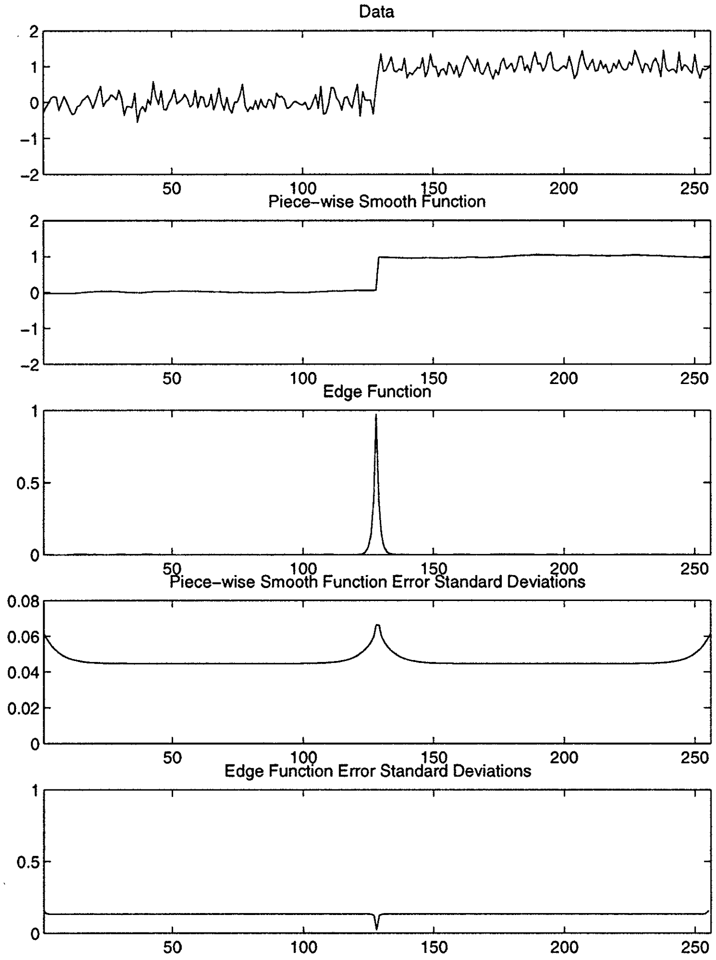


Figure 3-4: A step edge segmentation example.

scribed by (3.14), (3.15) for a small initial covariance of 0.01 and the edge function s_0, which is the same as in Figure 3-3 but is also graphed in Figure 3-6. After generating the realization f using the edge function s_0, one adds white Gaussian measurement noise with intensity $P_v = 1$ and then applies the segmentation algorithm using the parameters in Table 3.1 to obtain the estimates $\hat{f}$ of the realization f and $\hat{s}$ of the edge function s_0 as well as P_f and P_s, the error variances for these estimates that the algorithm generates. The main quantities of interest for each run are $e_f = (\hat{f} - f)$, $e_s = (\hat{s} - s_0)$, and the error variances P_f and P_s. From many independent runs, the following quantities are estimated: Ee_f^2, $Var(e_f)$, EP_f, Ee_s^2, $Var(e_s)$, and EP_s. As a reference, one can compare the error statistics associated with estimating the piece-wise smooth function to $\tilde{P}_f$, the optimal error variance for generating the estimate

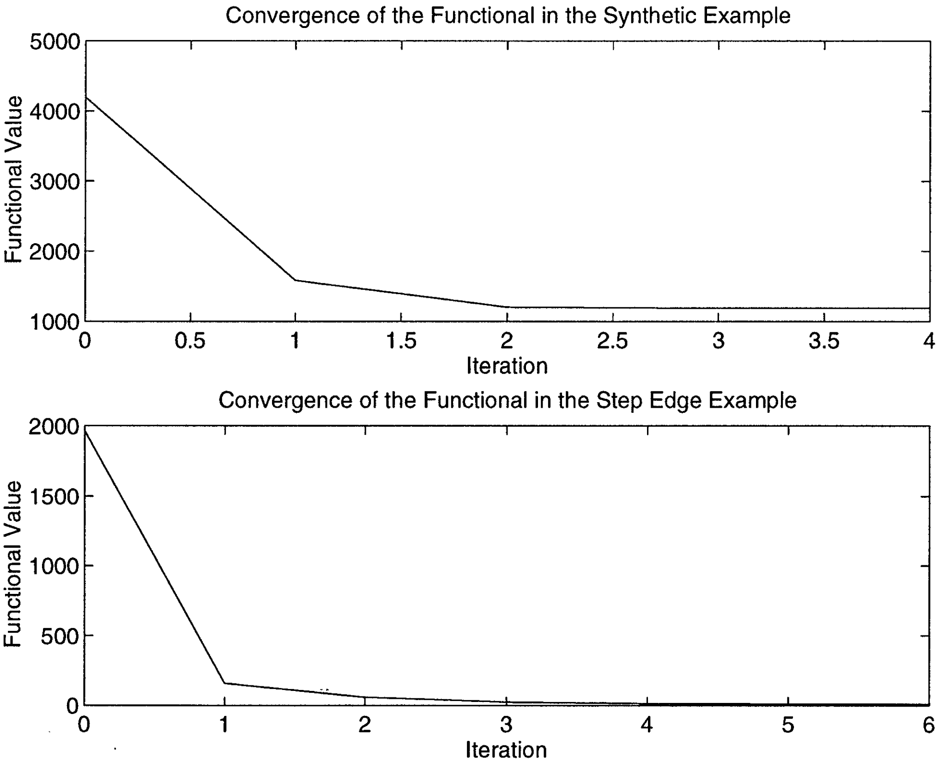


Figure 3-5: These plots depict how the value of the functional converges for the examples of Figures 3-3 and 3-4.

when the true edge function s_0 is known. The results from doing this for 100 runs are plotted in Figure 3-7, in which error bars at set at two standard deviations.

Figure 3-7a depicts the quantities relating to the errors for estimating the piecewise smooth function f. Notice that $Ee_f^2 \approx Var(e_f)$, which is close to $\tilde{P}_f$. This implies that the average error in the estimate of the piecewise smooth function f generated by the segmentation algorithm (which does not have prior knowledge of s_0 but must estimate it) is close to the optimal average error for estimating f when the underlying edge function s_0 is known. Thus, the estimate generated by the algorithm is quite good. In addition, $EP_f \approx \tilde{P}_f$. Hence, the error variance generated by the algorithm is close to the true error variance conditioned on knowing the edge function. So, the algorithm generates a good estimate of the piecewise smooth function, and the error in that estimate is fairly close to the error variance generated by the algorithm.

The error statistics for the edge function are depicted in Figure 3-7b. Both Ee_s^2 and $Var(e_s)$ are small relative to one. The first indicates that the estimate of the edge

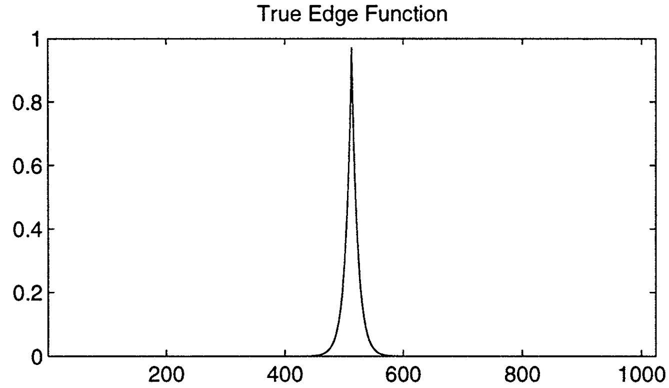


Figure 3-6: The edge function used to generate the realizations in the Monte Carlo runs of Figures 3-7 and 3-12 through 3-14.

function is fairly accurate, and the second indicates that the error does not vary very much from sample path to sample path. In addition to the magnitude, the shape of the plot of Ee_s^2 has two interesting features. The first is that the mean-squared error jumps close to the edge location. If one examines Figure 3-3, one can see that the algorithm generates edge functions that are more narrow than s_0, the edge function used to generate the realizations. Apparently, the algorithm tends to make the peak of the estimated edge function more narrow than that specified by the function used to generate realizations. This is actually preferable for segmentation. The other interesting point is that the shapes of $Var(e_s)$ and EP_s are not the same. $Var(e_s)$ increases where EP_s decreases. This is due to the fact that the peak of the estimate of the edge function appears in slightly different locations in each segmentation, thereby introducing a large variance near the edge. This does not necessarily imply a poor segmentation, however. The conclusion is that e_s is not the best measure of the error between edge functions. Some quantity that captures difference in locations between peaks would be more useful. Nevertheless, the statistics of e_s are important in assessing the error in the estimate of the edge function. In this particular example, the statistics of e_s indicate that the estimates of the edge function are close to s_0, and the error variance generated by the algorithm, P_s, is useful because it is of the same order of magnitude as $Var(e_s)$.

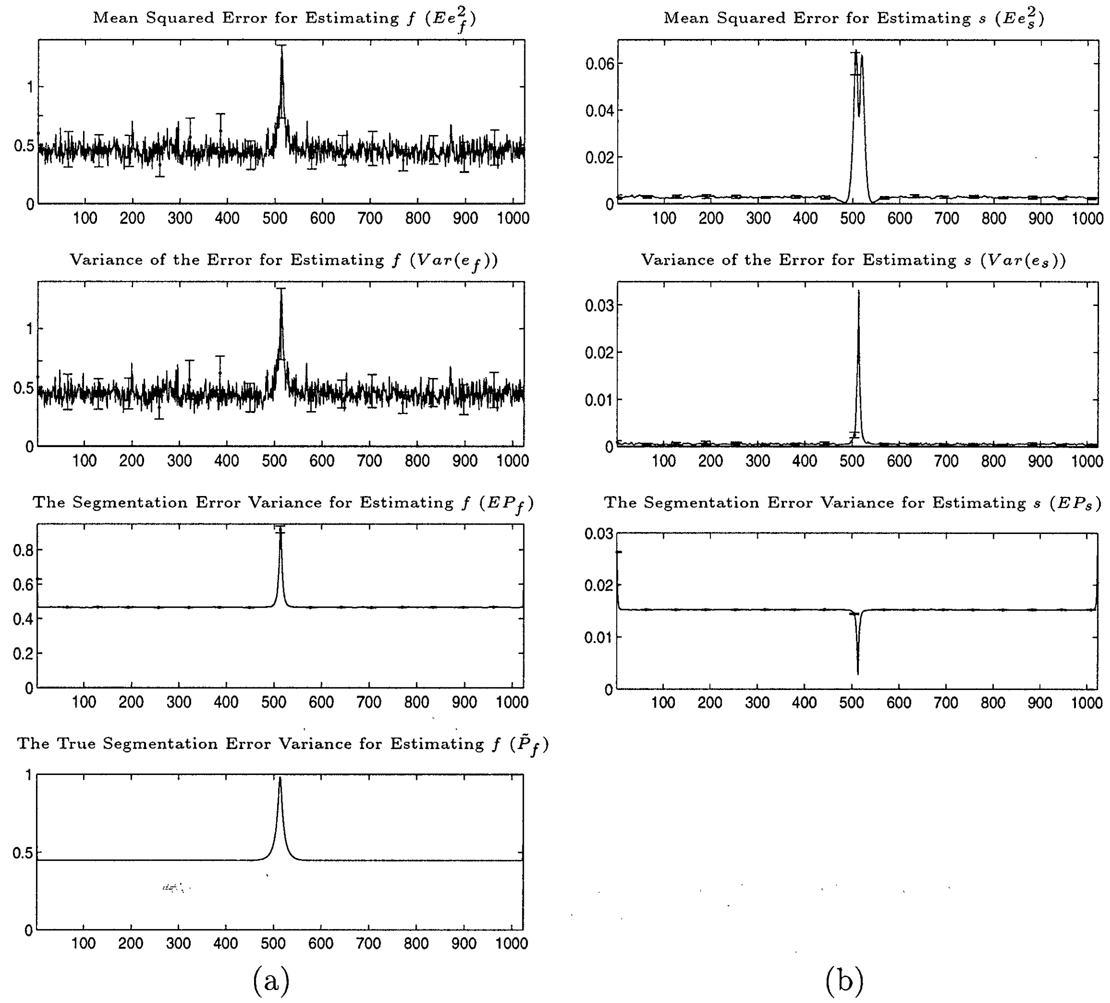


Figure 3-7: A comparison of various error statistics compiled using Monte Carlo techniques for segmenting synthetic data.

3.2.3 Edge Localization

Figures 3-9 to 3-11 present results from a series of Monte Carlo simulations designed to characterize how well the algorithm can segment signals for different sets of measurement noise and parameter choices. For each of the three experiments, the algorithm is segmenting a unit step edge. The quantity computed from each segmentation is W, the number of values of the estimated edge function that lie above a given threshold set close to one. This corresponds to the sum of the widths of the edges in the segmentation. For the step edge example, the desired value of W is one. In all of the figures, the parameters not being adjusted take on the values listed in Table 3.2, and the threshold is set at 0.9.

The first simulation characterizes how W is affected by measurement noise. Figure 3-9 presents the average of W versus the standard deviation of the measurement noise. For this example, the value of the parameter r is set equal to P_v, the variance of the measurement noise. The results are shown for 100 runs, and the error bars are set at three standard deviations. The results indicate that the algorithm performs very well up to a noise standard deviation of about 0.3. After this point, the algorithm still yields reasonable segmentations, especially considering that the signal length is 256 and that the noise level is so high that localizing the edge to one or two pixels is not possible. The difficulty is illustrated in Figure 3-8, which shows a unit step edge added to white Gaussian measurement noise for three different standard deviations: 0, 0.2, and 0.4. Localizing the edge and avoiding detection of spurious edges to noise are clearly more difficult as the noise increases. The conclusion is that W will increase with added noise, but the amount of increase is to be expected and is rather small considering the levels of noise involved.

In Figure 3-10, the results for varying the parameter λ are plotted while maintaining a value of $\sqrt{r}$ and measurement noise standard deviation $\sqrt{P_v}$ of 0.2. The results shown are for 500 runs, and the error bars are set at three standard deviations. Recall that the amount of smoothness the algorithm expects in f where there is no edge is directly related to λ. For $\lambda = 2.5 \times 10^3$, the algorithm generates the very flat step estimate of Figure 3-4. However, one needs to be careful setting λ, as Figure 3-10 shows, because the average of W will increase with λ. Remember that W is a measure of how many points are declared edges. If λ is set too high, the algorithm will declare edges in many places and set the estimate of the function f almost constant between edges. Yet, the slope of the curve in Figure 3-10 is not very steep, indicating that small perturbations in the value of λ will not severely diminish the performance of the algorithm.

Finally, Figure 3-11 plots the effect of changing the parameter ρ while fixing $\lambda = 2.5 \times 10^3$ and $r^{-1} = P_v = 0.04$. The results shown are for 500 runs, and the error bars are set at three standard deviations. Although the value of ρ is not listed

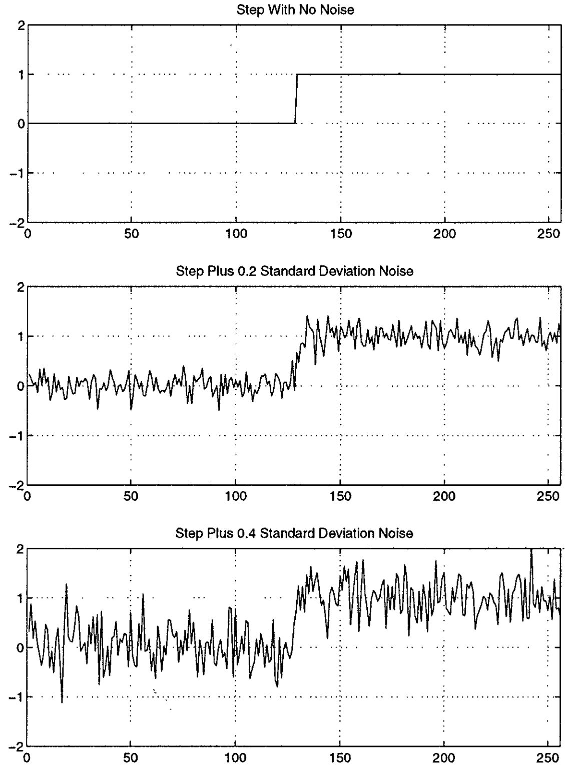


Figure 3-8: Step edges with differing amounts of measurement noise.

in Table 3.2, ρ is uniquely determined by b and c. For the b and c listed in Table 3.2, $\rho = 1$, which should be considered the nominal value of ρ for this example. Now, recall that the parameters b and c in the statistical formulation are related to ρ by: $c \propto \rho$ and $b \propto 1/\rho$. Also remember that the solutions of Shah's functional (2.4) converge to that of Mumford and Shah's (2.3) as $\rho \to 0$. Thus, one expects the width of the edges to decrease as $\rho \to 0$. and this is corroborated by the results in Figure 3-11. As in the case of adjusting λ, the slope of the curve in Figure 3-11 is not very steep, indicating that the effect of small changes in ρ will have a minimal effect on the results of the segmentation algorithm.

3.2.4 The Noise Parameter

In each of the previous experiments, the value of r is set equal to P_v. The next and last set of Monte Carlo simulations characterize the effects of adjusting the variance of the measurement noise P_v in the simulation separately from the measurement

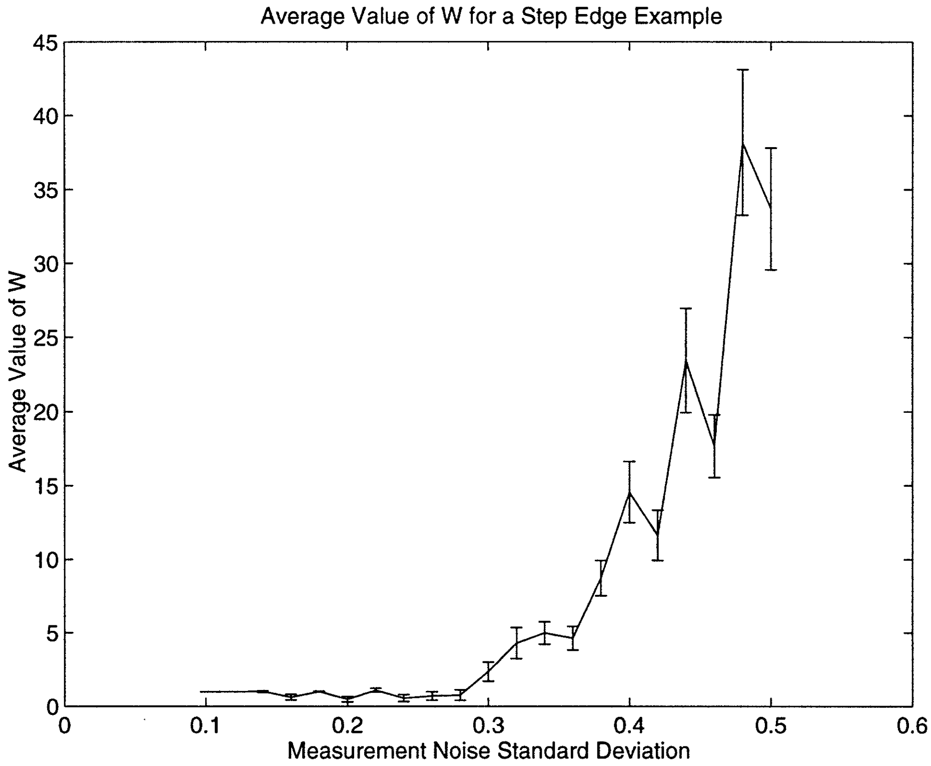


Figure 3-9: The average value of W for the step edge example of Figure 3-4 but with different levels of measurement noise added.

Figure	P_v	r
3-7	1	1
3-12	5	1
3-13	5	5
3-14	1	5

Table 3.3: Values of P_v and r used to characterize the error statistics.

noise parameter r in the algorithm. The first set of simulations presented here in Figures 3-12 to 3-14 address how adjusting r and P_v affects the error statistics. The experiment used is precisely the same as that used in Figure 3-7. The values of P_v and r considered are listed in Table 3.3.

The results displayed in Figure 3-12 are for $P_v = 5$ and $r = 1$. They are computed from 100 runs, and error bars are set at two standard deviations. In Figure 3-12a, $\tilde{P}_f$ is the optimal error variance for estimating the piecewise smooth function f given knowledge of the true edge function and assuming the measurement noise variance is 5. One observes that $Var(e_f)$ is significantly higher than $\tilde{P}_f$ away from the edge, but

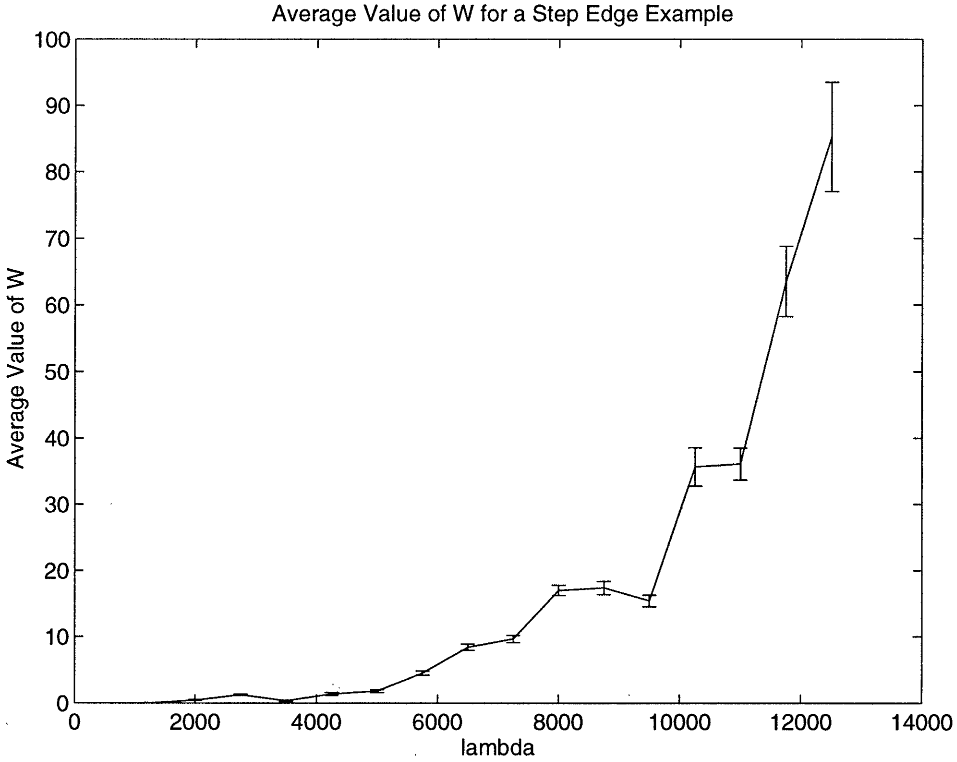


Figure 3-10: Average value of W for the step edge example of Figure 3-4 but for different values of the parameter λ.

$Var(e_f) \approx \tilde{P}_f$ in the vicinity of the edge. A similar examination of Ee_s^2 in Figure 3-12b indicates that the algorithm is able to fairly accurately estimate the edge function for this case. The conclusion is that maintaining $r = 1$ despite moderate increases in noise leads to segmentations for which the edge and the piecewise smooth function estimates are fairly good, especially in the vicinity of the actual edge, but the estimate of the piecewise smooth function f is poor away from the edge in comparison with $\tilde{P}_f$. This is because the algorithm thinks the measurements are very good; so, it is responsive to the data, localizing edges well but not smoothing the data away from the edge.

This is in contrast with the results in Figures 3-13 and 3-14, for which $r = 5$. One hundred runs were used to generate each of these plots, and the error bars are set at two standard deviations. In both Figures 3-13a and 3-14a, $\tilde{P}_f$ is the optimal error variance for computing f given the true edge function and assuming the actual amount of measurement noise P_v, which is 5 in Figure 3-13 and 1 in Figure 3-14. For

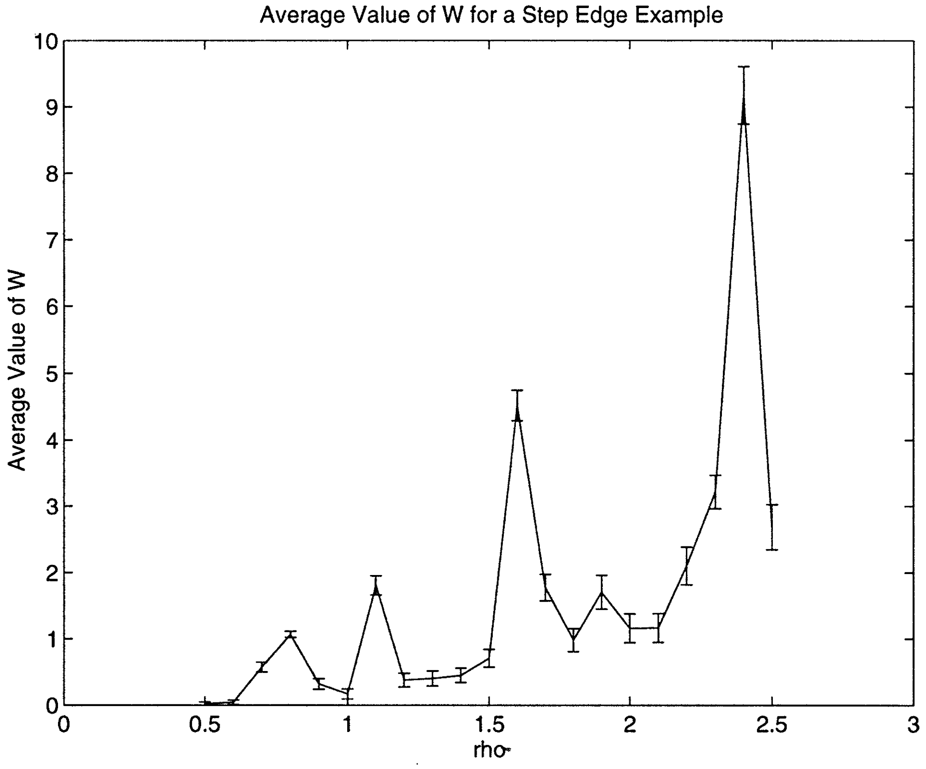
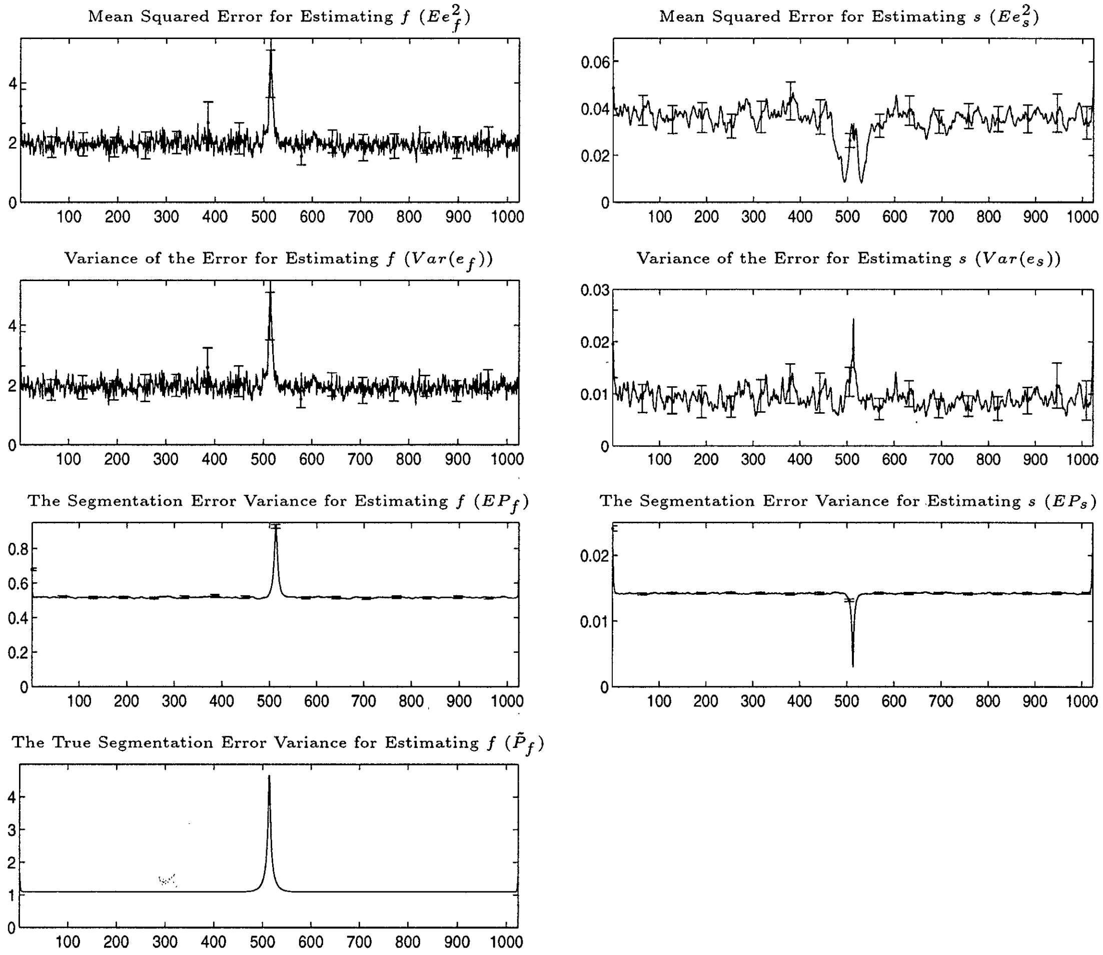


Figure 3-11: Average value of W for the step edge example of Figure 3-4 but for different values of the parameter ρ.

these two cases, $Var(e_f) \approx \tilde{P}_f$ away from the edge, but $Var(e_f)$ is significantly larger than $\tilde{P}_f$ close to the edge. One also observes that Ee_s^2 is fairly large near the edge and small away from the edge; although, $Var(e_s)$ is relatively small everywhere. One concludes that the estimates of the edge function s are less peaked near the edge. The explanation for these observations is that the algorithm thinks the data is poor when r is large; so, the algorithm is hesitant to produce a localized edge function and smoothes the data significantly. The result is that the estimates of the edge and piecewise smooth functions are good away from the edge, but large errors are incurred in the vicinity of the edge. This indicates that there is a tradeoff in setting the value of the parameter r. Smaller values of r will often lead to a more peaked edge function and better estimates of the piecewise smooth function near the actual edge location. Larger values of r result in less peaked edge functions but better estimates of the piecewise smooth function away from the edge.

The value of the parameter r also affects the relationship between $Var(e_f)$ and



(a) (b)

Figure 3-12: A comparison of various error statistics for $P_v = 5$, $r = 1$.

EP_f. In the figures for which $P_v = r$, $Var(e_f) \approx EP_f$ away from the edge, and $Var(e_f)$ is bigger than but on the same order of magnitude as EP_f close to the edge. For the case that $P_v = 5$, $r = 1$, one observes that $Var(e_f) \approx 5EP_f$. Although not pictured here, one also observes that $Var(e_f) \approx 10EP_f$ when $P_v = 10$, $r = 1$. This observation leads one to a guideline for setting the parameters. First, notice that the solution obtained by coordinate descent of the functional (3.10) depends not on the parameters r, λ, $b = \frac{\nu}{2\rho}$, and $c = \frac{\nu\rho}{2}$ but on the ratios $\frac{\lambda}{r}$, $\frac{b}{r}$, and $\frac{c}{r}$. So, the piecewise smooth and edge functions one would obtain for $r = \alpha P_v$ for some constant α are the same functions one would obtain if λ, b, and c were multiplied through by $1/\alpha$ and $r = P_v$. However, when the parameters are scaled so that $r = P_v$, the error variances P_f and P_s generated by the algorithm will scale by $1/\alpha$. The simulations indicate

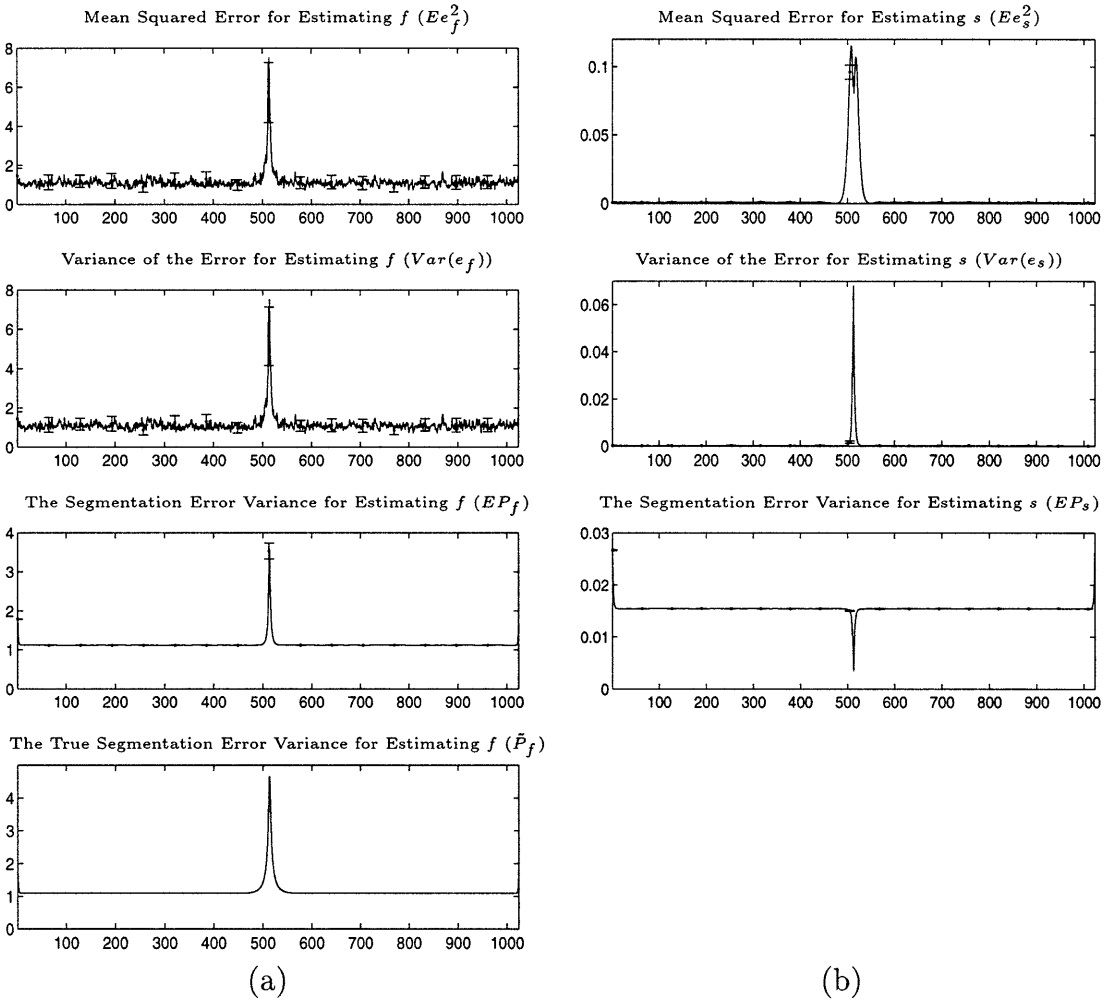
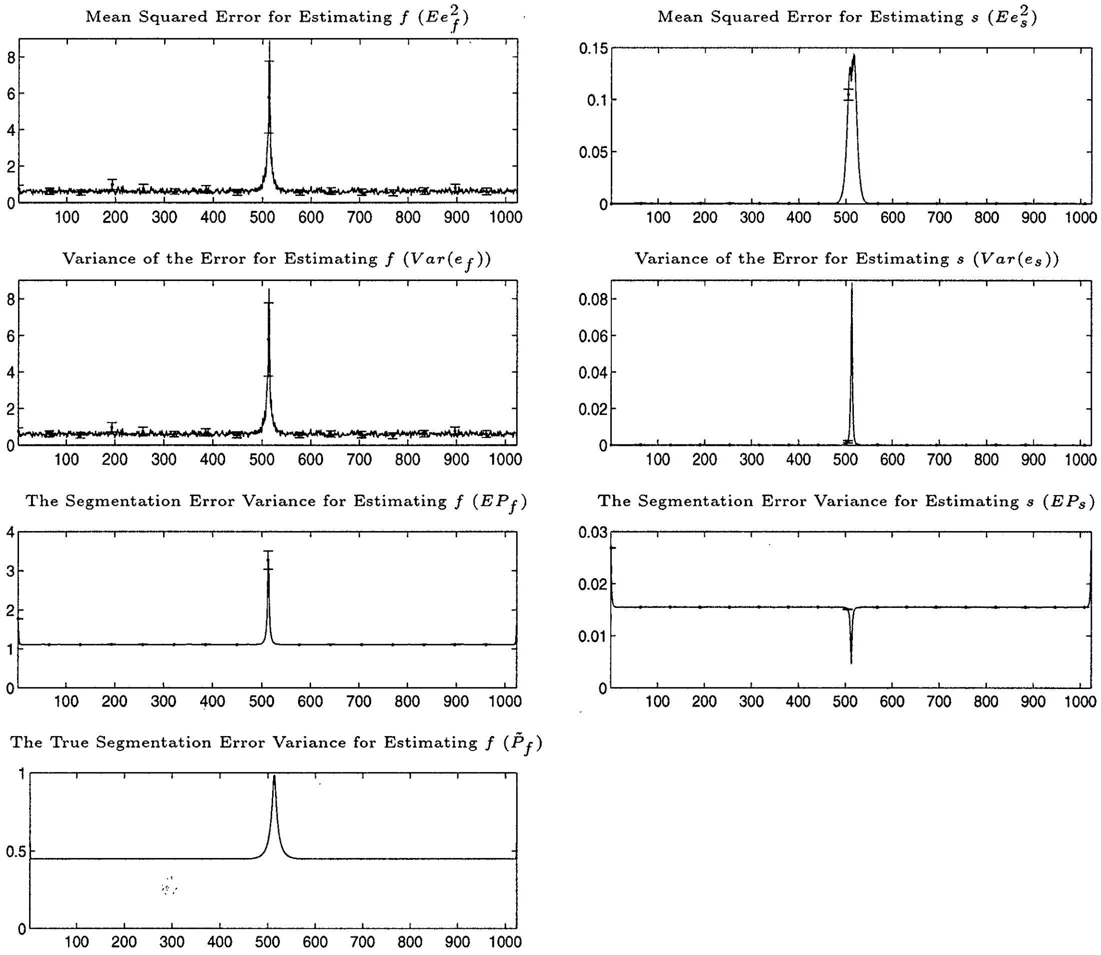


Figure 3-13: A comparison of various error statistics for $P_v = 5$, $r = 5$.

that the actual errors in the estimates closely match the error variances generated by the algorithm when the parameters are scaled such that $r = P_v$. Thus, if one were to set $r = \alpha P_v$ for $\alpha < 1$ in order to get accurate edge localization, then one should rescale the parameters so that $r = P_v$ in order to obtain accurate estimates of the error variances.

The previous set of results analyze how the error statistics are affected by different settings of P_v and r. The next result examines how the quantity W defined in Section 3.2.3 is affected by different values of P_v when the parameter r is kept fixed. The result displayed in Figure 3-15 is computed exactly as in Figure 3-9 except all the parameters of the algorithm are kept fixed to the values in Table 3.2, and only the actual noise variance is changed. Thus, the plot in Figure 3-15 is of W versus



(a) (b)

Figure 3-14: A comparison of various error statistics for $P_v = 1$, $r = 5$.

measurement noise standard deviation $\sqrt{P_v}$ when the measurement noise parameter $r = 0.2$. This is in contrast to the plot in Figure 3-9, in which the parameter r varies along with the actual measurement noise used. The results are shown for 50 runs, and error bars are set at three standard deviations. The graph is very interesting. It dips slightly at the point where the actual measurement noise and r coincide, and thereafter, the graph increases dramatically, at a slightly faster rate than in Figure 3-9. More points are being declared edges in the high noise region of Figure 3-15 than is warranted because the parameter r is set too low. The indication is that setting r too large has a slightly adverse affect, but setting r too small is relatively worse. Comparing these results with those of the previous paragraphs, one can conclude that although it is often advantageous to use a small value of r in order to get a sharply

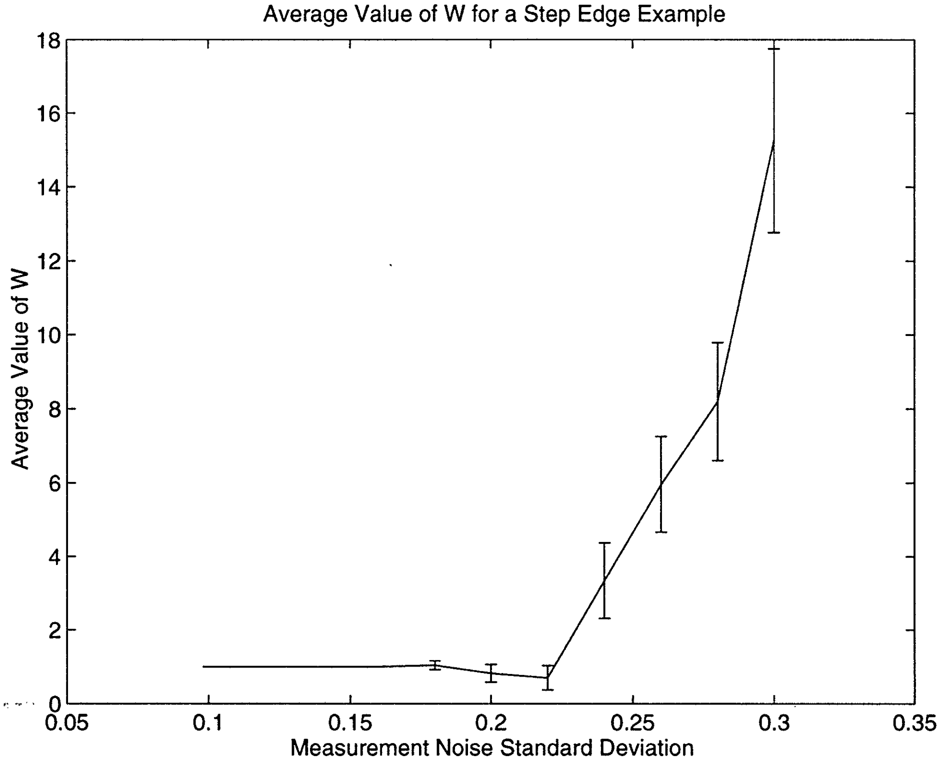


Figure 3-15: The average value of W for the step edge example of Figure 3-4 but with different levels of measurement noise added and r kept constant.

peaked edge function, this may lead to more locations being declared edges than would be otherwise declared if r were set equal to the measurement noise variance. The final guideline for selecting r is listed in the next section along with some general principals for selecting the other parameters.

3.2.5 Concluding Remarks

The results of the segmentation method presented in this section are quite promising. Each iteration of the algorithm is computationally efficient, and very few iterations are required to obtain convergence. Moreover, the algorithm is not only computationally efficient, but it is also capable of computing error statistics. In the case of the piecewise smooth function, these error statistics tend to coincide with the actual errors in the algorithm's estimates, which, in turn, are close to the optimal error for estimating the piecewise smooth function given the edge function. In addition, the empirical results indicate that the algorithm is robust to noise and small changes in the parameters.

3.3 Analysis

The simulations also have provided insight into tradeoffs in setting parameters. However, in order to make use of this segmentation algorithm, one needs a better understanding of it and how the choice of parameter values affects performance. One of the benefits of formulating a minimization problem in terms of statistics is that this yields a natural interpretation of the parameters. From this, one can obtain a loose set of guidelines, which were followed to pick the parameters for the examples of the previous section. One can obtain another perspective on the parameter choices from an analysis of the convexity of the one-dimensional discrete functional (3.10). The deterministic argument complements the statistical one, and the two together combined with the empirical results of the last section give one a good idea of the role of the parameters in the segmentation algorithm.

3.3.1 Parameter choices

First, note that each row of (3.15) can be written as

$$f_{i+1} - f_i = \frac{1}{(1-s_i)\sqrt{\lambda}} w_i^f \tag{3.21}$$

where w_i^f is zero mean Gaussian with unit variance. Defining $a_i = \lambda(f_{i+1} - f_i)^2$,

$$E[a_i] = \frac{1}{(1-s_i)^2}. \tag{3.22}$$

Examination of equations (3.19), (3.20), indicates that the derived quantity γ_i acts as an observation of the edge process. Thus, one would like γ_i to be close to zero where the gradient is small and close to one where the gradient is large. Since $\gamma_i = a_i/(a_i+b)$, one should set the parameter $b \gg 1$, where 1 is the expected value of a_i where there is no edge (*i.e.* (3.22) with $s_i \doteq 0$). However, one should not set b so large that γ does not approach one at edges.

Now, note that for a fixed value of M, one can use (3.20) to write

$$s_{i+M} - s_i = \frac{\sqrt{M}}{\sqrt{c}} \tilde{w}_i \tag{3.23}$$

where $\tilde{w}_i$ is a zero mean Gaussian random variable with unit variance. Thus,

$$E[(s_{i+M} - s_i)^2] = \frac{M}{c}. \tag{3.24}$$

If one desires that the edge process should change in value from its two extremes over an interval of length M, then one should pick the parameter $c \approx M$. Hence, c controls the width of the edges.

Thus, one has the following guidelines for parameter choices

- λ is inversely proportional to the amount of variability allowed in the piecewise smooth approximating signal f. One should pick a λ that matches one's expectations for the smoothness of the observed signal. Care should be taken so as to avoid picking λ's that are very extreme. Too large a choice of λ may lead to segmentations with many edge locations, as indicated by the simulation results in Figure 3-10. Fortunately, the algorithm is not too sensitive to the value of λ, as discussed in Section 3.2.3. Estimating λ within an order of magnitude will suffice.

- c is proportional to the width of the edges. Generally, c should be chosen small, but it should not be chosen so small that the edge function tracks all the subtle variations in the derivative. Erring on the large side will result in slightly thicker edges as indicated by the simulations result displayed in Figure 3-11. But this is not a major problem since the segmentation is not severely affected by small changes in this parameter, as discussed in Section 3.2.3.

- b controls the degree of edginess. One should generally pick $b \gg 1$; however, one must be careful not to pick b to be greater than the maximum possible value of

a for the type of discontinuity expected. In other words, choose

$$b \ll \lambda|\frac{df(x)}{dx}|^2, \text{for } x \text{ at edge locations.} \tag{3.25}$$

Not picking a b to satisfy this latter constraint can lead to segmentations where there are no edges. One observes this in Figure 3-11 for large values of b, which which correspond to small values of ρ since $\rho \propto 1/b$.

- r selects the variance of the measurement noise that the algorithm expects in the data. As discussed in regards to Figures 3-7 and 3-12 through 3-14, there is a tradeoff in setting r. Small values of r will yield better estimates of the edge and piecewise smooth functions near the edge, and large values of r will yield better estimates away from the edge. However, as discussed in regards to Figures 3-9 and 3-15, setting r smaller than the actual level of measurement noise may lead to the generation of spurious edge detections due to noise and consequently less accuracy in localizing the actual edge. Thus, one must carefully consider the amount of noise in the signal and weigh the tradeoffs when selecting this parameter.

These guidelines match Pien's choice of parameters in [19]. Recall that the parameters b and c are related to the parameters in equation (2.4) by:

$$\rho^2 = \frac{c}{b} \tag{3.26}$$

$$\nu^2 = 4bc. \tag{3.27}$$

Thus, by choosing $b \gg E[\lambda(f(i+1) - f(i))^2] = 1$ where there is no edge, one implies that the choice of ν should scale with the expected value of the derivative in the data. In [19], Pien gives segmentation examples for which ν is chosen to be on the same order as $|\nabla f|^2$. Thus, the guidelines outlined in this section are consistent with the values of the parameters used in Pien's work.

3.3.2 Convexity

The guidelines given above were derived with the goal of obtaining a good segmentation, but one can also consider how one should pick parameters so that the algorithm has good convergence properties. In particular, it is natural to ask under what conditions the discrete functional (3.10) is convex since this would imply that coordinate descent of the functional will converge to a global minimum. The following proposition gives a sufficient condition for convexity of the discrete functional (3.10).

Proposition *Let the function* $E : \mathbf{R}^n \times \mathbf{R}^{n-1} \longmapsto \mathbf{R}$ *be defined by*

$$E(f,s) = r^{-1}\sum_{i=1}^{n}(f_i-g_i)^2+\lambda\sum_{i=1}^{n-1}(f_{i+1}-f_i)^2(1-s_i)^2+b\sum_{i=1}^{n-1}s_i^2+c\sum_{i=1}^{n-2}(s_{i+1}-s_i)^2 \quad (3.28)$$

where r, λ, b, *and* c *are positive constants,* g *is observed data,* f *is a piecewise smooth approximating function, and* s *is an edge map such that* $s \in [0,1]$. *Then, if* B *is a positive constant,* $E(f,s)$ *is a convex function for*

$$(f,s) \in \left\{(f,s) \in \mathbf{R}^n \times \mathbf{R}^{n-1} | (f_{i+1}-f_i)^2 \leq B, i = 1\ldots n-1\right\} \quad (3.29)$$

provided

$$b - 3\lambda B \geq 0. \quad (3.30)$$

Proof: The proof comes in two parts. The first part decomposes E into a sum of terms, several of which are convex. The second part demonstrates that the conditions of the proposition guarantee that the remaining terms are convex. Since a sum of convex functions is convex, E is convex.

Now, $(f_i-g_i)^2$ and $(s_{i+1}-s_i)^2$ are convex functions; so, E is convex if the function K defined by

$$K(f_i, f_{i+1}, s_i) = \lambda(f_{i+1}-f_i)^2(1-s_i)^2 + bs_i^2 \quad (3.31)$$

is convex. To demonstrate this, we form the Hessian of K and show that it is positive semi-definite for $s_i \in [0,1]$ and $(f_{i+1} - f_i)^2 \leq B$. The Hessian is given by

$$\begin{pmatrix} 2\lambda(1-s_i)^2 & -2\lambda(1-s_i)^2 & 4\lambda(f_{i+1}-f_i)(1-s_i) \\ -2\lambda(1-s_i)^2 & 2\lambda(1-s_i)^2 & -4\lambda(f_{i+1}-f_i)(1-s_i) \\ 4\lambda(f_{i+1}-f_i)(1-s_i) & -4\lambda(f_{i+1}-f_i)(1-s_i) & 2\lambda(f_{i+1}-f_i)^2+2b \end{pmatrix} \tag{3.32}$$

To show positive semi-definiteness, we'll check that all of the determinants of the principal sub-matrices are greater than or equal to zero

- All of the diagonal terms are greater than or equal to zero.
- The upper left 2×2 submatrix and the whole matrix are singular and so their determinants are zero.
- The determinants of the bottom right 2×2 submatrix and the outer 2×2 submatrix are both equal to $4\lambda b(1-s_i)^2 - 12\lambda(f_{i+1}-f_i)^2(1-s_i)^2$. This quantity is greater than or equal to zero if and only if $b(1-s_i)^2 \geq 3\lambda(f_{i+1}-f_i)^2(1-s_i)^2$, which is guaranteed by (3.30).

So $J(f,s)$ is convex over the set specified in (3.29), which is convex since it is defined by linear constraints. □

This proposition is only sufficient for convexity. It is not clear whether it is necessary, but the derived condition is still interesting because of its marked similarity with one of the guidelines formulated from an analysis of the statistical formulation of the problem. The functional is convex if $b > 3\lambda B$ and $\lambda(f(i+1) - f(i))^2 \leq \lambda B$ for all i for all admissible f and some constant B. However, one of the guidelines derived for a good segmentation states that b should be chosen so that it will be less than the value of $\lambda(f(i+1) - f(i))^2$ at edge locations. Thus, it would appear that the parameters that make for a good segmentation are the ones that permit non-convex functionals. Although the goal was to demonstrate that there exist useful parameter values for which the functional is convex, this is still an interesting result that makes sense and is thoroughly grounded in the parameter discussion of this section.

3.4 Summary of One Dimensional Results

This chapter's analysis of the one dimensional segmentation problem has introduced in a simple framework many of the concepts and results that will be useful for an extension to the image segmentation problem. Using as a starting point the variational segmentation formulation of Shah, a pair of related statistical estimation problems was derived for use in an iterative algorithm to perform one dimensional segmentation. That algorithm is fast, gives good segmentations, and computes error statistics that may be useful in scientific applications. In addition, the parameters in the statistical formulation of the problem can be picked according to a set of guidelines that follow from simple properties of the statistical problems. The results are very promising and strongly motivate an extension to the two dimensional problem.

Chapter 4

Image Segmentation

The one-dimensional segmentation algorithm in the previous chapter satisfies the two main objectives set in Chapter 1 for a segmentation algorithm. It has a computational cost that grows proportionally to the number of data points, and it generates a set of error statistics. The goal is to extend this algorithm to two dimensions, keeping the overall structure the same. This structure is laid out in Figure 4-1. The principal components of the algorithm are the methods for computing estimates of the piecewise smooth and edge functions and the associated error statistics. The problem central to both of these components is how to compute an estimate of a two-dimensional process and associated error statistics given a map of the statistical variability in the underlying process. When computing an estimate of the piecewise smooth function, this map corresponds to the fixed edge function, and when computing an estimate of the edge function, this map is constant. In one dimension, this central problem is addressed by using a recursive estimation algorithm which computes the relevant quantities using the model equations (3.14), (3.15) and (3.19), (3.20) that specify an estimation problem equivalent to a discretization of Shah's variational problem. The problem is more difficult in two dimensions. Standard discretization of Shah's variational problem in two dimensions lead to estimation problems involving Gaussian Markov random field models defined on a rectangular lattice. As discussed in Chapter 2, any recursive estimation algorithm which can compute estimates and error variances given arbitrary Gaussian Markov random field priors must perform an

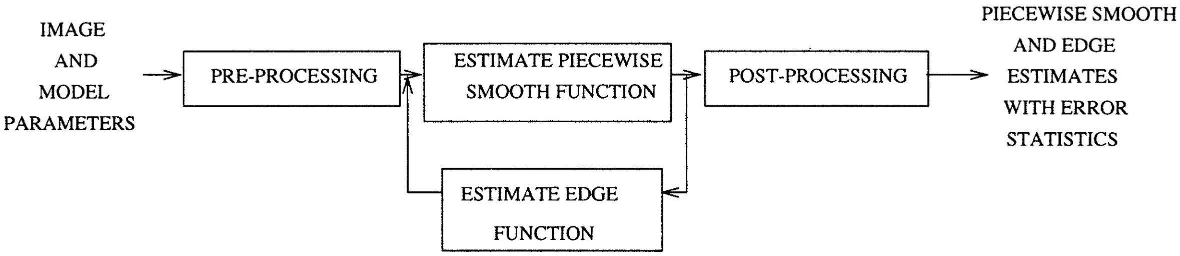


IMAGE
AND
MODEL
PARAMETERS
PRE-PROCESSING
ESTIMATE PIECEWISE SMOOTH FUNCTION
POST-PROCESSING
PIECEWISE SMOOTH AND EDGE ESTIMATES WITH ERROR STATISTICS
ESTIMATE EDGE FUNCTION

Figure 4-1: General structure of the two-dimensional algorithms.

unacceptably large $O(N^3)$ multiplications.

The focus of this chapter is to address, in the context of segmentation, the problem of computing estimates of a two-dimensional process and associated error statistics given a map of the variability in the underlying process. At the core of the two computational methods presented here is a recursive estimation algorithm that estimates a process given a map of its variability using, instead of a Markov random field prior model, a multiscale model. Such models are introduced in Chapter 2. The first computational method discussed involves an extension of $1/f$-like models [6] whose form is closely related to the one-dimensional models (3.14), (3.15), and (3.19), (3.20). The structure is chosen to ensure convergence of the iterative process of alternately estimating the piecewise smooth and edge functions. In order to remove artifacts, some additional processing is necessary before and after this iterative procedure, (see Figure 4-1). However, the pre- and post-processing complicates the algorithm, and, in particular, makes it difficult to analyze how to adjust the parameters so as to improve performance. The second computational method presented in this chapter is formulated so as to remove the necessity of pre- and post-processing. This simplifies the algorithm and allows one to analyze more easily the role of the parameters and the structural elements of the algorithm and then adjust them to obtain better results. This latter method makes use of a slightly different multiscale model, the thin plate model [6], which is also discussed in Chapter 2. Both computational methods produce good results as part of a segmentation algorithm, but the latter method shows more promise.

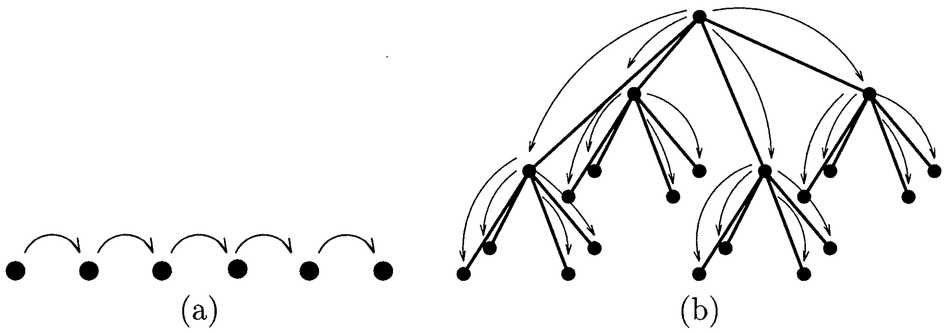


Figure 4-2: Progression of recursion in one-dimensional (a) and quad tree models (b).

4.1 Segmentation with 1/f-Like Models

The $1/f$-like multiscale models for segmentation are formed in analogy with the discrete one-dimensional models (3.14), (3.15) and (3.19), (3.20). These can be written as causal one-dimensional models. Since the structure of a quad-tree is linear in nature, one can form an extension of the causal one-dimensional models to multiscale ones involving recursions progressing from the root node to the leaves on a quad tree as diagrammed in Figure 4-2. The resulting multiscale models have strong connections to models used in previous works for estimating surfaces. For the case when there are no edges, the multiscale model is exactly the multiscale model developed by Fieguth [8] to estimate the surface of the ocean. The ocean surface is described well by a prior model whose power spectrum decays as $1/f$. Fieguth developed the multiscale model so as to be appropriate for surfaces described by such $1/f$ prior models; hence, these multiscale models are termed $1/f$-like. Their successful application to estimating the surface of the ocean motivates the application of the subsequent related multiscale models to image segmentation.

4.1.1 Derivation of the Algorithm

Any equation of the form (3.14), (3.15) and (3.19), (3.20) can be rewritten as a causal state space recursion provided the matrices L, S, and A have a suitably sparse structure. In particular, one can rewrite (3.14), (3.15) as the recursion

$$g_i = f_i + \sqrt{r} v_i^f \tag{4.1}$$

$$f_{i+1} = f_i + \frac{1}{(1-s_i)\sqrt{\lambda}} w_i^f \tag{4.2}$$

and (3.19), (3.20) as the recursion

$$\gamma_i = s_i + \frac{1}{\sqrt{\lambda(f_{i+1}-f_i)+b}} v_i^s \tag{4.3}$$

$$s_{i+1} = s_i + \frac{1}{\sqrt{c}} w_i^s \tag{4.4}$$

where w_i^f, v_i^f, w_i^s, and v_i^s are all independent unit variance Gaussian random variables. The two estimation problems described by these sets of equations form the core of the one-dimensional segmentation algorithm presented in the last chapter. Their form also motivates the multiscale model presented here.

Before discussing that model, some notation needs to be introduced to simplify the discussion. The plain letters f and s will denote multiscale processes or admissible multiscale functions in a minimization problem according to the context. Fixed multiscale functions appearing in the model equations for a multiscale process or in a functional of a minimization problem will be denoted by $\tilde{f}$ and $\tilde{s}$. The final estimates produced upon termination of the segmentation algorithm are denoted by $\hat{f}$ and $\hat{s}$.

Now, the specific multiscale model described here is an extension of the $1/f$-like models presented in [6] and [8]. Recall that one can specify a multiscale model using a root-to-leaf recursion of the form (2.10) with observations of the form (2.11). Let f be a multiscale process whose finest scale corresponds to a piecewise smooth image and whose other nodes correspond to aggregates of their four children. Then, by analogy with the one-dimensional segmentation algorithm, it is natural to construct an image segmentation algorithm which consists of alternately estimating f given a multiscale edge function $\tilde{s}$ and data g and estimating s given a particular $\tilde{f}$, where the equations describing the estimation problem are the multiscale equivalents of (4.1), (4.2) and (4.3), and (4.4). The problem of estimating f given a particular multiscale edge function $\tilde{s}$ is described by

$$g_\nu = f_\nu + \sqrt{r_\nu} v_\nu^f \tag{4.5}$$

$$f_\nu = f_{\nu\bar{\gamma}} + \frac{1}{(1-\tilde{s}_\nu)\sqrt{\lambda}d_\nu} w_\nu^f, \tag{4.6}$$

where v_ν^f and w_ν^f are independent Gaussian random variables with unit variance; the independent observations g_ν exist only at the finest scale and have a measurement noise variance r_ν; and d_ν are a set of constants that decrease geometrically as the scale becomes finer [6]. Likewise, one can write the problem of determining the multiscale edge function s given a specific multiscale piecewise smooth image intensity function $\tilde{f}$ as

$$\gamma_\nu = s_\nu + \frac{1}{\sqrt{a_\nu + b}} v_\nu^s \tag{4.7}$$

$$s_\nu = s_{\nu\bar{\gamma}} + \frac{1}{\sqrt{c}d_\nu} w_\nu^s \tag{4.8}$$

where

$$a_\nu = \lambda d_\nu^2 (\tilde{f}_\nu - \tilde{f}_{\nu\bar{\gamma}})^2 \tag{4.9}$$

$$\gamma_\nu = \frac{a_\nu}{a_\nu + b}, \tag{4.10}$$

and v_ν^s and w_ν^s are independent Gaussian random variables with unit variance. One now has a pair of estimation problems from which one can start to build an image segmentation algorithm. The principal novelty of the models in these two-dimensional estimation problems is the use of a first difference between scales in lieu of a local difference operator acting in a plane. This has the major advantage that calculating estimates using these priors is fast. One disadvantage is the introduction of some artifacts.

These artifacts arise because of the correlation structure associated with multiscale tree models. To understand how this can lead to problems, consider the model specified by (4.6) when the edge function $\tilde{s}$ is zero everywhere. Since the finest scale nodes of f correspond to a piecewise smooth version of the image, one desires that the prior model (4.6) will imply that nearest neighbors of f at the finest scale have relatively large correlation. Consider f_{ν_1} and f_{ν_2} in Figure 4-3. They are nearest

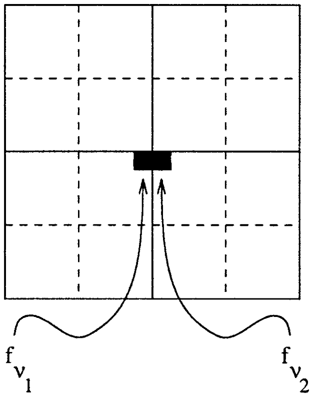


Figure 4-3: Even in the absence of edges, two process values on the finest scale of a quad tree, such as f_{ν_1} and f_{ν_2}, can be physically close together but have a small correlation.

neighbors at the finest scale, but their correlation is relatively low because the two paths from the root node to each of f_{ν_1} and f_{ν_2} share no branches in the tree. Thus, using the model equations (4.5), (4.6) to estimate a piecewise smooth function may lead to an estimate with undesirable edge features even when no edges are assumed present. Due to the nature of the quad tree prior, these artifacts are generally blocky.

The undesirable side effects of working with a multiscale tree prior can be diminished by working within the overlapping tree framework of Fieguth and Irving [7]. The general idea is to allow overlap between the regions represented by process values at a particular scale. For example, each of the finest scale process values depicted normally correspond to a single pixel, but in an overlapping framework, two different finest scale process values may correspond to the same pixel. Thus, one may designate f_{ν_1} and f_{ν_2} to represent the same image pixel in Figure 4-3. In order to estimate an image with an overlapping tree multiscale prior, one must duplicate measurements methodically, then estimate using a multiscale prior, and finally combine multiscale estimates to create an estimate of one's image. The additional operations occurring before and after the estimation process are termed injecting into the lifted domain and projecting back into the original domain respectively. These steps are performed using non-invertible linear operators that have a very particular structure to allow for rapid computation and good results. The general scheme for computing using overlapping trees is diagrammed in Figure 4-4. The example displayed in Figure 2-3

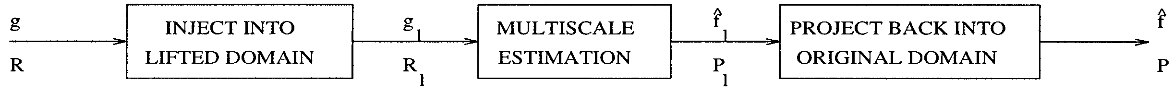


Figure 4-4: Using overlapping tree framework to compute estimates $\hat{f}$ with error covariance P from data g with measurement noise R.

was computed in the overlapping framework using Fieguth's multiscale models for surface reconstruction [6]. As one can observe, the results are quite good.

However, one has to be careful how one incorporates the overlapping framework into the segmentation algorithm for the same reason one has to be careful in one dimension with selecting a sampling framework. Namely, if one desires convergence of the process of alternately forming an estimate of the piecewise smooth function f and an estimate of the edge function s, then one should ensure that this process is performing coordinate descent of some functional. The problem is that the scheme outlined in Figure 4-4 for forming estimates in the overlapping tree framework does not necessarily correspond to finding the minimum of some functional. Thus, a segmentation algorithm making use of overlapping trees may have a problem with convergence, just as described in the previous chapter when discussing issues of sampling. One solution is diagrammed in Figure 4-5. One injects the data into the lifted domain once. Then, one alternates between estimating the piecewise smooth function f given the edge function $\tilde{s}$ and using the model (4.5), (4.6) and estimating the edge function s given the piecewise smooth function $\tilde{f}$ and using the model (4.7), (4.8). By construction, this iterative process is performing coordinate descent of the functional

$$E(f,s) = \sum_{\nu\in\mathcal{F}} r_\nu^{-1}(g_\nu - f_\nu)^2 + \sum_{\nu\in\mathcal{T}-\{0\}} \left(\lambda(f_\nu - f_{\nu\bar{\gamma}})^2(1-s_\nu)^2 + bs_\nu^2 + c(s_\nu - s_{\nu\bar{\gamma}})^2\right), \tag{4.11}$$

where $\mathcal{F}$ denotes the nodes of the tree at the finest scale and $\mathcal{T}-\{0\}$ denotes all nodes of the tree except the root node. This coordinate scheme will converge, and when it does, one can then project the estimates back into the original domain to get a final segmentation.

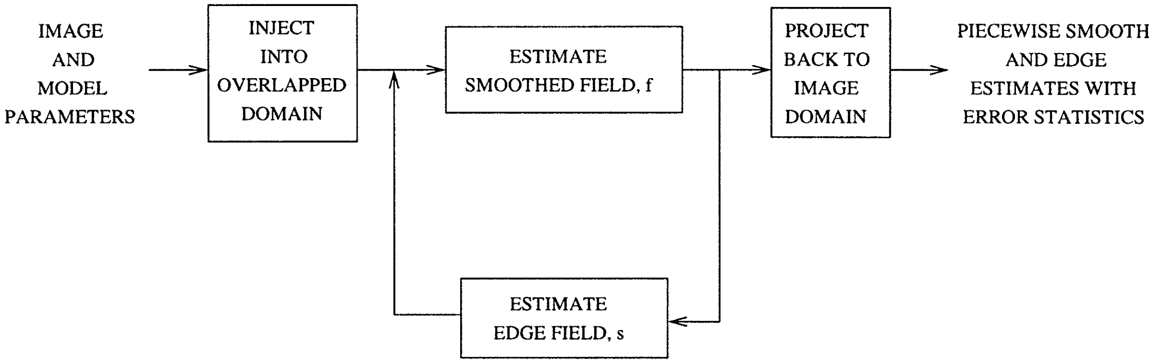


IMAGE AND MODEL PARAMETERS
INJECT INTO OVERLAPPED DOMAIN
ESTIMATE SMOOTHED FIELD, f
PROJECT BACK TO IMAGE DOMAIN
PIECEWISE SMOOTH AND EDGE ESTIMATES WITH ERROR STATISTICS
ESTIMATE EDGE FIELD, s

Figure 4-5: Multiscale segmentation algorithm formed by using overlapping projection operations just once.

4.1.2 Results

The segmentation algorithm making use of $1/f$-like multiscale models is almost completely specified. It remains only to list some of the implementation details and introduce the parameters associated with these details.

- *Prior Covariances*: As in the one-dimensional case, one must specify a pair of prior covariances P_0^f and P_0^s for use when estimating the piecewise smooth function and edge function respectively. In the multiscale case, these quantities are the prior covariances of the process values at the root nodes. Making use of reasoning analogous to that presented in the last chapter, P_0^f and P_0^s are typically set to relatively large values.

- *Clipping*: As in the one-dimensional case, one must clip the edge function estimates so that they lie in $[0, 1-\epsilon]$ for some small ϵ.

- *Initialization*: The algorithm is started by estimating the piecewise smooth function f assuming that the edge function $\tilde{s}$ is zero everywhere.

- *Stopping Criterion*: In the one-dimensional algorithm, the iterations are stopped when the change in the value of the functional being minimized falls below a given threshold. This criterion is not valid if the edge function s is persistently clipped. Clipping rarely occurs in the one-dimensional examples presented. However, the edge function is often clipped in the subsequent results. Thus, the

Parameter	*Value*
λ	1
b	1
c	1
P_0^f	10000
P_0^s	10000
ϵ	1.0×10^{-2}
I	5
T	0.9
$\mathcal{O}$	$\begin{pmatrix} 10 & 5 & 3 & 2 & 1 & 0 & 0 \end{pmatrix}$

Table 4.1: Parameter values for the synthetic example in Figures 4-6 and 4-7.

changes in the functional can not be used as a stopping criterion. The iterative process in this two-dimensional algorithm is stopped after a fixed number of iterations I. Typically $I = 5$ since the results indicate that the process has almost converged after five iterations.

- *Overlap Operators*: Making use of the overlapping computational framework presented in [6], one can specify the particular operators with a single vector of parameters $\mathcal{O}$. For an interpretation of their meaning, one should consult [6].

- *Imaging the Piecewise Smooth and Edge Function Estimates*: In the model equation (4.5), the image is an observation of the finest scale of f. Thus, one can use the finest scale estimates $\hat{f}$ to form an image of the piecewise smooth function estimate. However, the finest scale of the estimate of the edge function $\hat{s}$ does not necessarily correspond to an edge function estimate for the image because coarse scales of $\hat{s}$ provide information about the edges in the image that is not available at the finest scale of $\hat{s}$. Nonetheless, it is reasonable to consider the finest scale of $\hat{s}$ to correspond to an edge function of the image. The results that follow display the finest scale of $\hat{f}$ and $\hat{s}$ along with $\hat{s}^{Th}$, a thresholded image of the finest scale of $\hat{s}$ defined by

$$\hat{s}_{ij}^{Th} = \begin{cases} 1 & \text{if } \hat{s}_{ij} \geq T \\ 0 & \text{if } \hat{s}_{ij} < T \end{cases} \qquad (4.12)$$

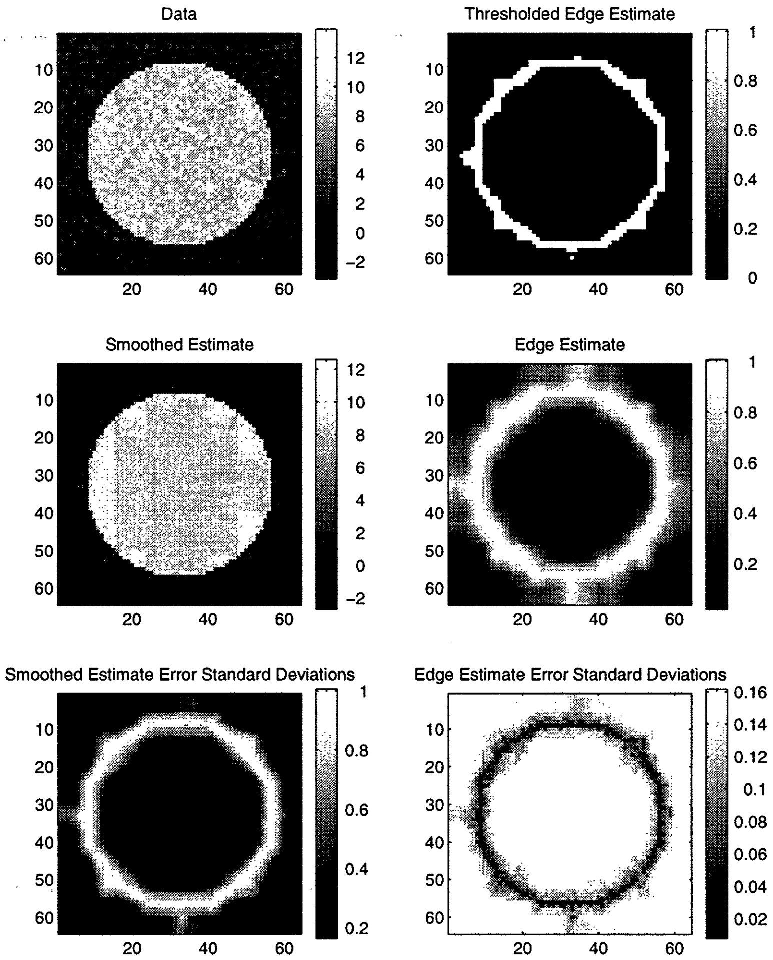
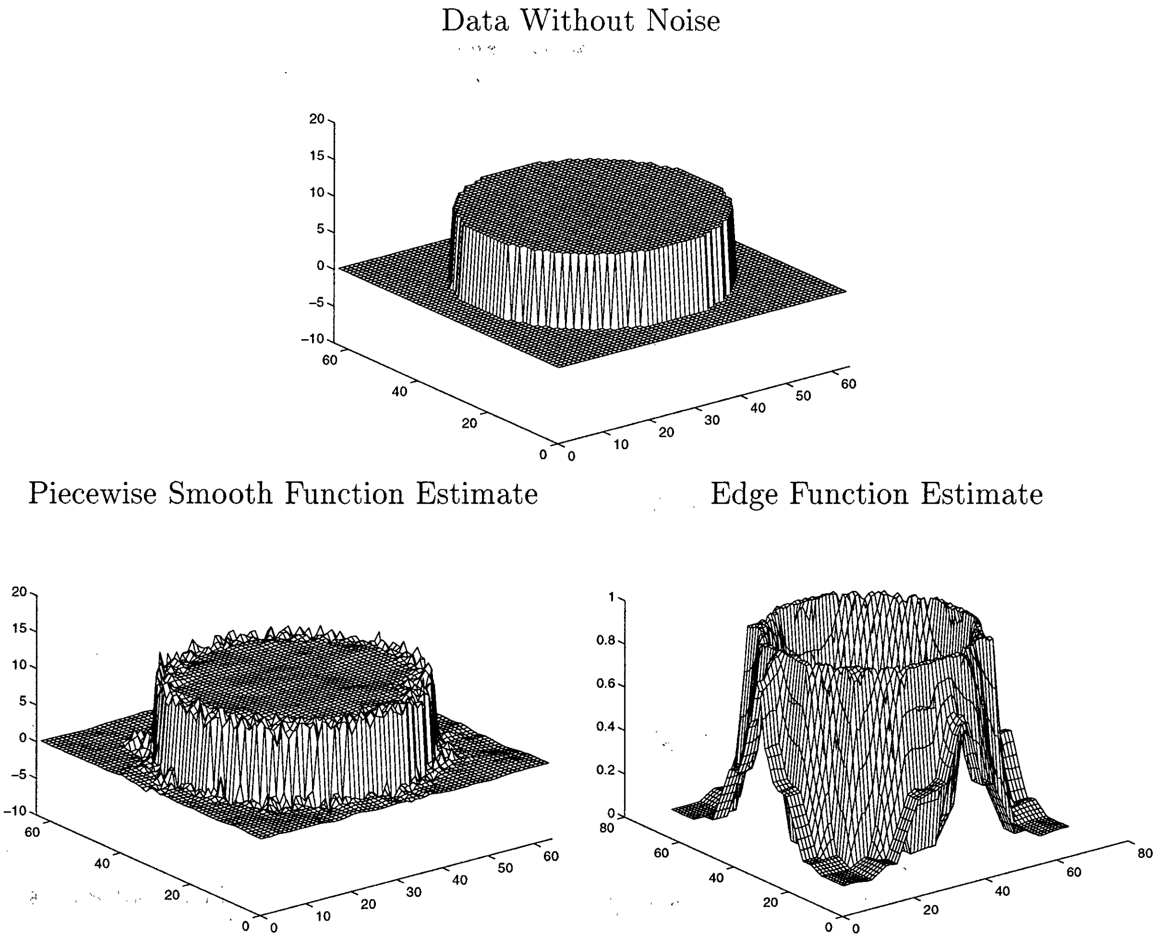


Figure 4-6: A circle segmentation example.

where the subscript ij denotes the (i, j) position in the image plane $(0, \ldots, n-1) \times (0, \ldots, n-1)$ and T is a constant approximately equal to one. The thresholded edge function estimate provides an indication of how well the edges are delineated by $\hat{s}$.

The parameters P_0^f, P_0^s, ϵ, I, $\mathcal{O}$, and T that have been introduced here as well as the previously mentioned parameters λ, b, and c are input to the segmentation algorithm along with the data.

A segmentation example for a circular step is presented in Figures 4-6 and 4-7. The data consists of a piecewise constant intensity function with a circular step of height 10 added to unit intensity white Gaussian noise. For this example, the algorithm uses the parameters listed in Table 4.1 and measurement noise parameters r_ν of unit



Data Without Noise

Piecewise Smooth Function Estimate

Edge Function Estimate

Figure 4-7: Mesh plots of the circle segmentation example.

value. As the table indicates, the number of iterations $I = 5$. The results for 10 iterations are not much different, indicating that 5 iterations is adequate to achieve convergence. A qualitative examination of the results in Figures 4-6 and 4-7 indicates that the segmentation is good. The piecewise smooth function estimate is smooth, especially far away from the circular edge. The edge function estimate also takes on the desired characteristic that it is close to one in the vicinity of the circular step and is close to zero away from the step. The thresholded edge function estimate provides a good binary map indicating the presence or absence of an edge. The error standard deviations have a similar structure and interpretation as those in one dimension. One thing that is noticeable in all of the results in Figure 4-6 is the presence of some blockiness despite the use of overlap. This blockiness is most noticeable in the edge function estimate, in which there is an undesirable broadening in certain

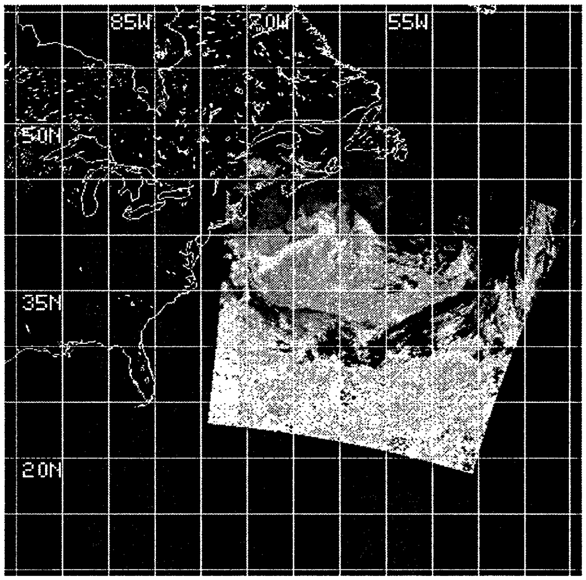


Figure 4-8: AVHRR data of the North Atlantic on June 2, 1995 at 5:45:01 GMT.

preferred directions along the ridge where the function is close to one. Nonetheless, the segmentation algorithm generates a fairly accurate piecewise smooth and edge function estimates and motivates an investigation into the algorithm's use on a real image.

The real images to be considered are formed from Advanced Very High Resolution Radiometer (AVHRR) data of the North Atlantic taken by NOAA satellites. The AVHRR images in this thesis were obtained from an online database maintained by The University of Rhode Island Graduate School of Oceanography. They are gray scale images which portray water temperature. Lighter tones correspond to warmer water, and darker tones to cooler water. A sample image with a coastline overlay is displayed in Figure 4-8. Data exists only for the lightly colored square-like region. One observes a thin northeast running warmer body of water off the coast of the Carolinas. This is the Gulf Stream. The position of the Gulf Stream is important to the shipping industry and oceanographers, among others. The features that are important for a segmentation algorithm to pick out are the north and south walls of the Gulf Stream and the occasional eddies that break off to the north and south of the Gulf Stream called warm and cold core rings respectively. The task of picking out these features is a good test of the segmentation algorithm because the underlying

Parameter	*Value*
λ	10
b	0.01
c	0.001
P_0^f	10000
P_0^s	10000
ϵ	1.0×10^{-2}
I	5
T	0.75
$\mathcal{O}$	$\begin{pmatrix} 10 & 5 & 3 & 2 & 1 & 0 & 0 \end{pmatrix}$

Table 4.2: Parameter values for the AVHRR data example in Figure 4-9.

data match our model. The task is difficult because there is also a significant amount of measurement error. The principal source of measurement error is the presence of clouds. The effect of clouds is to depress the temperature from its actual value. In particular, one can observe bands of black running through the image in Figure 4-8. These correspond to measured temperatures of zero degrees Celsius or lower. Such measurements are considered so poor that they are simply designated points where there exists no measurement. Clouds are present in other areas, but the measurements are not depressed below freezing. One has no separate knowledge of such locations; so, one can not mask them out from the data. The result is that the measurement error may be large in certain areas of the image. This can complicate matters.

Figure 4-9 displays a sample segmentation of AVHRR data. The parameters fed to the algorithm are listed in Table 4.2. The image is a portion of the North Atlantic image that is displayed in Figure 4-8. There are a handful of points in this image for which there is no data due to dense cloud cover. These points are marked as black in Figure 4-10. At locations where there exist measurements, the measurement noise parameters r_ν are set to unit value. The results in Figure 4-9 indicate that the algorithm is able to do a fair job of picking out the north wall of the Gulf Stream. However, the edge function estimate indicates that the algorithm is declaring edges within the Gulf Stream. From the thresholded edge function estimate, one can observe that points of the edge function estimate within the Gulf Stream are larger than points near the south wall. Now, the south wall is more difficult to isolate because

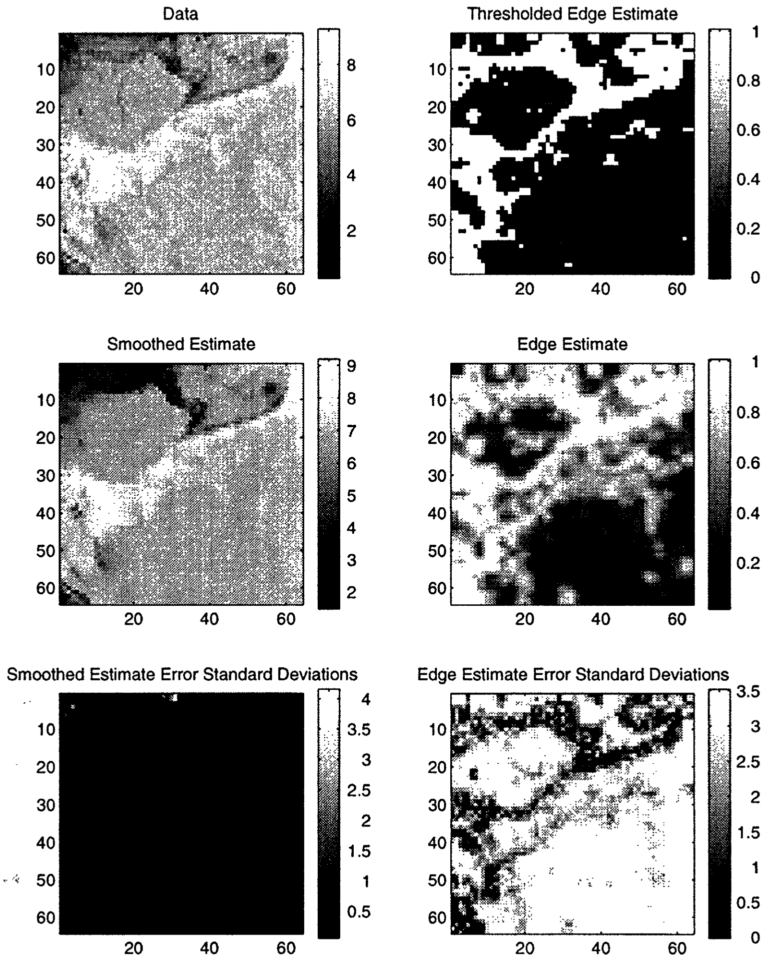


Figure 4-9: An AVHRR data segmentation example.

the temperature gradient is not as steep there, but it is still important to discern the boundary between the Gulf Stream and the waters to the south. Nonetheless, the algorithm is able to reasonably pick out the north wall, and this is more important in applications than being able to pick out the south wall precisely because the temperature gradient is steeper on the north side.

4.1.3 Analysis

These results indicate that the algorithm performs reasonably well on simple images such as the circle in Figure 4-6, but the performance is not very good on images such as the one in Figure 4-9, in which there is more variability. One of the reasons why the performance is poor is that the algorithm is difficult to tune because the

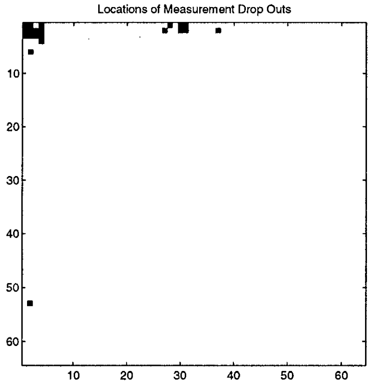


Figure 4-10: Location of data drop outs for the example in Figure 4-9.

operations associated with the overlap are difficult to analyze. Another reason is that information about the edges in the image is lost because of the method for imaging the edge function estimate.

The use of overlap for multiscale estimation was originally proposed as a method for using multiscale techniques to solve estimation problems associated with Markov random field priors [7]. In this framework, the operations associated with the overlap are applied just before and just after the multiscale estimation step as indicated in Figure 4-4. This contrasts with the overlap operator usage in the segmentation algorithm. As indicated in Figure 4-5, the overlap in the segmentation algorithm just presented is not applied before and after each estimation step. This is to ensure convergence of the algorithm. Although the results indicate that the segmentation algorithm works fairly well when the overlap is used as in Figure 4-5, analyzing the algorithm with the intention of improving it is difficult because the overlap was not designed to be used in this fashion.

Another problem with this image segmentation algorithm is the difficulty in forming an image from the multiscale edge function estimate $\hat{s}$. The edge function estimates displayed in Figures 4-6 and 4-9 are the finest scale of $\hat{s}$. However, the other values of the edge function estimate provide information about the edges in the image. One would like to use all of the multiscale edge function estimate to form an edge image from which one can easily pick out edges in the original image. Although this problem seems tractable, it is rather tricky. Rather than tackle this and the overlap problem head on, the next section presents a different multiscale segmentation algorithm in which problems concerning overlap and imaging the edge function estimate are not as difficult to address.

4.2 Segmentation with Thin Plate Models

The goal of this section is to present another multiscale algorithm which can compute an estimate of a two-dimensional process given a map of its underlying statistical variability. This multiscale method can be incorporated into the segmentation algorithm diagrammed in Figure 4-1 in such that way that no processing steps are required before or after the iterative procedure of alternately forming piecewise smooth and edge function estimates. The lack of any pre- or post-processing simplifies the structure of the overall algorithm, making it easier to understand and analyze for possible improvements. The starting point for the development of this segmentation algorithm is work done by Fieguth demonstrating the use of a particular multiscale method for reconstructing surfaces from noisy observations and given a binary map of edge locations [6]. Fieguth's computational technique is incorporated, as a black-box, into the algorithms used for computing the piecewise smooth and edge function estimates.

4.2.1 Derivation of the Algorithm

At the core of Fieguth's method for reconstructing surfaces from noisy observations and an edge map is a recursive algorithm for computing the necessary estimates and error variances using a particular multiscale model. The underlying multiscale model is an extension of the multiscale thin plate model discussed in Section 2.2 that performs surface estimation without knowledge of the discontinuities. That model results in surface estimates that are similar to the results obtained by finding the minimum of the functional

$$E(f) = \int \left((f-g)^2 + \alpha(p_x^2 + p_y^2 + q_x^2 + q_y^2) + \beta(|\nabla f|^2)\right). \tag{4.13}$$

Notice that the principal difference in the form of this functional and that of the thin membrane functional (2.1) is the presence of second-order derivatives in (4.13).

Likewise, the principal difference between the thin plate multiscale model and the $1/f$-like model (4.5), (4.6) when there are no assumed edges ($\tilde{s} = 0$) is that the thin plate model incorporates second-order multiscale derivatives by augmenting the state vector to include multiscale first-order derivatives of the surface. Discontinuities are taken into account in the thin plate multiscale model by introducing greater uncertainty in the scale-to-scale recursion where the line connecting the descendant to the parent node crosses a discontinuity. This is illustrated by the example in Figure 4-11. The thick line in the figure represents a known discontinuity. Let ν_i for $i \in \{0, 1, \ldots, 20\}$ denote the nodes of the quad tree on which the multiscale model is defined. Each node ν_i corresponds to a particular region in the image depending on the scale and position of the node in the tree. Dots are placed in the midpoints of these regions and are shaded according to the scale of the corresponding node ν_i: the darker the dot, the coarser the scale. For this example, greater uncertainty is added into the recursion from ν_0 to ν_1 and ν_2, from ν_1 to ν_8, from ν_2 to ν_{11} and ν_{12}, and from ν_3 to ν_{13}, but no additional uncertainty is added in the other parent-child recursions. The additional uncertainty is introduced in the form of higher process noise and a small modification in the recursive model so that gradient information contained in the state variable is not propagated across discontinuities. Since this technique is viewed as a black-box component of the segmentation algorithm, the specific details are not important to the current discussion and can be found in Appendix A and [6]. The important point is that Fieguth's algorithm provides one with a method for estimating a surface given a binary map of discontinuities.

The multiscale image segmentation algorithm presented in this section uses Fieguth's multiscale thin plate techniques for surface reconstruction to compute estimates of both a piecewise smooth function f and edge function s. The discontinuity map used to compute an estimate of f given a fixed estimate $\tilde{s}$ is $\tilde{s}^{Th}$, a thresholded

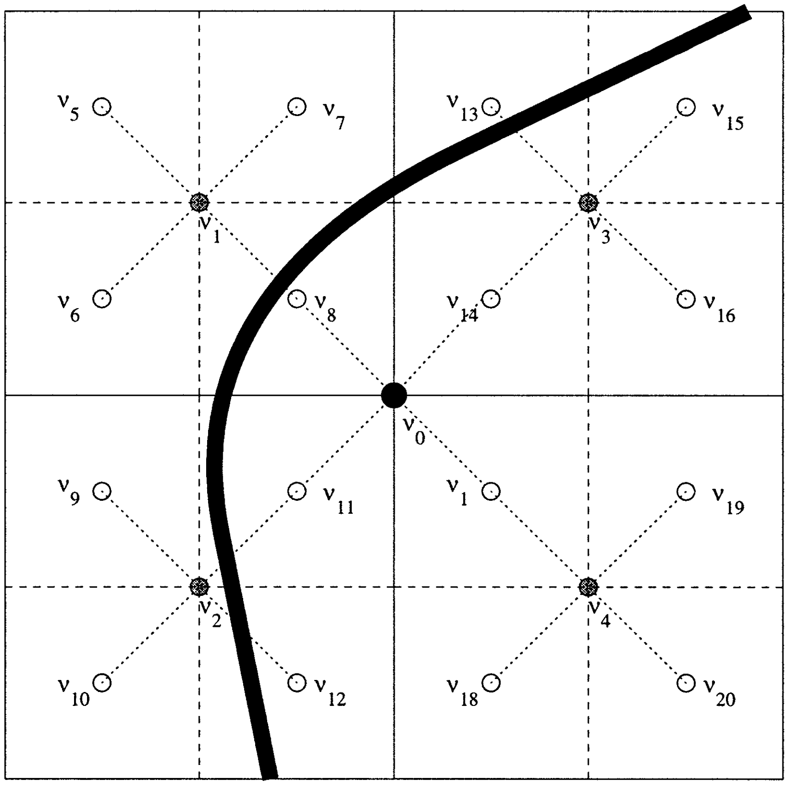


Figure 4-11: How discontinuities are incorporated into the thin plate multiscale model.

version of $\tilde{s}$ defined by

$$\tilde{s}^{Th}_{ij} = \begin{cases} 1 & \text{if } \tilde{s}_{ij} \geq \tau \\ 0 & \text{if } \tilde{s}_{ij} < \tau \end{cases} \tag{4.14}$$

where the subscript ij denotes the (i, j) position in the image plane and τ is a constant approximately equal to one. The map $\tilde{s}^{Th}$ is fed along with the data g and observation measurement noise levels to the multiscale thin plate algorithm to compute an estimate of a piecewise smooth function f. A multiscale thin plate algorithm is also used to estimate s given a fixed estimate $\tilde{f}$. The prior used to estimate s is always a fixed multiscale thin plate model which assumes no discontinuities are present in s. The measurements for s are formed from $\tilde{f}$, in analogy to (3.19) and (4.7), as follows. Let $\tilde{f}_{ij}$ denote the value of $\tilde{f}$ at position (i, j) in the image plane $(0, \ldots, n-1) \times (0, \ldots, n-1)$. For $(i, j) \in (0, \ldots, n-2) \times (0, \ldots, n-2)$, define

$$|(\nabla \tilde{f})_{ij}|^2 = (\tilde{f}_{(i+1)j} - \tilde{f}_{ij})^2 + (\tilde{f}_{i(j+1)} - \tilde{f}_{ij})^2 \tag{4.15}$$

in analogy to the magnitude of the gradient squared in a continuous setting. Then, one can make the following definitions akin to (4.9) and (4.10):

$$\begin{aligned} a_{ij} &= \lambda|(\nabla\tilde{f})_{ij}|^2 && (4.16) \\ \gamma_{ij} &= \frac{a_{ij}}{a_{ij}+b}, && (4.17) \end{aligned}$$

As before, γ_{ij} acts as an observation of the edge function at position (i,j), and $1/(a_{ij}+b)$ is the observation error variance. One can now use the multiscale thin plate approach to compute s at positions (i,j) in the image plane $(0,\ldots,n-2)\times(0,\ldots,n-2)$. The methods described here for computing an estimate of f given a fixed estimate of s and vice versa form the computational components of an image segmentation algorithm.

4.2.2 Results

The key elements of the segmentation algorithm are specified, but, as with the other algorithms, there are a few implementation details and associated parameters that need to be listed.

- *Thin Plate Model Parameters*: In Appendix A, it is noted that the overlapping thin plate models require the specification of three parameters B_s, B_g, and R_p. The first of these is the standard deviation of the noise affecting the first-order derivatives in the thin plate model. The latter two parameters are the standard deviations of the noise affecting the second-order derivatives in the model. They have physical interpretations as discussed in [6].

- *Prior Covariances*: The multiscale model also requires the specification of prior covariances P_0^s and P_0^f, which are set to large values for the reasons discussed in regards to the one-dimensional algorithm.

- *Calculating the Line in the Discontinuity Model*: Additional uncertainty is incorporated into the multiscale model recursion at locations where the line connecting the midpoints of the regions corresponding to the parent and the child nodes cross a discontinuity. In a discrete setting, there is not a unique definition of a line connecting two points. For the examples in this section, the following definition of a line is applied. Let (x_0, y_0) and (x_1, y_1) be the integer valued coordinates of the end points in the image plane. Without loss of generality, assume $x_0 \leq x_1$. Then, the line connecting these points is the set

$$\left\{(x, y) \in \mathbf{Z}^2 | x_0 \leq x \leq x_1, y_x^{min} \leq y \leq y_x^{max}\right\} \tag{4.18}$$

where

$$\begin{aligned} m &= \frac{y_1 - y_0}{x_1 - x_0} && (4.19) \\ y_x^{min} &= \min(\lfloor (x - x_0 - 1)m + y_0 \rfloor, \lfloor (x - x_0)m + y_0 \rfloor) && (4.20) \\ y_x^{max} &= \max(\lfloor (x - x_0 - 1)m + y_0 \rfloor, \lfloor (x - x_0)m + y_0 \rfloor) && (4.21) \end{aligned}$$

- *Parameters in the Discontinuity Model*: Additional process noise is added into the first derivative terms of the thin plate model where the edge function exceeds a certain level along the line connecting the midpoints of the regions corresponding to the parent and child nodes. Thus, one must specify the variance of the added noise, η, and the threshold τ. One should set $\sqrt{\eta}$ to be approximately the size of the expected discontinuities and $\tau \approx 1$.

- *Parameters in Edge Function Observation*: The parameters λ and b affect the observation (4.17) for the edge function. From the form of (4.17), one observes that increasing λ will increase the sensitivity of the edge function to gradients in the data and that increasing b will decrease the number of points of the edge function close to one. This interpretation of λ and b can then help guide one to pick appropriate values.

- *Overlap Operators*: The estimation algorithm using this thin plate model is applied in an overlapping framework; so, one must also specify an overlap vector $\mathcal{O}$ chosen according to guidelines in [6].

Parameter	*Value*
λ	10
b	3
B_s	20
B_g	0.2
R_p	0.35
P_0^f	1×10^6
P_0^s	1×10^6
η	10
τ	0.9
ϵ	1.0×10^{-2}
I	2
T	0.9
$\mathcal{O}$	$\begin{pmatrix} 16 & 10 & 7 & 4 & 2 & 2 & 0 & 0 \end{pmatrix}$

Table 4.3: Parameter values for the synthetic examples in Figures 4-12 and 4-13, 4-15, and 4-16.

- *Clipping*: As before, the estimates of the edge function s are clipped so that they lie in $[0, 1 - \epsilon]$ for some small ϵ.

- *Initialization and Stopping Criterion*: As in the segmentation algorithm using $1/f$-like models, the algorithm is started by estimating the piecewise smooth function f assuming an edge function $\tilde{s}$ which is zero everywhere, and the algorithm is stopped after a fixed number of iterations I. Convergence is not an issue in choosing I since one is not performing coordinate descent of a functional. Typically $I = 2$ since this appears to generate good results.

- *Imaging the Piecewise Smooth and Edge Function Estimates*: The results subsequently plotted include the estimates $\hat{f}$ and $\hat{s}$ of the piecewise smooth and edge function estimates as well as $\hat{s}$ thresholded at some value T, where $T \approx 1$.

To summarize, the list of items input to the segmentation algorithm are the data, the measurement noise variances, and the parameters B_s, B_g, R_p, P_0^s, P_0^f, η, τ, λ, b, $\mathcal{O}$, ϵ, I, and T.

Figures 4-12 and 4-13 display a circle example using the parameters in Table 4.3 and assuming unit intensity measurement noise everywhere. The data is the same as that in Figures 4-6 and 4-7, but the results for estimating the edge function are better.

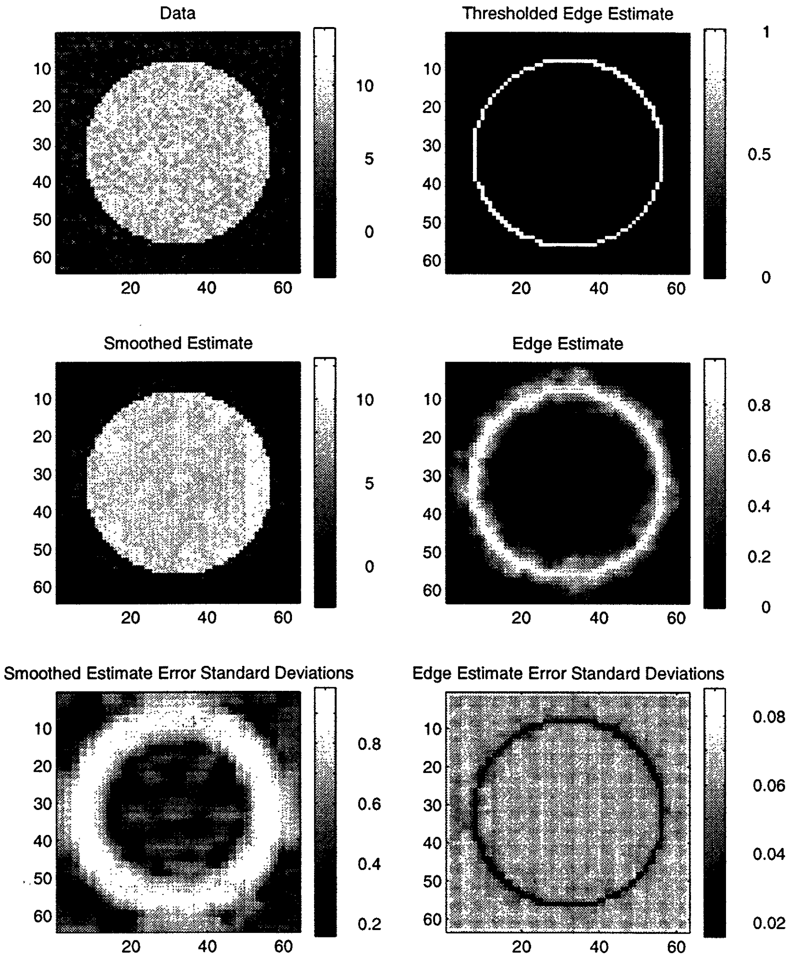
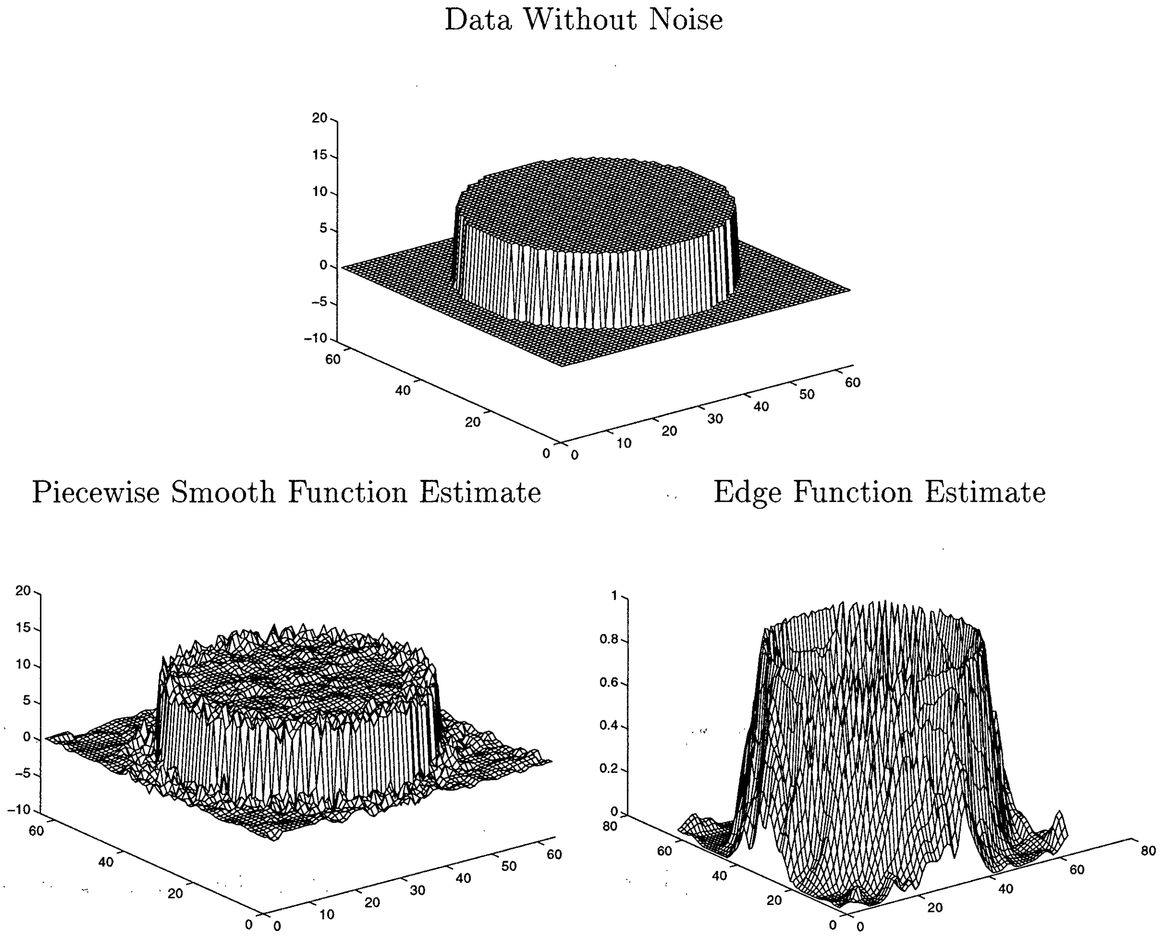


Figure 4-12: A circle segmentation example computed using the multiscale thin plate approach.

A lot of the blockiness in the edge function estimate has disappeared, and there is a very distinct ridge marking the edge location, as indicated by the thresholded edge function estimate. The piecewise smooth function estimate is fairly smooth away from the edge. However, the piecewise smooth function estimate in Figures 4-12 and 4-13 is not as smooth in the vicinity of the edge as the piecewise smooth function estimate appearing in Figures 4-6 and 4-7. In Figure 4-12, the lack of smoothness near the circular edge in the piecewise smooth estimate is made quantitatively precise by noting the presence of a wide ring in which the piecewise smooth estimate error standard deviations are large. Although the piecewise smooth function estimate in Figures 4-12 and 4-13 is not as good at that in Figure 4-6 and 4-7, the main quantity


Data Without Noise

Piecewise Smooth Function Estimate

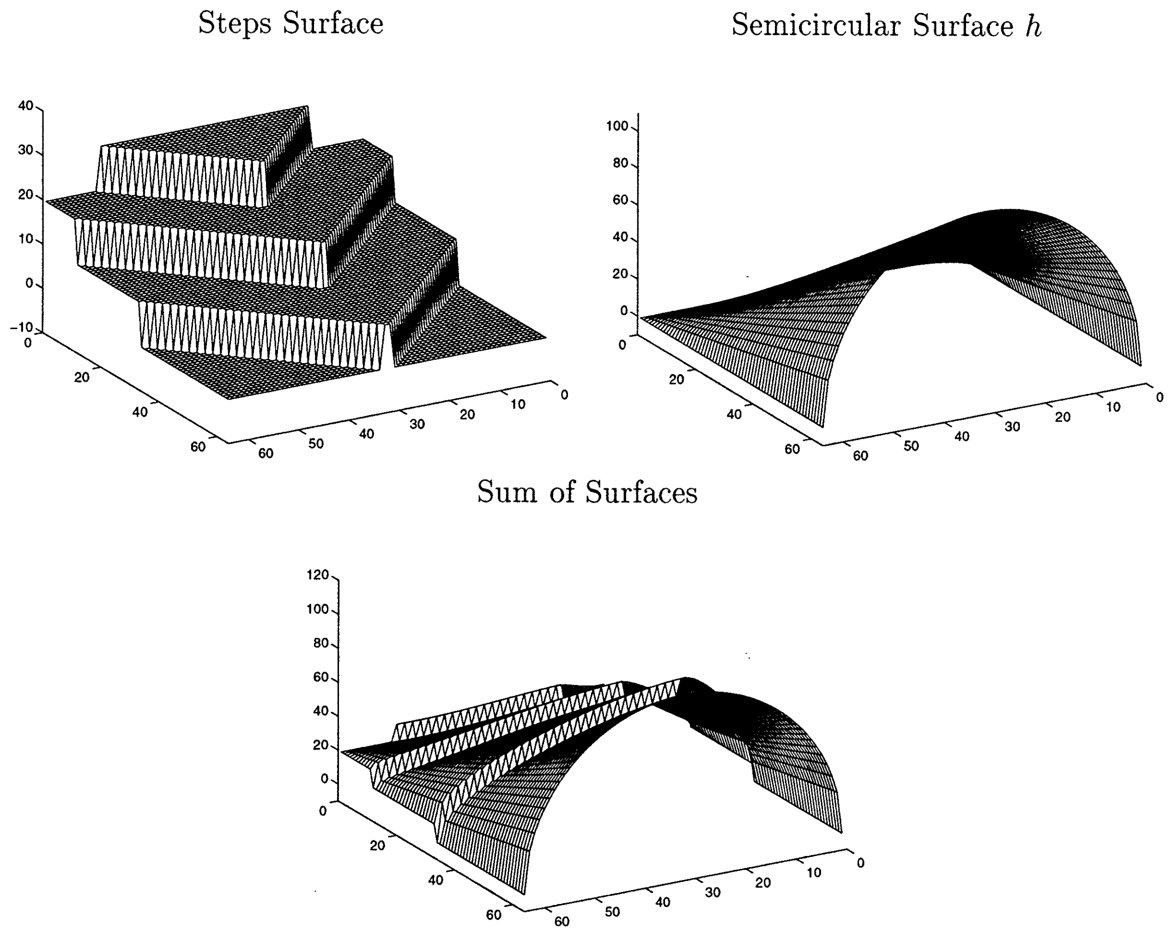
Edge Function Estimate

Figure 4-13: Mesh plots of the circle segmentation example computed using multiscale thin plate models.

of interest is the edge function estimate, and the edge function estimate generated by this multiscale algorithm is very impressive.

The segmentation example in Figure 4-15 is for a different synthetic image that tests how the algorithm responds to gradients in the underlying data. This example is generated using the parameters in Table 4.3 and specifying measurement noise variances of unity. The data in this example is a sum of two surfaces, which are plotted along with the sum in Figure 4-14. The first constituent surface is the steps surface also shown in Figure 2-3. The second is the surface h defined by

$$h_{ij} = 100j\sqrt{1 - \frac{(i - \frac{n-1}{2})^2}{\frac{n-1}{2}}} \tag{4.22}$$


Steps Surface

Semicircular Surface h

Sum of Surfaces

Figure 4-14: Surfaces comprising the data for the example in Figure 4-15.

where h_{ij} denotes the height of the surface at position (i, j) in the image plane $(0, \ldots, n-1) \times (0, \ldots, n-1)$. The function h is semi-circular in profile and thus has very steep gradients near the edges. Hence, one expects that the edge function estimate will tend towards one at the boundary of the image domain. One observes this in Figure 4-15. The indication is that the algorithm is performing as desired.

An example of how the algorithm performs on a synthetic image with missing measurements is presented in Figure 4-16. This is an important example to consider because there are missing measurements in AVHRR imagery. The dark region in Figure 4-17 corresponds to the portion of the image lacking measurements. The measurement noise variances are specified as unity at all locations where measurements exist. The parameters for this example are the ones listed in Table 4.3. The performance is quite good. The edge function estimate has a distinct ridge at the boundary

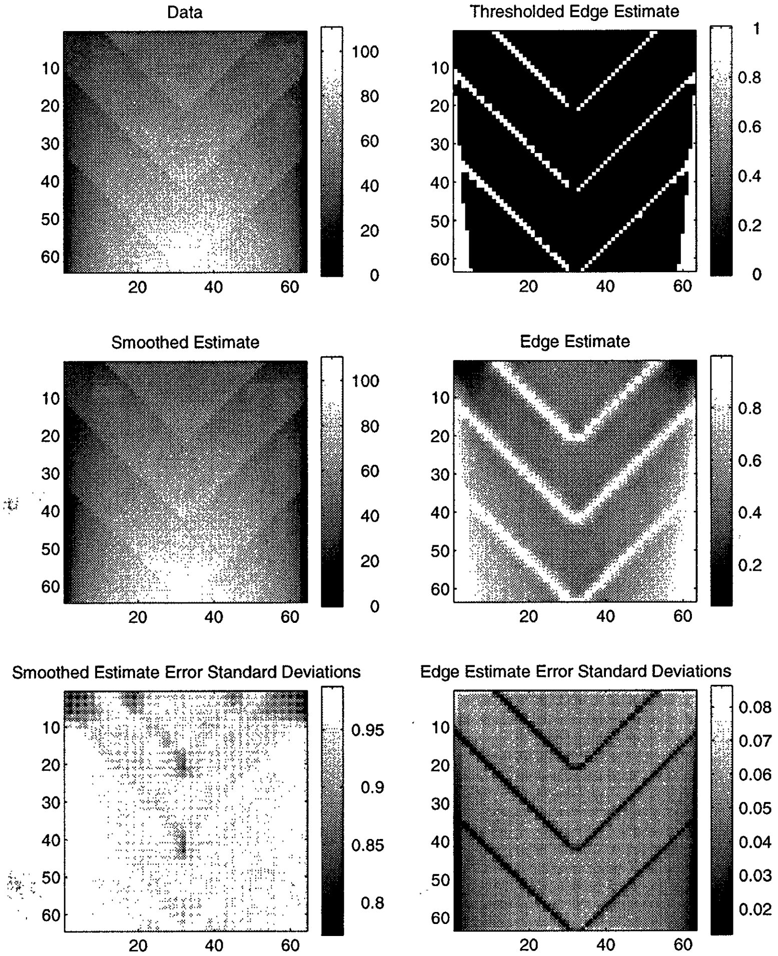


Figure 4-15: Segmentation of a semicircular steps surface using the multiscale thin plate approach.

of the circle where there are measurements, and the piecewise smooth function estimate is a good representation of the circular step, even at locations where there are no measurements. Given the good performance on this synthetic example and the ones in Figures 4-12 and 4-15, one is interested in testing the performance on real AVHRR imagery.

One set of results obtained from using the algorithm on AVHRR imagery is displayed in Figure 4-18. The parameters for this example are listed in Table 4.4. The data is the same as that in Figure 4-9, and the measurement noise variances used by the algorithm are again all unity. Comparing Figures 4-9 and 4-18, one observes that the edge function estimate in the latter has a more distinct ridge at both edges of the

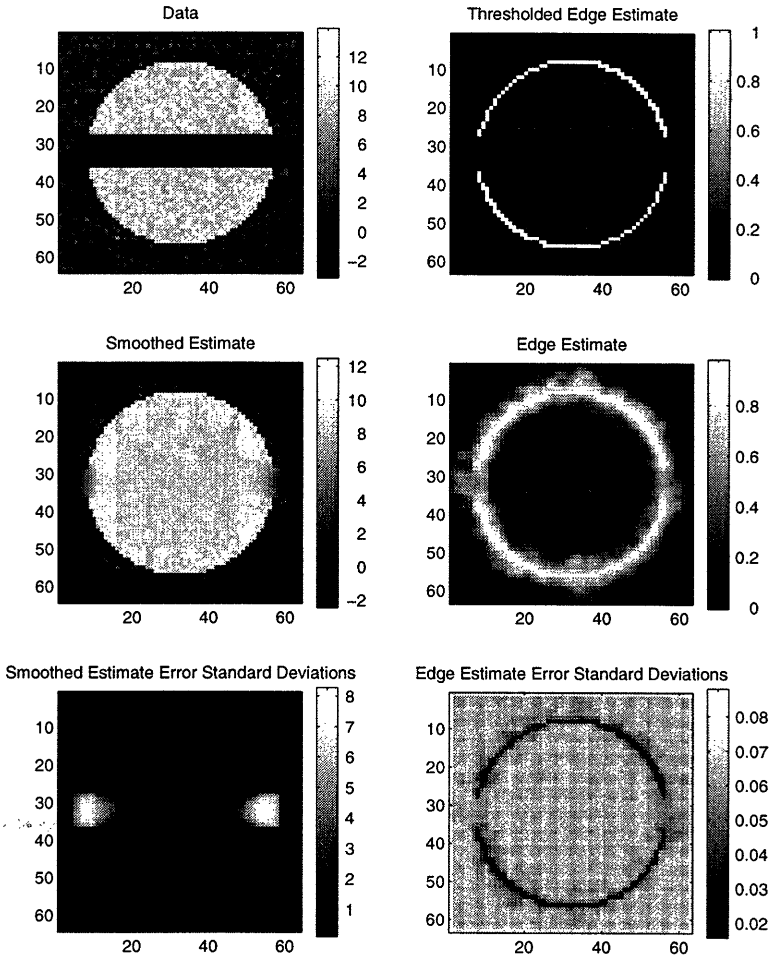


Figure 4-16: A circle segmentation example with data drop outs.

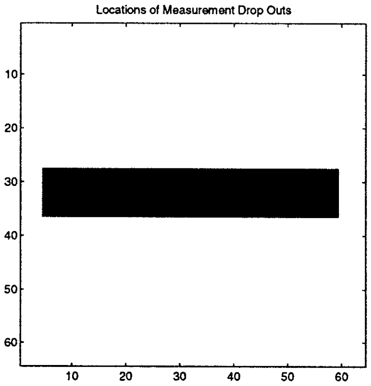


Figure 4-17: Locations of data drop outs for the circle example in Figure 4-16.

Parameter	*Value*
λ	100
b	10
B_s	20
B_g	0.2
R_p	0.35
P_0^f	1×10^6
P_0^s	1×10^6
η	10
τ	0.55
ϵ	1.0×10^{-2}
I	2
T	0.55
$\mathcal{O}$	$\begin{pmatrix} 16 & 10 & 7 & 4 & 2 & 2 & 0 & 0 \end{pmatrix}$

Table 4.4: Parameter values for the AVHRR segmentation in Figures 4-18.

Parameter	*Value*
λ	100
b	10
B_s	20
B_g	0.2
R_p	0.35
P_0^f	1×10^6
P_0^s	1×10^6
η	10
τ	0.55
ϵ	1.0×10^{-2}
I	2
T	0.5
$\mathcal{O}$	$\begin{pmatrix} 16 & 10 & 7 & 4 & 2 & 2 & 0 & 0 \end{pmatrix}$

Table 4.5: Parameter values for the AVHRR segmentation in Figure 4-19.

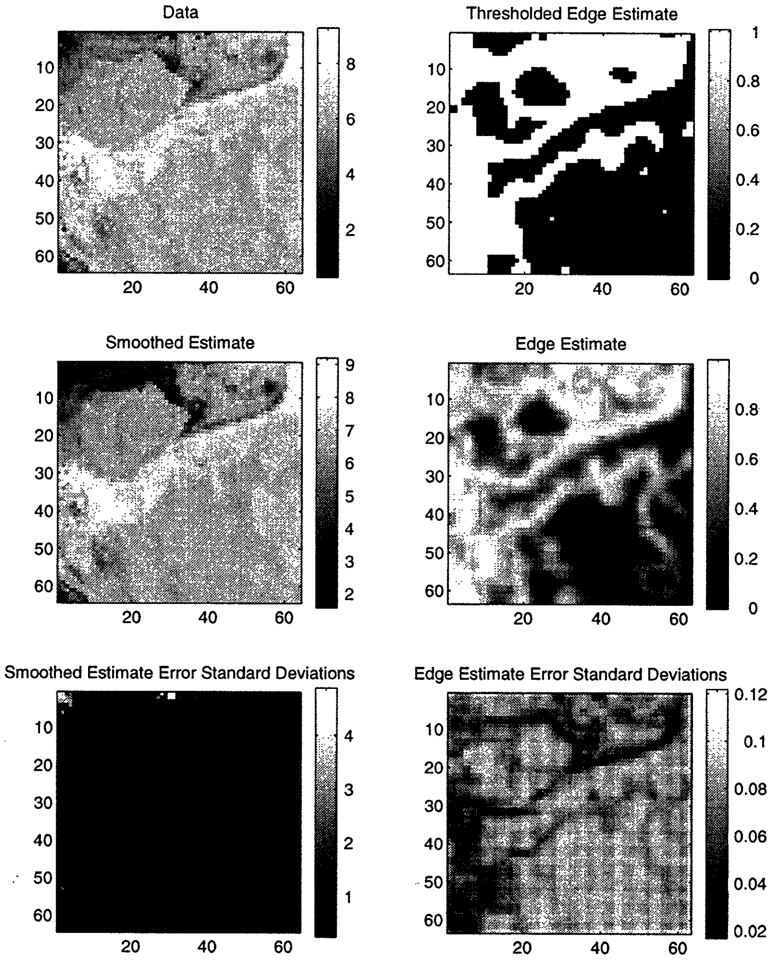


Figure 4-18: Segmentation of AVHRR data using the multiscale thin plate approach.

gulf stream. In addition, the edge function estimate takes on smaller values over the Gulf Stream region in Figure 4-18. Thus, the edge function estimate in Figure 4-18 is more desirable than the one in Figure 4-9. On the other hand, the piecewise smooth function estimate in Figure 4-9 is smoother than the one in Figure 4-18. However, it is more important that the Gulf Stream be clearly delineated by the edge function estimate, and this is the case in Figure 4-18.

Recall that one not only desires to locate the boundaries of the Gulf Stream but also the boundaries of any warm or cold core rings that might be present in the data. A segmentation of AVHRR imagery containing a warm core ring is presented in Figure 4-19, for the parameters listed in Table 4.5 and a specified measurement noise variance of unity everywhere. Locations of data drop outs are marked in Figure 4-

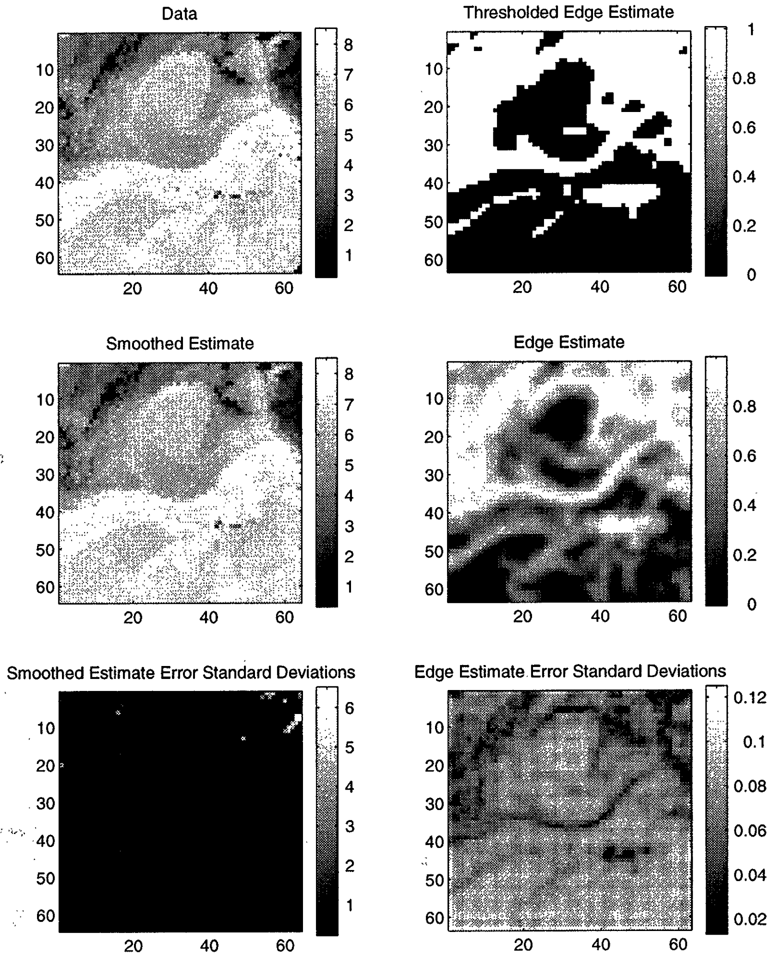


Figure 4-19: Segmentation of AVHRR data containing a warm core ring.

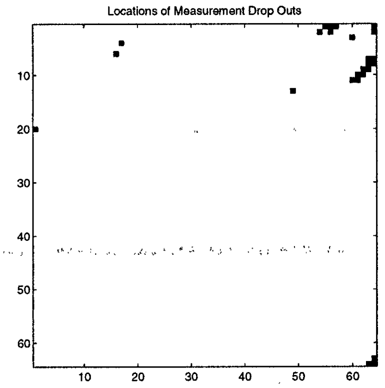


Figure 4-20: Locations of data drop outs for the AVHRR data example in Figure 4-19.

20. If one examines the data, one observes a warm core ring directly to the north of the Gulf Stream. In Figure 4-19, the edge function estimate delineates both the north and south wall of the Gulf Stream. It also marks the northern and part of the southern boundary of the warm core ring well, but the remainder of the ring boundary is marked only by very low edge function estimate values. However, one should note that the south edge of the ring is much less sharply delineated than the north edge. This fact makes the determination of the south edge of the ring very difficult. Thus, the performance attained by the algorithm is quite good.

4.2.3 Analysis

One of the main advantages of the structure of the thin plate algorithm compared to that of the $1/f$-like algorithm is that the modularity of the former allows for simple high level interpretations of the components that can guide improvements. In the case of the segmentation algorithm using $1/f$-like models, the results generated at each iteration by the multiscale estimation algorithm are difficult to interpret in the context of segmentation because the estimates remain in the lifted domain. However, each of the components of the thin plate segmentation algorithm has a simple interpretation in the context of Shah's general approach to segmentation introduced in Chapter 2. Consequently, it should be possible to improve the performance of each piece separately so as to improve the performance of the whole algorithm. In particular, one should be able to incorporate changes to increase the quality of the piecewise smooth function estimate and decrease the thickness of the ridge delineating the borders of the Gulf Stream in the AVHRR imagery.

One possible improvement involves making a small change in the model for estimating the piecewise smooth function given the continuously varying edge function $\tilde{s}$. Recall that the variance of the process noise increases by the additional amount η at locations where the line connecting the parent to the descendent node crosses a discontinuity as diagrammed in Figure 4-11. However, in light of the fact that the edge function is continuously varying, it is more appropriate to allow a varying amount of process noise. In particular, one could use the sum of $\frac{1}{(1-\tilde{s}_{ij})^2\lambda}$ across the line connecting the parent to the descendent node to multiplicatively increase the process noise. This is motivated from the form of the one-dimensional statistical models. One still needs to form a binary edge map by thresholding $\tilde{s}$ because the algorithm must make a fundamental change to the measurement equation at edge locations so as to prevent the propagation of gradient information in the state variable across edges. Details of the change to the measurement equation are given along with other specifics of the multiscale thin plate model in Appendix A. Simply allowing the increase in process noise to vary continuously, however, should yield piecewise smooth function estimates which are more smooth in the vicinity of an edge.

To decrease the thickness of the ridges in the edge function estimates, one can consider adjusting the parameters in the model for s. Recall that there are three parameters in this model: B_s, B_g, and R_p. The values of the parameters adjust the relative penalties placed on first and second order derivatives of s in the model. After adjusting the parameters to reduce the penalty on second order derivatives, one should obtain edge function estimates which better localize the edges in the original image.

4.3 Conclusion

A $1/f$-like and a thin plate multiscale segmentation algorithm are derived in this chapter. Both use as a starting point the discussion in the previous chapter of the one-dimensional statistical variant of Shah's variational segmentation algorithm. The $1/f$-like algorithm displays moderate performance on synthetic images but poor performance on real AVHRR imagery. However, there are possibilities for improving the results by considering other methods for interpreting the multiscale edge function estimate. The results obtained with the thin plate algorithm are better overall, and its modular structure makes it simpler to improve upon. This motivates a future refinement of the thin plate algorithm so as to achieve even better results.

Chapter 5

Conclusions and Extensions

5.1 Brief Summary

The objectives of this thesis are set out in the introduction: to develop an image segmentation algorithm which has constant computational complexity per pixel and is capable of generating error statistics. A variational formulation of the segmentation problem due to Shah [20] is used as a starting point. Shah's approach to segmentation involves coordinate descent of a nonlinear functional. Each of the minimization problems associated with the coordinate descent is amenable to a statistical interpretation. The advantage of casting the problems into a statistical framework is that this provides the proper context for discussing error statistics.

In one dimension, a precise statistical interpretation of the minimization problems is developed. This leads to the creation of a segmentation algorithm in one dimension in which one alternates between using a recursive estimation algorithm to compute piecewise smooth and edge estimates and associated error variances. A set of typical examples demonstrates that the estimates provide one with a good segmentation. A sequence of Monte Carlo simulations indicates further the good quality of the estimates. The experiments also characterize the error variances generated by this iterative segmentation process and establish a strong relationship between them and the actual errors in the estimates. The results are very good and motivate an extension to two dimensions.

In two dimensions, segmentation algorithms are considered whose structure involves alternately forming piecewise smooth and edge estimates and associated error variances. It is noted that standard discretizations of Shah's variational approach to image segmentation are associated with Gaussian Markov random field estimation problems. For such problems, there exists a lower bound on computing estimates and error variances with a recursive estimation algorithm [10]. In order to develop an image segmentation algorithm with constant computational complexity per pixel, this lower bound is circumvented by replacing the Markov random field prior model in the estimation problem with a multiscale prior. This is done in two different ways, yielding two different multiscale segmentation algorithms. One makes use of $1/f$-like multiscale models, and the other thin-plate multiscale models. The performance of the algorithms is characterized by segmenting a variety of synthetic images as well as AVHRR imagery of the Gulf Stream. Both algorithms segment well, but the one incorporating thin-plate models generally performs better.

5.2 Extensions

One major extension of this work involves the investigation of other multiscale prior models for the piecewise smooth process f and edge process s. Obtaining a multiscale model for the piecewise smooth process f is difficult because the model must incorporate information from a previous edge estimate. To obtain a model for f, one could conceivably use the recently developed modeling framework of Irving [12]. He has developed a set of algorithms for generating a multiscale model whose statistics approximate those of a given arbitrary Gaussian random field [12]. Unfortunately, the current process for computing the approximate models is intensive. Thus, it would not be appropriate to use Irving's algorithms to compute an approximate model for the piecewise smooth process f at each iteration in a segmentation algorithm. For the special structure of the segmentation problem, however, one may be able to find a simpler technique for generating approximate multiscale models. This may result in a multiscale model for f that leads to piecewise smooth estimates which are less noisy in the vicinity of an edge. Generating a multiscale model for the edge process s is not as difficult because the model does not change at each iteration. In this case, one could apply Irving's algorithms to obtain a multiscale model which approximates the random field associated with Shah's variational problem for finding s. The resulting multiscale model for the edge process will probably lead to edge estimates which are better able to localize the edges in an image.

Another possibility for improving the quality of the edge and piecewise smooth estimates would be to use a Markov random field prior for both the piecewise smooth and edge processes and to develop a different algorithm to compute estimates and error variances. Recall that the lower bound on the computational complexity applies only for computing estimates of Markov random fields using a recursive estimation algorithm. This bound is circumvented in Chapter 4 by using multiscale models instead of Markov random fields. One could also consider using Markov random field priors and an estimation algorithm which computes approximations of the optimal estimates and error variances. An approximate algorithm may be able to find near-optimal estimates efficiently since such an algorithm is not subject to the computational lower bound. Some steps towards developing such an estimation algorithm have already been started. This work is closely related to that of Chin [3] and looks promising, especially when applied to MRI imagery of the brain.

Both avenues of research will be explored in the future and are expected to yield even better segmentation algorithms which are both efficient and capable of generating error statistics.

Appendix A

Multiscale Thin Plate Model

The segmentation algorithm derived in Section 4.2 makes use of Fieguth's multiscale thin plate model for surface estimation given a binary edge map. The principal details of this model are presented here, and more details are available in [6]. First, the standard form of the root-to-leaf recursion and observation equations are laid out. Then, the discussion turns to the modifications of these equations that occurs at discontinuity locations.

The state vector in the multiscale recursive model is a four dimensional vector $(z_\nu \quad p_\nu \quad q_\nu \quad zp_\nu)^T$. The first component z_ν corresponds to the the height of the surface. The second and third components p_ν and q_ν correspond to first partial derivatives of the surface. The fourth component zp_ν is the value of z_ν at the parent node. The recursion for the model is given by

$$\begin{pmatrix} z_\nu \\ p_\nu \\ q_\nu \\ zp_\nu \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} z_{\nu\bar{\gamma}} \\ p_{\nu\bar{\gamma}} \\ q_{\nu\bar{\gamma}} \\ zp_{\nu\bar{\gamma}} \end{pmatrix} + \begin{pmatrix} B_s 2^{-m(\nu)/2} & 0 & 0 \\ 0 & B_g 2^{-m(\nu)/2} & 0 \\ 0 & 0 & B_g 2^{-m(\nu)/2} \\ 0 & 0 & 0 \end{pmatrix} w_\nu \tag{A.1}$$

where $m(\nu)$ is the scale of node ν, and the w_ν are independent unit variance Gaussian random variables. The recursion essentially specifies a $1/f$-like model for each of z, p, and q.

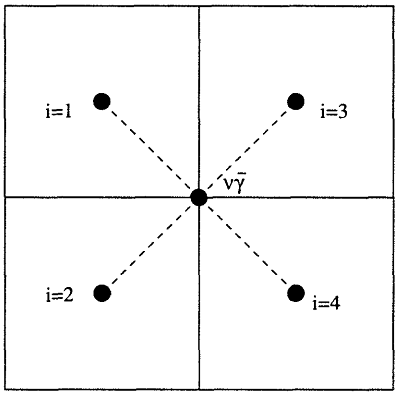


Figure A-1: How nodes are numbered in the thin plate model.

Observations are of the form

$$\begin{pmatrix} h_\nu \\ 0 \end{pmatrix} = \begin{pmatrix} C_\nu & 0 & 0 & 0 \\ -2^{m(\nu)-M} & \alpha_i & \beta_i & 2^{m(\nu)-M} \end{pmatrix} \begin{pmatrix} z_\nu \\ p_\nu \\ q_\nu \\ zp_\nu \end{pmatrix} + v_\nu \tag{A.2}$$

where the v_ν are independent Gaussian random variables, C_ν is 1 at the finest scale and 0 elsewhere, and h_ν is a measurement of the surface. The index i is determined by the location of the node ν in relation to its parent $\nu\bar{\gamma}$ as indicated in Figure A-1. For $i \in \{1, 2, 3, 4\}$, the values of α_i and β_i are given by:

$$\begin{array}{cccc} \alpha_1 = -1 & \alpha_2 = -1 & \alpha_3 = 1 & \alpha_4 = 1 \\ \beta_1 = 1 & \beta_2 = -1 & \beta_3 = 1 & \beta_4 = -1 \end{array} \tag{A.3}$$

The second row in (A.2) is actually a component of the prior model that has been recast as an observation so as to avoid necessitating an increase in the state dimension. This component of the observation acts to ensure that the gradient terms p and q are close to approximations of the gradient of the surface z.

At locations where a discontinuity has been determined as diagrammed in Figure 4-11, the process noise in the surface term is increased by η, and the model is modified so that no information about the gradient terms p and q is propagated across the edge. The recursive model at these locations is given by

$$\begin{pmatrix} z_\nu \\ p_\nu \\ q_\nu \\ zp_\nu \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} z_{\nu\bar{\gamma}} \\ p_{\nu\bar{\gamma}} \\ q_{\nu\bar{\gamma}} \\ zp_{\nu\bar{\gamma}} \end{pmatrix} + \begin{pmatrix} (B_s^2 2^{-m(\nu)/2} + \eta)^{\frac{1}{2}} & 0 & 0 \\ 0 & (P_0^z)^{\frac{1}{2}} & 0 \\ 0 & 0 & (P_0^z)^{\frac{1}{2}} \\ 0 & 0 & 0 \end{pmatrix} w_\nu \tag{A.4}$$

where P_0^z is the prior variance of the root node (generally quite large). The observation equation at edge locations becomes

$$h_\nu = \begin{pmatrix} C_\nu & 0 & 0 & 0 \end{pmatrix}^T + v_\nu. \tag{A.5}$$

Thus, information contained in the gradient terms p and q is not propagated across discontinuities.

Bibliography

[1] L. Ambrosio and V.M. Tortorelli. Approximation of functionals depending on jumps by elliptic functionals via Γ-convergence. *Comm. Pure and Appl. Math*, 43(8), December 1990.

[2] L. Ambrosio and V.M. Tortorelli. On the approximation of free discontinuity problems. *Bollettino Della Unione Matematica Italiana*, 6-B:105–123, 1992.

[3] T. M. Chin. *Dynamical Estimation in Computational Vision.* PhD thesis, MIT, October 1991.

[4] K. Chou. *A Stochastic Modeling Approach to Multiscale Signal Processing.* PhD thesis, MIT, May 1991.

[5] R. Courant and D. Hilbert. *Methods of Mathematical Physics*, volume 1. Interscience Publishers, New York, 1953.

[6] P. Fieguth. *Application of Multiscale Estimation to Large Scale Multidimensional Imaging and Remote Sensing Problems.* PhD thesis, MIT, June 1995.

[7] P. Fieguth, W. Irving, and A. Willsky. Multiresolution model development for overlapping trees via canonical correlation analysis. In *Proc. IEEE International Conference on Image Processing.* IEEE, 1995.

[8] P. Fieguth, W. Karl, A. Willsky, and C. Wunsch. Multiresolution optimal interpolation and statistical analysis of Topex/Poseidon satellite altimetry. Technical Report LIDS-P-2271, MIT Laboratory for Information and Decision Systems, October 1994.

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

[10] A. George and J. W. Liu. *Computer Solution of Large Sparse Positive Definite Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1981.

[11] S. L. Horowitz and T. Pavlidis. Picture segmentation by a tree traversal algorithm. *Journal of the Association of Computing Machinery*, 23(2):368–388, 1976.

[12] W. Irving. *Multiscale Stochastic Realization and Model Identification with Applications to Large-Scale Estimation Problems.* PhD thesis, MIT, September 1995.

[13] N. P. Judish. Polygonal random fields and the reconstruction of piecewise continuous functions. Master's thesis, MIT, May 1993.

[14] S. Krishnamurthy, S. S. Iyengar, R. J. Holyer, and M. Lybanon. Histogram-based morphological edge detector. *IEEE Transactions on Geoscience and Remote Sensing*, 32(4):759–767, 1994.

[15] M. Luettgen. *Image Processing with Multiscale Stochastic Models.* PhD thesis, MIT, May 1993.

[16] M. Luettgen, W.C. Karl, and A.S. Willsky. Efficient multiscale regularization with applications to the computation of optical flow. Technical Report LIDS-P-2175, Laboratory of Information and Decision Systems, 1993.

[17] J. Morel and S. Solimini. *Variational Methods in Image Segmentation.* Birkhäuser, Boston, 1995.

[18] D. Mumford and J. Shah. Boundary detection by minimizing functionals, I. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, 1985.

[19] H. Pien and J. Gauch. Variational segmentation of multi-channel MRI images. In *Proc. IEEE International Conference on Image Processing*. IEEE, November 1994.

[20] J. Shah. Segmentation by nonlinear diffusion, II. In *Proc. IEEE Computer Vision and Pattern Recognition Conference*. IEEE, 1992.