Approximate Spatial Layout Processing in the Visual System: Modeling Texture-Based Segmentation and Shape Estimation

by

Michael Hucka

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in The University of Michigan
1998

Doctoral Committee:

Professor Stephen Kaplan, Chair
Professor Charles K. Butter
Professor John E. Laird
Assistant Professor Sang W. Lee

Dedicated to the memory of my father,
Vladimir Joseph Hucka
(1925–1996).

Acknowledgements

First and foremost, I would like to thank my wife, Renee Rottner, for her understanding, patience, love and encouragement throughout my many years as a graduate student. I often wonder how she put up with me and my ways, and I can only hope that I will have been worth it once I have completed this degree and can become a real person again. I also thank my mother and late father, especially for supporting me in my interests no matter where they led. I can't imagine having more patient and loving parents.

I thank Stephen Kaplan for his encouragement, wisdom, help and guidance. Steve taught me entirely new ways of thinking and looking at the world, and I consider myself very fortunate to have been his student. It has also been a tremendous pleasure to be in Steve's research group, and I thank all of the members of SESAME and ANT for many discussions and feedback on different aspects of this work.

I also thank John Laird for many years of support and guidance when I started out as a graduate student, and his continued help and friendship even after I abandoned Soar research to pursue topics in neural computation. The background and approach in this dissertation are quite different from the context in which John normally works. I sincerely appreciate the effort he was willing to put into taking on and working within the perspective of this research, to respond to this work within the frame of reference posed here.

My thanks also go to my other committee members, Charles Butter and Sang Lee, who probably did not anticipate having to deal with a dense, 400 page dissertation when I first approached them a few years ago. I'm very grateful for their efforts and their feedback.

I also want to thank the following people for assistance with many technical issues: Dave Goodmanson and Louis Lauzon, for their detailed explanations about the proper application of the Fourier transform; David Heeger, for help in understanding the contrast-normalization model of cortical cells; Sang Lee, for explanations of many matters in computer vision; Ko Sakai, for help in understanding his average peak frequency model of shape-from-texture; and Eero Simoncelli, for generously making available his Matlab code for convolution and Gaussian pyramids and for answering my many questions about

various related matters. At many times, I would not have been able to make progress without their generous help.

I am indebted to many other people who have been willing to discuss various aspects of this project early in its inception. I especially thank Patti Reuter-Lorenz for discussions about the global precedence phenomenon and visual attention; K. P. Unnikrishnan and Jun Zhang for discussions about neural computation; and Dick Young and Ron Lesperance for discussions about the operation of cortical visual cells and the Gaussian derivative model.

Many other people have generously spent time reading and commenting on various drafts of this dissertation. In addition to my committee members, I want to thank in particular Sandy Bartlett, Dan Berwick, Ron Chong, Clare Bates Congdon, Karen Coulter, Eric Ivancich, Randy Jones, Karl Kluge, Jay Obermark, and Doug Pearson.

I thank Randy Frank, who encouraged me to come to the University of Michigan, and also arranged financial support for me, working as a computer systems programmer at CAEN, for a good deal of time when I first started out. Don Winsor later hired me part-time to work for DCO, for which I am very grateful. It has been a pleasure working for both organizations, and I am glad to have had the opportunity to meet and work with so many wonderful people.

I also thank the past and present staff of the Artificial Intelligence Laboratory at the University of Michigan: Karen Alexa, Clint Bidlack, Dolores Bolsenga, Kathy DeWitt, Al Dobryden, Rob Giles, Jeff Kopmanis, Sara Staebler, Deb Stark, and Cindy Watts. They put up with a lot from me over the years, and have always been kind and generous with their time and help.

Last, but not least, I want to thank the many other members of the AI Lab. Its many current and former members have made it a friendly place to work and a great environment in which to do research. Its facilities have always been outstanding, thanks to the leaderships of its past and present directors Ramesh Jain and John Laird. And it is where I have met many friends that have made my days as a graduate student unforgettable: Steve Anastasi, Dave Burna, Ron Chong, Eric Chown, C. T. Clark, Clare Congdon, Arie Covrigaru, Dan Damouth, Joe Dionese, Randy Jones, Patrick Kenny, Marc Huber, Dave Kortenkamp, Dave Messink, Craig Miller, Dave Musliner, Doug Pearson, Joe Philips, Jim Pierce, Suzi Roberts, Yuval Roth, Neal Rothleder, Peter Wiemer-Hastings, and Bob Wray.

Table of Contents

Appendices

List of Tables

Table

List of Figures

Figure

List of Appendices

Appendix

Chapter 1

Introduction

Moving through the environment, grasping objects, orienting oneself, and countless other tasks all require information about spatial organization. This in turn requires determining where surfaces, objects and other components of a setting are located and how they are arranged. Humans and other natural agents can extract spatial organization from vision rapidly and automatically, which suggests it is a fundamental capability. Yet the nature of this spatial layout information and the ways by which the visual system obtains it remain poorly understood issues in vision research.

To better understand this capability, it would be useful to know how the visual system can make an *initial estimate* of the spatial layout. Without time or opportunity for a more careful analysis, a rough estimate may be all that the system can extract. Nevertheless, such rough spatial information may be sufficient for many purposes, even if it is devoid of details that are important for object recognition. It could also serve as a stepping-stone to more extensive analysis in less constrained situations, giving other mechanisms of visual perception a starting point for making sense of a scene.

The human visual system uses many sources of information for estimating layout. Here I focus on one source in particular: visual texture. I present a biologically-motivated model of how the system can exploit patterns of texture for performing two basic tasks in spatial layout processing: *locating possible surfaces* in the visual input, and *estimating their approximate shapes*. Separately, these two tasks have been studied extensively in vision research, but they have not previously been examined together in the context of a model grounded in neurophysiology and psychophysics. I show that by integrating segmentation and shape estimation, a system can share information between these processes, allowing the processes to constrain and inform each other as well as save on computations. In support of the model developed here, I describe in detail a software simulation that can perform texture-based segmentation and shape estimation on images containing multiple, curved, textured surfaces.

1.1 Approximate Spatial Layout in Vision

By “spatial layout” of a visual scene, I mean simply the spatial organization of elements in it—where things are and how they are arranged. The spatial organization of a visual scene is one of its most fundamental qualities. Assessing the spatial layout of structures in the scene is one of the most basic tasks facing the visual system.

A full description of a scene’s spatial layout would entail describing the location and other spatial attributes of every feature in it. For many purposes this would be far too much information, and in fact it is extremely unlikely that our visual systems expend the effort necessary to compute such detailed information (Watt, 1995). A description that captures the spatial organization created by the major elements in the scene, but in less detail, can be called an *approximate spatial layout* representation. It is the kind of description that might come out of an artist’s rough sketch of a scene, or a person’s impression after getting the barest glimpse of their surroundings. Figure 1.1 illustrates the idea. The representation provides a rough sense of where different surfaces (or structures that at first glance appear to be surfaces) are located with respect to the line of sight, coupled with other information about them such as their spatial orientations with respect to the viewer, their rough shapes, and their overall sizes in the field of view.

Spatial layout is a central element in many theories of vision. In many ways, the idea pursued here of describing the spatial layout of a visual scene is compatible with

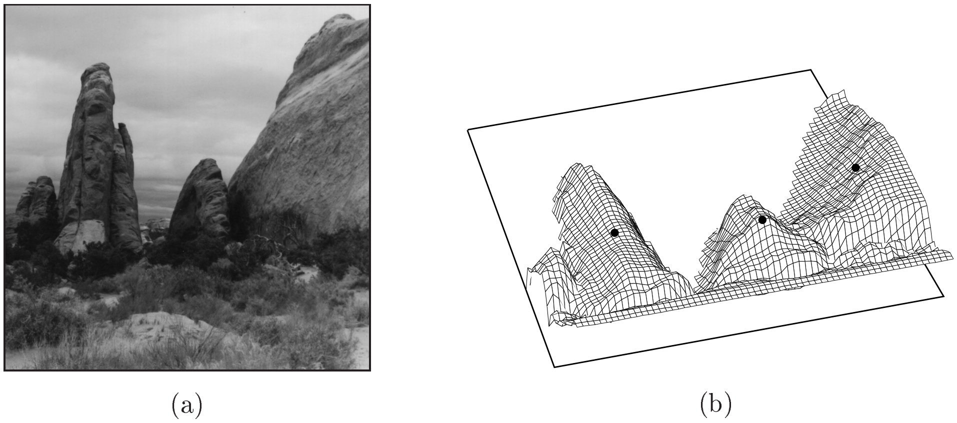


Figure 1.1: (a) An example of a natural scene, and (b) an example of one possible type of description of the scene’s approximate spatial layout. The description provides a rough sketch of the scene’s composition in terms of surfaces seen from the vantage point of the observer. The particular type of representation shown here captures both the rough shapes of the surfaces and shows the spatial orientations of the surfaces at different locations. A description of approximate spatial layout might reasonably include other kinds of information. For example, it might include a summary of the location of each region in terms of its visual center-of-mass, indicated here by the black dots.

Marr and Nishihara's seminal theory of the $2^{1}/_{2}$-D sketch (Marr, 1982; Marr & Nishihara, 1978). Despite the theory's age, the concept of a $2^{1}/_{2}$-D sketch still appears to be broadly consistent with results from current research on the psychology and neuroscience of vision (Bruce, Green, & Georgeson, 1996), although it is now widely believed that various aspects of Marr's overall theory of vision are not tenable (Watt, 1988; Zucker, 1992).

Marr (1982; Marr & Nishihara, 1978) believed that vision serves an adaptive purpose: biological vision systems evolved to inform organisms about the physical structures in their environments by extracting information from available ambient light. In his theory, Marr proposed that one of the intermediate goals of early visual processing is to produce a viewer-centered description of the visible surfaces in the environment prior to more elaborate processing. This *$2^{1}/_{2}$-D sketch* includes the positions, relative depths, orientations, and other information estimated about the surfaces. These are represented in a reference frame centered on the eye, the natural frame of reference for the basic kinds of processes involved in producing the description. All of the various processes that derive information from vision, such as stereopsis, the analysis of motion and contour, and others, are assumed to contribute to the sketch. The name "$2^{1}/_{2}$-D sketch" is meant to emphasize the fact that the full three-dimensional structure of the surfaces is not made explicit, and that the elements of the representation are viewer-centered rather than object-centered. Marr saw this stage as the last step before the roles of different surfaces are interpreted by higher-level processes.

Similarly, the present research is intended to explore how the visual system can exploit one particular type of optical information, visual texture, to derive descriptions of the spatial layout of surfaces in the environment. The results of this analysis might reasonably be conceived as one component of a *qualitative* $2^{1}/_{2}$-D sketch. It is qualitative because the types of information and processes explored here cannot recover the exact structure of a scene; at best, the processes provide an incomplete and approximate description. But although individual, approximate visual mechanisms of the sort investigated in this research are only weakly useful on their own, collecting them into a system can allow their individual results to reinforce each other and lead to an effective summary of a visual input.

There are other parts to Marr's (1982) theory of vision, but it is not necessary to assume the other elements; the concept of a stage in visual processing that derives sketches of surfaces stands on its own. Indeed, it is an element that is implicit in many other theories of visual processing (e.g., Kaplan & Kaplan, 1982; Nakayama & Shimojo, 1992; Treisman, 1986).

1.1.1 The Utility of an Approximate Spatial Layout Analysis

Approximate, in the context of this research, means that the system may not have detailed information about the structures in the visual input. Approximateness suggests a lack of precision and definition, and is often regarded as undesirable, especially in work on computer vision systems. But approximateness can be useful in many ways.

Natural agents inhabit complex and dynamic environments, where survival requires balancing accuracy against speed. One way to cope with this challenge is to exploit parallelism: multiple subsystems can process the visual input simultaneously, and indeed biological vision systems are known to employ massive parallelism at many different levels. In fact, one of the most significant examples of higher-level parallelism involves object recognition and spatial layout analysis: many neurobiological and psychological studies of the visual systems in humans and animals suggest that object features are processed in areas of the brain different from the areas in which information about space is processed (Kosslyn, Flynn, Amsterdam, & Wang, 1990; Mishkin, Ungerleider, & Macko, 1983; Zeki, 1993). Another, complementary way of coping with speed-accuracy pressures is to avoid performing detailed, exacting analysis until necessary. In other words, rather than routinely computing precise information over the whole visual field and attempting to find a coherent description for all the data simultaneously (which is expensive in terms of computational resources, as well as difficult given the uncertain and sometimes rapidly changing nature of the environment), it is better to let fine-grained processing be guided by initial, cursory processing. This does not mean that the ultimate results of visual processing will necessarily be less accurate or less reliable. Since it is not the final analysis, other stages of processing can later improve upon the initial estimate.

An initial assessment of spatial layout, even if it lacks information about the precise shapes of objects or their identities, nevertheless can be very useful in its own right. For instance, the information can help guide visual orienting movements, so the agent can move its eyes to different targets (Kowler, Anderson, Dosher, & Blaser, 1995). Further, many tasks requiring information about spatial organization can be performed adequately using only approximate layout information. For example, an agent may be able to make preliminary assessments of qualities that can help guide its behavior, judging different settings in terms of their options for movement and action (Kaplan & Kaplan, 1989). Experiments have also shown that even a rough representation of a scene, when presented very briefly to human subjects, is enough to allow scene discrimination (Schyns & Oliva, 1994). Similarly, in the field of computer vision, some models of visual navigation are based on using qualitative, pattern-recognition approaches instead of quantitative scene reconstruction (Nelson, 1988). And finally, without the time or opportunity to have a good look at a scene, rough spatial layout may be all the agent *can* get. Thus, in situations where time and available information are limited, the rough spatial organization of different visible structures and empty spaces may be enough to allow the organism to make at least some decisions.

Approximateness can thus be usefully exploited. It fits with the idea that the system will compute just what it needs, and not more. “The goal of biological visual systems is to rapidly compute approximate solutions to perceptual problems. The solutions are always adequate for the job on hand, but rarely optimal” (Ramachandran, 1990, p. 23).

1.2 Overview of the Research

The types of processing being investigated in this research take place in the early stages of analyzing an input. *Early vision* is a part of the visual system that is not precisely defined in vision research, but is commonly assumed to have the following characteristics (Zucker, 1992). First, it is concerned with general-purpose assumptions about how the physical world is related to the structure of the retinal image. The assumptions must be general-purpose because they must hold for many real-world situations. Second, early vision involves many processes that operate automatically and continuously on the “stuff” that constitutes the visual input. The processes are largely *preattentive*: they attempt to infer various qualities out of the input and reach conclusions on their own, without explicit direction from the controlling influence of selective attention. However, the preattentive aspect is not fixed firmly; higher-level cognitive mechanisms can sometimes influence the processes in a top-down fashion on short time scales, and adaptation can adjust their operation over long time scales.

There is a large body of psychological research demonstrating preattentive processing of many basic attributes such as size, location, orientation, and color (Treisman, 1986; Wolfe & Bennett, 1997). In addition, there is also evidence for preattentive processing of more abstract attributes such as simple three-dimensional surface configurations (Enns & Rensink, 1990, 1991) and configural closure (Donnelly, Humphreys, & Riddoch, 1991).

There are many possible sources of optical information for these analyses. Among the most prominent are stereopsis, motion, texture, color, shading, and contour. In the visual system, all of these sources of information are used together. In this research, I focus exclusively on the role of texture, partly because it can serve double duty for visual segmentation and estimation of surface shape, and partly because there exists a large body of research on the topic. In addition, to make the work manageable, I further limit consideration to static, achromatic (gray-scale) scenes viewed without stereoscopic cues.

1.2.1 The Focus: Two Components of Spatial Layout

The goal of this work is to develop a model of how the visual system may make initial estimates about two particular components of spatial layout based on patterns of visible texture. The specific components are (1) the locations of possible surfaces in the input, and (2) their rough shapes.

Visual texture is a property of image regions containing a large number of repetitive markings, where the exact positions of the markings within the regions are irrelevant. Mechanisms tuned to detecting contrast variations in the visual input can serve as a basis for extracting information about texture. The complex cells of the primary visual cortex have especially useful properties in this regard. Each complex cell is selectively responsive to stimuli containing a particular range of spatial frequencies and orientations. However, unlike a simple cell, a complex cell is largely insensitive to the position of a stimulus within its receptive field; it is thus well-suited to analyzing regional properties of a stimulus.

It is well-known that the visual system can use texture as a basis for segmenting an input into regions. In its simplest form, the result of this kind of processing localizes regions in a retinal frame of reference. As explained in Chapter 2, one of the problems with most current biologically-motivated models of texture-based segmentation is that they are designed for the restricted case of inputs consisting of flat surfaces oriented to face the viewer. These models function by searching the outputs of a collection of complex-cell-like mechanisms for regions of homogeneous responses. However, in natural scenes, surfaces are generally not flat nor oriented facing the viewer. This results in gradients and other systematic variations within regions of texture projected onto the eyes. The presence of these distortions in the image texture means that the texture will not necessarily evoke homogeneous responses in the complex cells. Most current models of texture segmentation will fail in this situation. A more realistic segmentation approach must allow for shape-induced distortions within regions of texture.

In some contexts, *segmentation* refers to the process of isolating an object or parts of an object from the background. This would be synonymous with solving the figure-ground problem, which is not intended in this research. Instead, by segmentation I mean simply the process of locating areas in an image containing prominent elements. An advantage of using texture for this purpose is that, because texture is a surface property, a texture-based segmentation amounts to a first-cut description of the locations of regions likely to contain different surfaces.

Systematic variations in the texture of a region in an image can be a clue to other spatial properties, in particular the rough shape of the surface that gave rise to the region. The same mechanisms that provide a starting point for segregating the input into regions can also provide information allowing other mechanisms to estimate the rough shapes of textured surfaces within each region. As explained in Chapter 2, one of the problems with most current models of texture-based estimation of surface shape is that they assume only one surface is given in the input. A more realistic approach therefore requires augmenting the process of shape estimation with a segmentation process.

The idea of using texture for either segmentation or estimation of surface shape is not new. However, most previous work has examined one or the other process in isolation. As a result, the majority of existing models of these processes have serious limitations that prevent them from functioning on all but the most simplified visual inputs. Moreover, as discussed in the next chapter, most existing models make incompatible assumptions about the nature of the visual input—the assumptions made by the segmentation models are exactly what prevents the shape estimation models from working on the same visual scene, and vice versa (Krumm & Shafer, 1994). And perhaps most surprisingly, little attention has been given to the problem of integrating them into a single visual system, yet biological visual systems perform both texture-based segmentation and shape estimation in an integrated architecture. Investigating how the processes may be combined is therefore an essential step in understanding biological vision.

1.2.2 Overview of the Model

The bulk of this dissertation presents an implementation of a model of integrated, texture-based segmentation and shape estimation. The model is a synthesis and enhancement of several other lines of research. The contributions of this work are discussed in Section 1.3.

There are three main components to the model:

1. *The initial processing component.* The model begins at the level of the simple and complex cells of the primary visual cortex. It treats all the stages of visual processing up to that point as a black box—from input and preprocessing by the retina, to delivery of neural signals to simple cells in the cortex. Complex cells are modeled functionally as taking their inputs from multiple simple cells. Each point in the visual input is analyzed by a collection of these model neurons, and the output of this initial processing component is a set of values representing the strengths of different cell responses at each point in the input. A total of 80 types of complex cell responses is computed at each pixel location of each input image.

   Complex cells are a common starting point in research on models of texture-based visual processing. The current work is no exception, but it is unusual in its greater attention to documented physiological properties of complex cells, as well as its synthesis of two complementary lines of research focusing on the properties of simple and complex cells in the primary visual cortex. These two lines of work are the contrast-normalization model of simple and complex cell function (Heeger, 1991, 1992a, 1992b), and the Gaussian derivative model of simple cell receptive fields (Young, 1985, 1991; Young & Lesperance, 1993). The model developed here is unique in employing a wide range of Gaussian derivatives, from first through sixth order, to model more closely the characteristics of actual cortical neurons.

2. *The surface shape estimation component.* The collection of outputs from the initial processing stage can be used as a starting point for estimating surface shape. The basis for this is that the texture on a physical surface becomes distorted when projected onto the retinas of the eyes. The distortions can be measured in the pattern of responses of cortical cells and related to the three-dimensional shape of the physical surface that produced the image.

   The method of shape estimation used here is an adaptation of a model developed by Sakai and Finkel (1995). Their neural model of texture-based estimation of surface shape is based on relatively simple principles, described in detail in Chapter 4, that fit well with the present emphasis on rough spatial layout analysis.

   The basic approach developed by Sakai and Finkel (1995) shares the limitation of nearly all published shape-from-texture methods in being designed around the assumption of viewing a single surface at a time. To overcome this limitation and develop a model that can cope with inputs containing multiple surfaces, I have combined their approach with a texture-based segmentation scheme. The resulting new model can estimate the rough shape of each textured surface in a visual scene.

3. *The segmentation component.* The output of the initial processing stage (the complex cells) can also serve as a starting point for texture-based segmentation. The basis for this is that different textures give rise to activity in different subsets of the cortical cells, and the resulting patterns of activity can be analyzed to locate different regions of texture in the visual input.

   Most existing models of texture-based segmentation operate by detecting homogeneous regions of image texture; they cannot handle distortions in texture patterns caused by changes in the shape of a surface. The segmentation model used here is different from the most popular class of other models. It expresses the segmentation process as a competition between region interpolation and boundary detection, and is not limited to segmentation based on the homogeneity of complex cell responses. It is based on an approach known as the *coupled-membrane model*, developed by

Lee (1993, 1995; Lee, Mumford, & Yuille, 1991, 1992). The approach can handle gradients and other distortions in texture in an image, an essential quality for being able to work with realistic visual inputs. It is also a type of model that can be implemented in a parallel distributed architecture such as a neural network.

I show in Chapter 5 that Lee's (1993, 1995; Lee et al., 1991, 1992) approach requires only a modest change in order to start from intermediate results produced by the texture-based shape estimation component mentioned above. The intermediate result is indirectly related to a measure of local surface orientation. This change significantly simplifies the segmentation process, thereby making it substantially more efficient and faster than the original coupled-membrane model without seriously jeopardizing the system's ability to produce a reasonable result. In addition, elements of the new segmentation equations can be related more easily to surface shape qualities, giving the model a more vivid conceptual interpretation.

The integration of texture-based segmentation and shape estimation has been investigated in the field of computer vision (Krumm & Shafer, 1994; Moerdler & Kender, 1987). The present work differs from other attempts in that it is motivated by a desire to understand brain mechanisms of visual perception, rather than a desire to solve the problem from a purely engineering perspective. The new, integrated model presented here is unique in combining texture-based segmentation and shape estimation in a biologically-reasonable framework.

1.2.3 Methodology

The particular approach I use in this project might appropriately be called *neurally-inspired computational modeling.* The simulation is not, strictly speaking, a neural network, but it is roughly at the same level of information processing. The simulated complex cells, for example, are implemented as mathematical functions that perform an operation on their inputs and communicate an output in the form of a number. The number signifies the strength of the response of a model cell and is analogous to the firing rate of a real neuron, with a higher value on the output signifying a higher firing rate. This type of simulation is not the equivalent of building a neural network model, but instead is at a level describing the processes and flow of information involved. The result consists of roughly the same number of modules that a comparable neural network model would have, but is more efficient to simulate because individual neurons are not implemented.

Why not simply define the components and describe their intended properties in lieu of implementing a simulation? The reason is that simulations are very useful tools in developing theories. One of their most important features is to help prevent vagueness: implementing a simulation forces one to fill in details that could otherwise be glossed over. For a simulation to work, it is necessary to be explicit about *how* different steps are performed. Of course, this does not assure the correctness of the theory being simulated, but a functioning simulation at least improves the odds that the theory is workable.

In developing the model presented in this dissertation, I have used a combination of three research methodologies: *top-down analysis*, a *small-experiments* approach using systematic exploitation of failure (Kaplan, Sonntag, & Chown, 1991; Weaver, 1993), and *parallel exploration of alternatives.*

Top-down analysis involves mapping out the desired capabilities of a system from the highest, most abstract level down to lower, more detailed levels. I used this approach to outline the general features of the model, such as the need for an initial processing component patterned after cortical complex cells. This methodology makes sense when there is significant prior work from which to draw examples and infer requirements.

I used the small-experiments approach to fill in the details and develop actual working mechanisms. The small-experiments approach is straightforward, yet it is rarely articulated in the literature on modeling perceptual mechanisms (but see Graham, Beck, & Sutter, 1992). The essential ideas behind the approach are the following:

1. Begin with a set of mechanisms that is actually believed to be too simple, in the sense that the simulation is expected to fail. The idea is that one should not put in, right at the outset, every mechanism that is believed to be necessary, but rather to start with a small, clearly delineated subset. Otherwise, one risks having an unmanageable mess that is difficult to study and understand, and even if it works, it will be difficult to determine which mechanisms are necessary and which are superfluous (Weaver, 1993).

2. Experiment with the simulated model and determine how and where the model fails. This is a critical component of the approach; it guides the next modification to the model. By starting with an intentionally oversimplified set of mechanisms, one *expects* the resulting system to fail, and if one did a good job, it will fail exactly as expected—but, more often, it will fail in a surprising manner, and this is what leads to important lessons. If, on the other hand, the system does fail in the expected manner and the addition of the missing mechanism corrects the failure, it will confirm the appropriateness of the theory in a more convincing manner than if all components had been included straight from the beginning.

3. Modify the system to attempt to overcome the failure, but make only one, small, modification. This improves the chances of being able to understand the interactions that result between system components and the contributions of each additional element.

4. Go back to step 2 and iterate, stopping when the simulation achieves the objectives or it becomes clear that the model cannot ever do so.

The small-experiments approach offers several benefits (Weaver, 1993). First, it is easier to begin the simulation work, easier to manage it at each step, and easier to keep it motivated—the whole enterprise is more compelling. Second, the approach leads to results that are easier to communicate. New components are added one step at a time, each new component has a specific reason for having been introduced, and the whole process is divided into discrete experiments that are easier to follow and understand.

In situations where there was no clear next step in the development process, I used parallel exploration of alternatives to search for possible avenues. This approach is made possible in part by the availability of networked computers, which permit rapidly searching a space of alternatives by simultaneously performing multiple small experiments. An example of the application of this method is the search for parameter values for a given mechanism when there are no readily-available constraints on the most appropriate values. By running multiple alternatives on a large number of interconnected computers, it is possible to explore the space of parameter values much more rapidly than if each alternative had to be attempted piecemeal.

1.3 Contributions of the Research

The main contribution of this dissertation is the development of a model that combines texture-based segmentation with texture-based shape estimation. To date, the only other published attempts at integrating these two capabilities have been engineering efforts in the context of computer vision (Krumm & Shafer, 1994; Moerdler & Kender, 1987). By contrast, the primary motivation for the present research is to further our understanding about brain function, in particular about mechanisms of spatial perception. Although the model here is not a neural network model, all of the computational components that it uses are simple enough that they can reasonably be expected to be implementable using biological neural circuits.

In fact, all of the major components of the model are based directly on neurobiological and psychophysical research results. One of my goals throughout this project has been to synthesize results from research in a number of separate subject areas:

- Neurophysiological and computational studies of complex cells in the visual cortices of mammals;
- Psychophysical, computational and neurophysiological studies of texture-based segmentation; and
- Psychophysical, computational and neurophysiological studies of texture-based estimation of surface shape.

In addition to contributing to research on integrating segmentation and shape estimation using texture, this work makes several smaller contributions to a variety of topics:

1. *Models of cortical complex cells.* There are two main contributions to this area. First, creating a simulation of complex-cell-like mechanisms that have properties resembling real neurons requires paying attention to a great many details. Although the particular parameter values chosen here are specific to some of the goals of this research, the design approach that I have followed and described in Chapter 3 should be useful to other modelers developing simulations of complex cells. Second, as mentioned above, the formulation of complex cells used here is a combination of two lines of research: the contrast-normalization model of simple and complex cells and the Gaussian derivative model of simple cell receptive fields. Unlike previous uses of these two models together (e.g. Simoncelli & Heeger, 1997), the model presented in Chapter 3 uses Gaussian derivative receptive fields having many different spatial scales, rather than one scale only. To make the contrast-normalization model work correctly for this case, the receptive field operators must be normalized so that they evenly tile a portion of the spatial-frequency spectrum (an issue that is explained in more detail in Chapter 3). This kind of smooth tiling is technically impossible to do exactly using Gaussian derivatives. However, I have developed a method for doing the tiling in an approximate fashion. This method is described in Section 3.3.2.

2. *Models of texture-based segmentation.* The segmentation approach used in this research is based on a coupled-membrane model developed by Lee (1993, 1995; Lee et al., 1991, 1992). In Chapter 5, I present a modified version of this model that allows it to reuse an intermediate result produced by the shape estimation component. Doing so simplifies the model and simultaneously reduces the number of computations required by the segmentation process. This work thus represents an evolution of the coupled-membrane model of segmentation.

3. *Models of texture-based shape estimation.* The model of texture-based shape estimation used in this work is based on the average peak frequency model of Sakai and Finkel (1995). In Chapter 4, I present a number of enhancements to the model that improve the smoothness of the shape estimates that it produces. I also explore its performance and some of the conditions for its successful application. In Chapter 6, I present modifications to the model that allow it to operate within regions of image texture found by the segmentation component, as well as to detect and label regions lacking texture and regions corresponding to flat surfaces facing the viewer.

Limiting the scope of this project to static, monocular visual information admittedly reduces the generality of the results; after all, biological systems make extensive use of motion information as well as binocular information for spatial layout (at least in creatures that have two frontally oriented eyes). However, treating the full problem of spatial perception, including motion and binocular vision, is well beyond the scope of a single dissertation. Further, the limited scope is reasonable given that the visual system *is* capable of deriving at least some spatial layout information from brief glimpses, in which the environment will appear static, and without using binocular information, which is effective only at limited distances and only within a certain range of the visual field.

1.4 Organization of the Dissertation

This project involves a great deal of synthesis; it brings together evidence, ideas and techniques from numerous disciplines including psychology, visual neuroscience, computational vision, and adaptive systems analysis. In an attempt to make the research understandable to a broad audience, I summarize in Appendix A a number of background concepts about vision that are central to the dissertation. Interested readers may wish to review the material presented in this Appendix prior to jumping into the thick of the dissertation proper.

The following is an outline of the main chapters in this dissertation:

Chapter 2: In this chapter, I review evidence about texture-based visual processing and show that the evidence supports the hypothesis underlying this research. I also review existing models of segmentation and shape estimation, and show that most models of each process have limitations that make them unusable for realistic visual inputs. Finally, I outline the solution to the problem that is pursued in the rest of the dissertation.

Chapter 3: Here, I present a model of the initial processing stage, the simple and complex cells of the primary visual cortex. I also present Experiment One, whose purpose was to test the behavior of the simulated cells in response to images containing textures.

Chapter 4: In this chapter, I present a model of texture-based shape estimation. The model uses as inputs the responses of the simulated complex cells described in Chapter 3. I also describe Experiment Two, designed to test the performance of the shape estimation system.

Chapter 5: In this chapter, I present a model of texture-based segmentation that uses as a starting point a portion of the results produced by the shape estimation component of Chapter 4. I also describe Experiment Three, whose purpose was to test the performance of the segmentation method.

Chapter 6: The segmentation and shape estimation components are brought together in this chapter into a combined framework. I present additional mechanisms that perform tasks such as detecting image regions that are devoid of texture and detecting regions that contain flat surfaces. Experiment Four explores the performance of the resulting system.

Chapter 7: In this chapter, I conclude the dissertation with a summary of the results, the remaining problems, and plans for future research.

Chapter 2

Visual Texture-Based Segmentation and Surface Shape Estimation: Review of Previous Work and a Theory of Integrated Processing

This chapter presents a background review of empirical and theoretical evidence concerning several topics at the heart of this dissertation. In particular, I examine the utility of visual texture for both segmentation and the estimation of surface shape, and review research on both subjects in order to evaluate existing models of the processes involved. Evidence and models developed from the perspectives of psychology, neuroscience, and computer vision are all treated together, although my primary focus is on models developed with a goal of biological reasonableness.

A close examination of popular models of each process (segmentation and shape estimation) shows that each has limitations that prevent its use on visual inputs containing multiple regions of curved or slanted textured surfaces. I discuss existing attempts at overcoming these limitations, and outline a new approach that I pursue in the rest of the dissertation.

This chapter is organized as follows. Section 2.1 discusses contemporary views about the roles of surfaces and textures in vision. Section 2.2 then examines texture-based segmentation and highlights an important limitation of the most popular class of biologically-motivated segmentation models. Section 2.3 examines texture-based estimation of surface shape and highlights an important limitation in the majority of shape estimation models. The following section examines the challenge of overcoming the limitations in each type of model and combining the two texture-based processes into a single framework. It also outlines a proposed solution that synthesizes elements of existing models in a new way. Finally, Section 2.5 summarizes the main points of this chapter.

2.1 Surfaces and Texture

The environment can be conceived as a layout of visible surfaces (Gibson, 1950b; Marr, 1982; Sedgwick, 1983), and one of the fundamental goals of the visual system is to interpret the patterns of light entering the eyes as providing information about surfaces in the world. Contemporary research on human vision supports the view that "perception of surfaces is an an autonomous process, minimally subject to object-specific knowledge about the world" (Nakayama & Shimojo, 1996, p. 1362), and representations of surfaces and the spatial configurations of surfaces have been implicated as an essential element in a variety of preattentive visual processes (Enns & Rensink, 1990, 1991; He & Nakayama, 1992, 1994a, 1994b; Kanizsa, 1976; Mattingley, Davis, & Driver, 1997; Ramachandran, 1988, 1990; Rossi, Rittenhouse, & Paradiso, 1996; Vincent & Regan, 1997; Verghese & Stone, 1997; Watt, 1991a).

As mentioned in Chapter 1, there are many ways by which surfaces may be distinguished visually, including: color, contour, shading, stereo disparity, texture, motion, and others. Texture is an especially versatile source of information. It has also been the subject of a great deal of research in psychology, computer vision, and neuroscience, which provides a rich body of work from which to draw insights and ideas.

A precise definition of texture has been difficult to formulate, even in computer vision (Nalwa, 1993), but the following qualities are generally agreed upon (Bhushan, Rao, & Lohse, 1997; Braunstein, 1976; Nalwa, 1993; Pickett, 1970; Tamura, Mori, & Yamawaki, 1978; Watt, 1992, 1995). *Texture* refers to patterns of visible markings across a physical surface or within an image region. It arises over regions that are much larger than the basic pattern of elements composing it. The features that comprise the elements of a surface texture are small, locally homogeneous regions surrounded by variations in dimensions such as reflectance, orientation, color, and/or shape. For example, the primary textural features of gravel or a rocky slope are roundish blobs of similar size and reflectance relative to other elements in the scene [Figure 2.1(a) on the following page], whereas the primary features of a stand of trees or grass in a field are thin lines oriented in roughly the same direction [Figure 2.1(b)]. In all cases there is a large number of instances of some basic element. Further, natural textural patterns are irregular; it is their statistical structure that is important, rather than individual, local details of the textures. A region of texture is characterized by some degree of statistical similarity between the individual features in every part of that region. Two regions will usually be treated as distinct if the variations within their respective patterns are less than the variations between the two patterns (Watt, 1995).

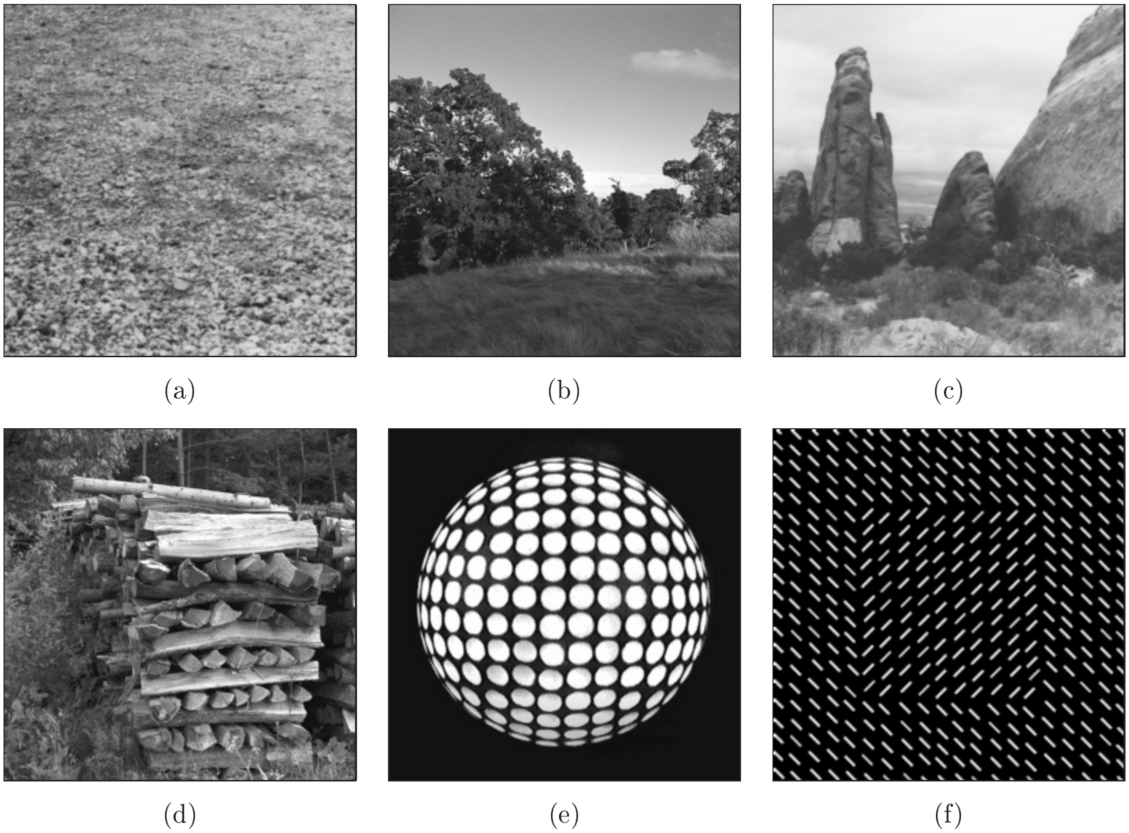


Figure 2.1: Examples of textures. (a) Gravel on the ground. (b) A grassy plain with trees. (c) A rocky scene. (d) A stack of logs in the forest. Note how the collection of logs gives the impression of a surface, even though there is no actual surface bounding the stack. (e) A golf ball with its outer surface carefully painted black. (f) An artificial texture patch consisting of oriented line elements centered inside a field of orthogonally-oriented line elements. This last image is an example of a type of stimulus used in many studies of visual segregation of textures, discussed in the text. Sources: (b) VisTeX database (Picard et al., 1995); (d) PRIP Image Database (PRIP, 1997).

Texture is all around us: all physical surfaces are textured, some more roughly and visibly than others. Moreover, a collection of independent objects can also give the appearance of a textured region and form an implicit surface. An example is shown in Figure 2.1(d), where a collection of split logs gives the impression of a vertical surface. Another example is treating the leaves of a deciduous tree or a bush as forming a roughly globular surface (Watt, 1992).

Visual texture information can serve many functions. Different surfaces in a scene typically can be located, discriminated, and segregated based on differences in their textures. Spatial qualities such as surface orientation and local shape often can be estimated based on visible texture, as can relative distances between different points on the surface and relative sizes between different objects touching the textured surface. Surface qualities such as expected friction can also be judged and often the identity of the material composing the surface can be determined based on its texture (Yuille & Ullman, 1990), although even when textures cannot be identified, they can still be discriminated and segregated (Laws, 1980). Many methods in computer vision, such as stereo matching, structure from motion, and depth from defocus, rely on the presence of surface texture in order to function (Sundaram & Nayar, 1997).

Watt (1988, 1992, 1995) has suggested another powerful use of texture: handling complexity in interpreting visual information. In many situations, the visual system processes coarse spatial scales for spatial relationships between regions, whereas it processes fine scales for information about surface qualities. This allows the visual system to avoid computing spatial relationships between every element in a scene—something that would be extremely time-consuming—and instead, to compute relationships only for a set of regions segregated based on their texture similarities. The scale at which this distinction is made depends on task and scene characteristics. For example, in Figure 2.1(c) on the page before, the division is at the scale of large boulders; at finer spatial scales regions are analyzed for properties such as surface orientation, whereas at larger scales only the locations of regions are extracted.

2.2 Segmenting the Visual Input

One of the first tasks facing a visual system is parceling an input into regions that correspond to some higher level of organization. The array of light intensities in the retinal image needs to be organized into larger chunks to be useful to the organism.

This is a problem of *segmentation*, of partitioning the visual input into a set of regions. Although segmentation is sometimes meant in the sense of isolating object parts from the rest of the image (Jain, Kasturi, & Schunk, 1995), here I use the term in a broader sense of segregating image regions on the basis of different image properties. The image patches that result may or may not be parts of objects. Interpreting their relationship to objects is a process assumed to be performed elsewhere in the visual system, because the task of labeling structures in the input is full of ambiguity and depends largely on the use to which the information is put (Bruce et al., 1996; Kaplan & Kaplan, 1982; Marr, 1982).

At the early stage of processing being studied here, the segmentation is preliminary and may have errors in it—some region boundaries may appear where the physical scene has a single surface, and some boundaries in the scene may be missed. Texture is one source of information that can be used to derive a rough segregation of the image. Because texture is a property of surfaces (or collections of structures that can, as a first approximation, be treated as a surface), it follows that segmentation based on texture provides a first estimate of where surfaces may be located in the visual input.

The result of a texture-based segmentation will be only an approximate description. A complete visual system is not presumed to rely exclusively on this single estimate, though in situations in which no other information is available, it presumably could. It is clear that the human visual system has at its disposal many processes that work in parallel with other sources of information. Their collective estimates serve to reinforce and correct each other—the sum of many approximate segmentations can yield a more accurate sketch of the input than can the individual estimates alone.

2.2.1 Texture as a Basis for Image Segmentation

Psychologists have long been interested in the tendency of the visual system to spontaneously organize visual inputs into coherent regions (Gurnsey & Laundry, 1992). An example of a simple stimulus that can be used to demonstrate this tendency with texture is the image shown in Figure 2.1(f) on page 17. Images such as this can be shown to human observers for durations as short as 100 to 200 msec, and their visual systems can still detect the presence of multiple regions of texture.

Analytical studies of texture-based visual segmentation in humans began in earnest with the works of Julesz (1962, 1965) and Beck (1966a, 1966b, 1967). The phenomenon that drove this research was the effortless and nearly instantaneous manner in which the visual system often could segregate two texture patches placed side-by-side. Julesz emphasized a statistical perspective, attempting to describe the properties of random-dot textures that lead to different textures being preattentively discriminable from each other. Beck, on the other hand, approached the problem from a Gestalt psychology perspective, casting it as a question of how the visual system automatically grouped elements such as line segments together into larger wholes.

Julesz's work was especially influential. His study of statistical properties of textures eventually led to conceptualizing texture segregation in terms of the interactions of conspicuous local features—the *texton theory* (Julesz, 1984, 1986; Julesz & Bergen, 1983). Textons are features such as elongated blobs, line segments, terminators of line segments, and intersections between segments. The texton theory holds that it is differences in the density of texton features, and not their spatial organization, that leads to effortless texture segregation. Figure 2.1(f) on page 17 gives an example of the type of stimuli used in experiments on visual segmentation of texton-based textures.

Elements such as elongated blobs and line segments are the same types of features that early artificial intelligence theories conceived as being the basic starting points of visual analysis. Unlike texton theory, however, these models focused on the need for a level of organization beyond simply assessing texton densities (c.f. Olson & Attneave, 1970). In Marr's (1976) and other feature-based models of texture analysis (e.g., Fox & Mayhew, 1979; Voorhees & Poggio, 1987, 1988), the local features are first extracted at each location in an image, and then subsequently are subjected to various grouping operations. The grouping operations used are based on the principles of proximity and similarity—the principles studied by the Gestalt psychologists (Bruce et al., 1996).

The development of theories explaining early vision in terms of mechanisms tuned to detecting oriented spatial frequencies (Campbell & Robson, 1968; Campbell, Cooper, & Enroth-Cugell, 1969; Robson, 1966) inspired a new model of visual texture perception. This was the view that instead of involving an initial feature detection step, discrimination of textures may be more directly explained in terms of the activities of a collection of visual mechanisms selectively responsive to limited ranges of spatial frequencies and orientations in the image (Harvey & Gervais, 1978, 1981; Richards & Polit, 1974; for reviews, see Bergen, 1991, and Sagi, 1991). This view held that two textures were perceived as similar when they evoked activity in similar sets of visual spatial-frequency detectors. Historically, a related idea had already been pursued in computer vision in the form of spatial-frequency analysis for texture classification (Bajcsy, 1973; Dyer & Rosenfeld, 1976; Rosenfeld, 1962; Lendaris & Stanley, 1970). The computer vision approaches were motivated by a desire to automatically discriminate and classify regions of texture in images, especially satellite images.

Early approaches were often global in nature, analyzing entire images using methods such as the Fourier transform. They evolved into methods that used local measures (i.e., in small neighborhoods at many locations in the visual field). One of the earliest examples was developed by Laws (1980). His computer program used linear spatial filters to measure texture properties locally at each point in the image, and then segmented the image on the basis of the filter responses. The filters had specially-designed profiles intended to detect texture qualities such as coarseness, roughness, uniformity, density and regularity.

These various developments eventually led to conceptualizing preattentive texture-based segmentation in terms of the operation of localized spatial filters resembling cortical cells. (Appendix A summarizes some of the concepts behind spatial filtering models of cortical neurons.) The mechanisms are viewed as acting directly on the intensity distribution in the input image, thus obviating the need for feature detection, and having properties thought to be characteristic of the receptive fields of simple and complex cells in the primary visual cortex. The result has been an explosion of psychological and neural modeling efforts (e.g., Beck, Sutter, & Ivry, 1987; Bergen & Adelson, 1988; Bergen & Landy, 1991; Caelli, 1993; Casadei, Mitter, & Perona, 1992; Chubb & Landy, 1991; Clark, Bovik, & Geisler, 1987; Fogel & Sagi, 1989; Graham, Sutter, Venkatesan, & Humaran, 1992; Gurnsey & Browse, 1989; Julesz & Kröse, 1988; Landy & Bergen, 1991; Malik & Perona, 1990; Mesrobian & Skrzypek, 1995; Rentschler, Hübner, & Caelli, 1988; Rubenstein & Sagi, 1990; Sagi, 1991; Sutter, Beck, & Graham, 1989; Tadmor & Tolhurst, 1994; Turner, 1986; Victor & Conte, 1989; Watson, 1983; Van Hulle & Tollenaere, 1993; for reviews, see Bergen, 1991, Graham, 1991, and Sagi, 1995). The characteristics of the most popular type of model are discussed in the next section.

Many computer vision researchers have followed similar ideas. A large number of computer vision algorithms for texture-based segmentation using spatial filters have been developed (e.g., Azencott, Wang, & Younes, 1997; Chang & Kuo, 1993; Coggins & Jain, 1985; Bovik, Clark, & Geisler, 1990; Bovik, 1991; D'Astous & Jernigan, 1984; Dunn, Higgins, & Wakeley, 1994; Farrokhnia, 1990; He & Wang, 1991; Hsu, Calway, & Wilson, 1992; Jain & Farrokhnia, 1991; Jain & Karu, 1995; Khotanzad & Chen, 1989; Lifshitz & Pizer, 1990; Randen & Husøy, 1993, 1994; Reed, Wechsler, & Werman, 1990; Unser & Eden, 1990). Some of these approaches have been motivated by drawing on the similarities to the theoretical neural mechanisms developed in psychological and neural modeling work; other approaches simply use spatial filters or direct spatial-frequency measurements as an effective starting point for texture analysis. Some examples of tasks to which computer vision systems for texture segmentation have been applied include automated inspection of industrial materials and automated analysis of satellite and medical images.

2.2.2 Spatial Filtering Models of Texture Segmentation

Neurophysiological studies support the idea that cortical simple and complex cells have appropriate response properties to serve as possible starting points for texture segmentation (Knierim & Van Essen, 1992; Nothdurft, 1985; Nothdurft & Li, 1985; Nothdurft, 1990; Van Essen et al., 1989). This has made segmentation approaches based upon spatial filtering an especially popular class of models of human texture segregation performance.

The vast majority of these models are based on the following simple set of ideas (Bruce et al., 1996; Chubb & Landy, 1991; Sagi, 1991):

1. The responses of filters tuned to each combination of frequency and orientation are registered in a map of responses somewhere in the visual system. The *maps* are simply arrays of neurons having a systematic correspondence to locations in the visual field.

2. Regions of an input image that are differently textured must differ in their internal spatial structure. Spatial filters of various sizes and orientations are selectively activated by different spatial structures, depending on the degree of match between a filter's properties and the portion of the image falling within its receptive field. If a variety of filters are applied to an input containing areas of different textures, at least some of the filters will be more strongly activated by one texture than another.

3. Two textures are discriminable when filter responses are relatively constant in value within each texture region, and different in value between the texture regions. To accentuate the homogeneity of filter responses within texture regions, the filter outputs can be averaged in a neighborhood about each spatial location using a moderately-sized smoothing function. (The averaging function must be applied to each separate response map; that is, to the responses within each separate spatial frequency/orientation band.)

4. Segmentation can be performed on the resulting sets of values either by searching for rapid changes (which are taken to indicate boundaries between different texture regions), or by clustering regions with similar values.

An example of this class of models is provided in the work of Bergen and Landy (1991; Landy & Bergen, 1991). Their model is illustrated in Figure 2.2 on the following page. It begins by processing a visual input using linear spatial filters of different sizes and orientations. Bergen and Landy used four spatial frequencies and four orientations; the number of frequencies and orientations differs for other models. Each filter's output is then passed through a nonlinear operation, in this case squaring. The need for such a nonlinearity is due to the fact that linear filters by themselves produce the same average value over the whole input, and therefore by themselves would not help discriminate between different textures. A nonlinearity at this stage is needed to differentiate the responses prior to averaging. The use of squaring in particular has certain advantages: it simulates the rectified outputs of two coincident simple cells, one on-center and one off-center, and it has the effect of emphasizing strong responses relative to weak ones. After the nonlinearity, several filter outputs are averaged in a small neighborhood at each location to simulate the response of a complex cell.

Many contemporary texture segmentation models employ an initial component of this type, using some variation on mechanisms resembling complex cells (e.g., Caelli, 1985; Fogel & Sagi, 1989; Graham, 1992; Graham et al., 1992, 1992; Graham, Sutter, & Venkatesan, 1993; Graham & Sutter, 1996; Gurnsey & Browse, 1989; Mesrobian & Skrzypek, 1995; Rubenstein & Sagi, 1990; Sutter et al., 1989; Sutter, Sperling, & Chubb,

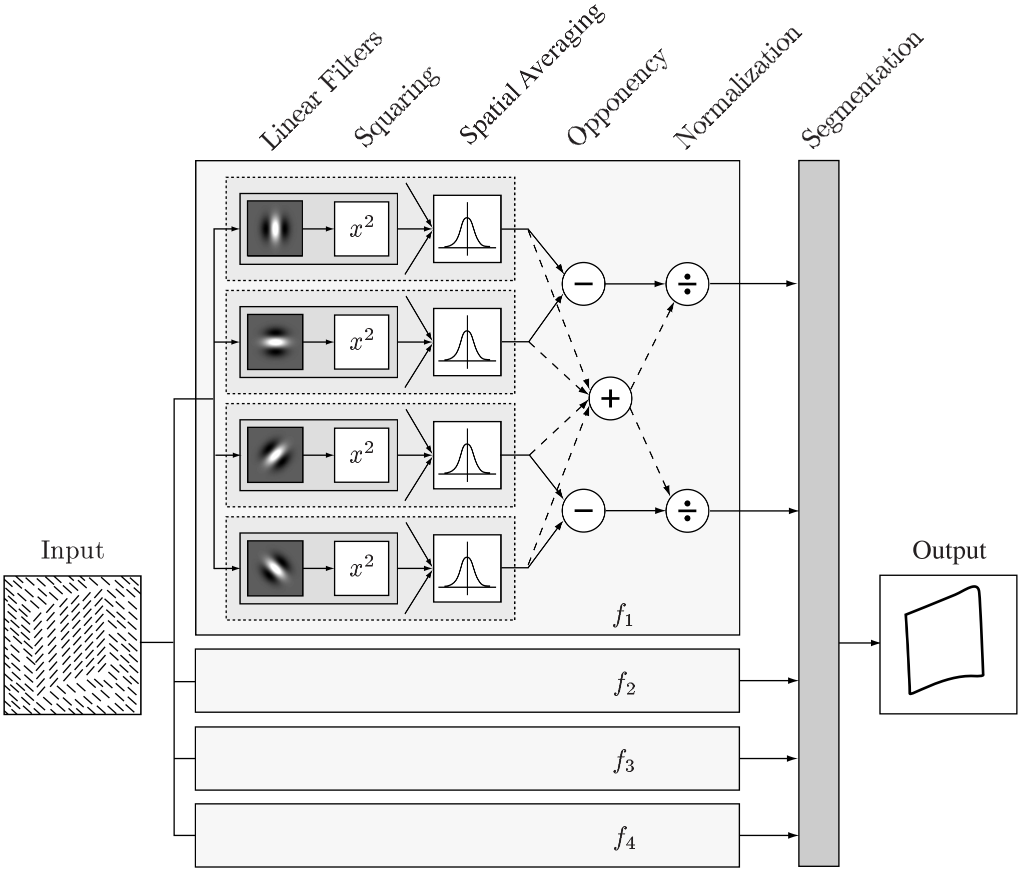


Figure 2.2: Illustration of texture segmentation model by Bergen and Landy (1991). The general architecture that they use is similar to that of many other models. Their model begins by analyzing an input image with linear spatial filters (i.e., linear receptive field operators) tuned to different spatial frequencies and orientations. In essence, these filters are arranged in arrays that are in register with the input, with one array of values for each combination of frequency and orientation. The responses of the filters are each squared and then averaged over a small spatial neighborhood, to simulate the processing performed by cortical complex cells. The subsequent stages of the model are more task-specific to the particular problem studied by Bergen and Landy (1991). The complex cell outputs are passed through an opponency stage to enhance orientation differences within each frequency f_i, and then normalized for contrast. The results are used in a segmentation process that looks for differences in the average regional responses.

1995; Sutter & Graham, 1995; Turner, 1986; Van Hulle & Tollenaere, 1993). The next stage in the model is more specific to the task studied by Bergen and Landy (1991). They were interested in the segmentation of textures defined by orientation differences. To enhance the filters' responses to different orientations, the fourth stage in the model subtracts the responses to orthogonal orientations (horizontal versus vertical, and left oblique versus right oblique). The fifth stage is a normalization step in which each filter's output is divided by the sum of the outputs of all filters tuned to a particular spatial frequency. This separates image structure information from the image contrast and also suppresses weak responses. At the end of this stage, the original input has in essence been converted into an intensity image, in which different regions of texture are marked by different magnitudes of regional cell activities. It is then a simple matter to use a standard, intensity-based edge detection mechanism to find the texture borders.

Some variants of this type of model do not use edge detection in the final stage (Caelli, 1993, 1995). Nevertheless, many of the core assumptions (such as that texture regions are distinguished by a certain commonality of filter responses) remain the same.

As noted in Appendix A, a complex cell can be modeled as taking the sum of multiple spatial filters that have properties similar to those of simple cells, pooled over a small region, and with some additional processing following the pooling. This makes a complex cell insensitive to the specific location of a stimulus within its receptive field while retaining sensitivity to other qualities such as spatial frequency and orientation. This is often called an "energy" mechanism, for technical reasons that are made more explicit in the next chapter. Intuitively, the energy analogy comes from the idea that the response of a complex cell can be thought of as representing the *amount* of a particular type of spatial structure, such as texture, within its tuning range and its region of spatial pooling (Adelson & Bergen, 1985; Bergen & Landy, 1991). Models of segmentation that rely on differences in "texture energy" are therefore often called *energy models* of texture segmentation.

One of the strengths of this class of models is that it is not necessary to be able to detect individual texture elements in order to segregate different regions. It is possible to use an alternative mode of analysis based directly upon the distribution of image intensities, by characterizing the stimulus in terms of spatial frequency and orientation. This is in contrast to approaches that emphasize extracting features from an image and performing texture processing using the results (Julesz, 1986; Marr, 1976; Voorhees & Poggio, 1988). Detecting individual texture elements in images of natural scenes is a difficult theoretical problem (Aloimonos, 1988; Blostein & Ahuja, 1989); moreover, many textures that humans can segment preattentively, such as filtered noise textures (Landy & Bergen, 1991), do not have any regular elements at all.

2.2.3 Limitations of Energy Models of Texture-based Segmentation

The segmentation energy model has been so popular that it has been nicknamed the "back-pocket model", because vision researchers routinely "pull it out of their back pockets" to interpret new examples of preattentive texture segregation (Chubb & Landy, 1991, p. 297). Despite its compelling qualities, there is mounting evidence that this type of model is inadequate for explaining texture-based segmentation in humans.

At the core of the problem are the limited types of stimuli that have usually served as test cases in research on this class of models. The vast majority of energy models of segmentation have been developed expressly for the problem of segmenting patches of flat textures facing the viewer. For this, the models have been relatively successful in demonstrating performance correlated to that of humans on the same tasks (e.g., Fogel & Sagi, 1989; Graham, 1991; Landy & Bergen, 1991; Malik & Perona, 1990; Sagi, 1991). This success notwithstanding, focusing on this type of stimulus has obscured some important limitations of these models. There are at least three lines of evidence supporting this conclusion.

First, there is evidence from psychophysical experiments by Gurnsey and Laundry (1992). Many energy models of segmentation are based on detecting abrupt gradients between texture patches. This is the case in the model by Landy and Bergen (1991) discussed above, in which segmentation is performed by searching for the transitions between otherwise homogeneous regions of texture in the image. Gurnsey and Laundry tested whether this assumption is valid for humans. They presented observers with stimuli consisting of flat texture patches in which the boundaries between texture regions were smoothed. This made the gradients between texture regions less pronounced. Most energy models of segmentation would predict significantly impaired segmentation performance in that situation, but Gurnsey and Laundry (1992) found only a modest 10% decline in their subjects' performance. This suggests that sharp gradients may not always be necessary for human texture-based segmentation.

Second, there is evidence from work by He and Nakayama (1994b; Nakayama & He, 1991) that manipulating certain surface qualities affects the ability of observers to perform rapid texture segmentation. In their experiments, He and Nakayama presented human subjects with the task of discriminating a target region containing L-shaped elements from a background region containing bar-shaped elements. By manipulating stereoscopic cues, they could make the L-shaped and bar-shaped elements appear to be either separate shapes located above a surface, or portions of larger shapes partially occluded by nearer rectangles. When the stereoscopic cues made the elements appear to be larger shapes occluded behind nearer shapes, the observers' segmentation performances were impaired compared to the case where the stereoscopic cues made the elements appear to be separate L-shaped elements floating above a surface. This is surprising because the two-dimensional shapes of the elements were identical in both cases, which means that an energy model of segmentation would predict no difference in performance. He and Nakayama included control conditions in which they removed the occlusion effect while maintaining the two-surfaces-in-depth quality, and in this situation, the subjects' performances were equal for both versions of the stimulus presentations. These results are difficult to explain with an energy model of segmentation. He and Nakayama argued their evidence shows that, “in rapid texture discrimination, the visual system cannot ignore information regarding spatial layout” (He & Nakayama, 1994b, p. 151). They concluded that texture-based segmentation cannot be understood based simply on the properties of the spatial filtering performed by cortical neurons; instead, it appears to involve an aspect of surface representation as well.

Finally, and most important for the goals of the present research, most energy models of texture segmentation are designed with an assumption that prevents their application to general scenes. This nearly universal assumption is that a given texture in an image will produce roughly identical responses, across the texture’s entire extent, in some subset of spatial filters. The models require that the filter responses characterizing the texture must be stable across the entire region, because homogeneity of filter responses is precisely the models’ basis for segregating the region of texture in the image. The texture in the image must therefore be free from distortions due to surface curvature or orientation with respect to the viewer. For this to happen, the physical surface on which the texture lies must be approximately flat and oriented perpendicularly to the viewer’s line of sight.

This assumption is acceptable for the limited kinds of situations to which these models have been applied, but it is rarely satisfied in natural viewing conditions. Figure 2.3 on the next page illustrates the problem. If a textured surface is curved or slanted in three dimensions, the texture pattern projected in the image shows variations in texture *within* a single region. This leads to different sets of spatial filters responding at different locations within a single texture region. Thus, the assumption of homogeneous filter responses is violated, and the simple approach to segmentation described above fails in this situation (Krumm & Shafer, 1994).

The fact that natural environments contain surfaces that, in general, are neither flat nor facing the viewer means that a segmentation mechanism must be able to cope with textures that are distorted in the image. The popular energy model does not offer a viable explanation of texture-based segmentation for the general case; it is necessary to go beyond this formulation and allow more realistic types of inputs. Further, variations in texture do not merely constitute noise to be overcome: they provide important information, as described in the next section.

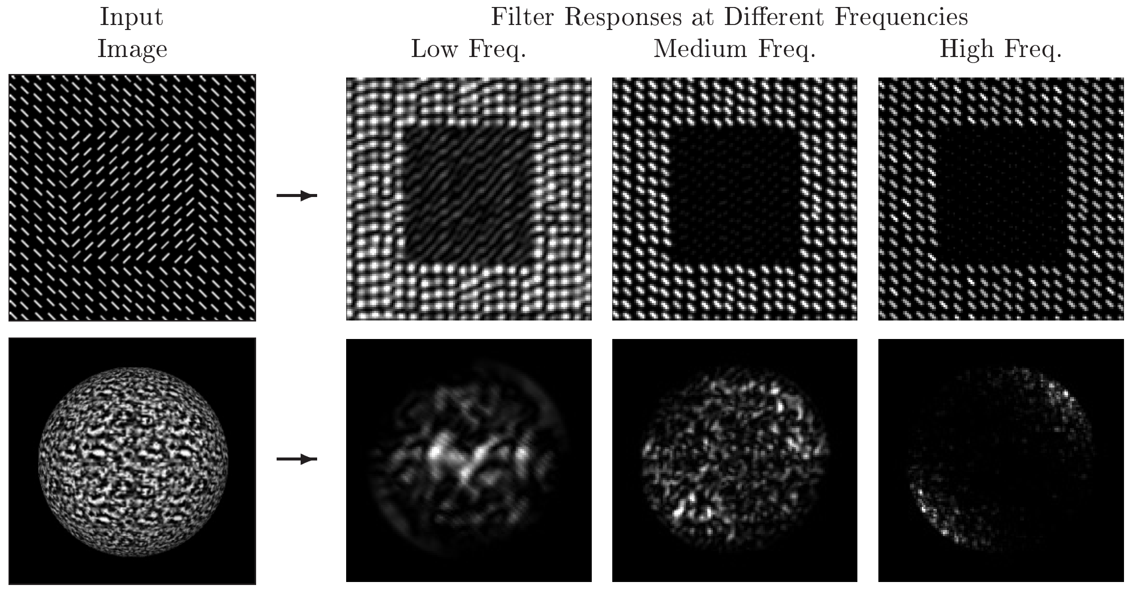


Figure 2.3: Illustration of spatial filter responses to texture. (Top row) Results for a simple artificial texture pattern inset inside a larger one, consisting of orthogonally-oriented line elements. The input image is shown at the left, and the responses of complex-cell-like filters tuned to several spatial frequencies are shown in the right-most three columns. All filters are tuned to an orientation of 45° measured counter-clockwise from vertical. Note how the responses to the outer texture are homogeneous over the entire extent of that texture in the image: filters that are activated by that texture are activated over the whole region equally. (Bottom row) Results for a curved, textured sphere. Again, the responses are shown in the right-most three columns. Note that the responses in this case are not homogeneous over the extent of the surface—filters tuned to different spatial frequencies are activated at different locations over the texture in the image.

2.3 Estimating the Overall Shapes of Surfaces

In order to make an initial estimate of the spatial layout of surfaces in a scene, the visual system must do more than just produce a segmentation. It must also recover information *about* the segregated regions. One of the most basic types of layout information is the three-dimensional shape of a surface as seen by the viewer.

The way in which texture is distorted when a surface is projected onto an image is a function of the shape of the physical surface, its distance from the observer, and its orientation relative to the observer. The distortions thus provides cues to three-dimensional shape. Pictorial artists have used this fact for centuries in their paintings. However, Gibson (1950a, 1950b; Gibson & Cornsweet, 1952; Gibson & Dibble, 1952; Beck & Gibson, 1955) is generally acknowledged to have been the first to systematically study the psychology of perceiving spatial layout from patterns of texture.

The analysis of surface shape using texture is often called *shape from texture*, a general term that also covers estimating the orientations of planar surfaces. Shape-from-texture methods often emphasize recovering highly accurate information about object shapes. In contrast, the emphasis in this research is more on estimating overall surface shape and on situations in which surfaces may only be implicit, as in the apparent surface formed by trees at the edge of a forest or a cluster of boulders. Therefore, in this work I use the term *texture-based shape estimation* instead of "shape from texture", to emphasize the more qualitative nature of the processes under study here.

As in the case of texture-based segmentation, I expect that the results of these shape estimation processes are approximate and only one part of the overall perception of a scene. Other preattentive processes such as stereopsis will contribute to the estimates of surface shapes, and together the various estimates can reinforce and correct each other. In this context, it is worth noting that experimental evidence suggests the visual system processes depth information from stereoscopic vision and shape from texture mostly independently, and combines the results using a relatively simple rule (Buckley, Frisby, & Spivey, 1990; Johnston, Cumming, & Parker, 1993).

2.3.1 Texture Distortions as Cues to Surface Orientation and Shape

There are many types of texture distortions that can arise from the way that the retinal image of a physical surface is formed. Any or all of them could potentially be interpreted by a visual system to infer shape and orientation from texture. There are two principal causes of visual texture distortions (Cumming, Johnston, & Parker, 1993; Stevens, 1983), summarized here but treated in more detail in Chapter 4:

- *Changes in distance from the viewer to a surface.* Changes in distance to a surface produce changes in the local scale of elements projected onto the eyes. This is known as perspective. It induces several effects, and two effects in particular have received significant attention in texture research. One is that the apparent sizes of texture elements decrease with increasing distance from the observer. This results in a size gradient in the image and is known as *scaling*, or alternatively, a *perspective gradient*. A second effect is that elements appear to become more closely spaced with increasing distance from the observer, leading to a perceived change in texture element *density*. Figure 2.4 on the following page illustrates these effects.

- *Changes in surface orientation with respect to the line of sight.* This induces two primary effects on the projected texture. One is that texture elements in the image undergo increasing *compression* with increasing surface slant away from the viewer; that is, the apparent ratio of element width to length (the aspect ratio) changes with the physical surface's orientation. A second effect is an increase in the apparent

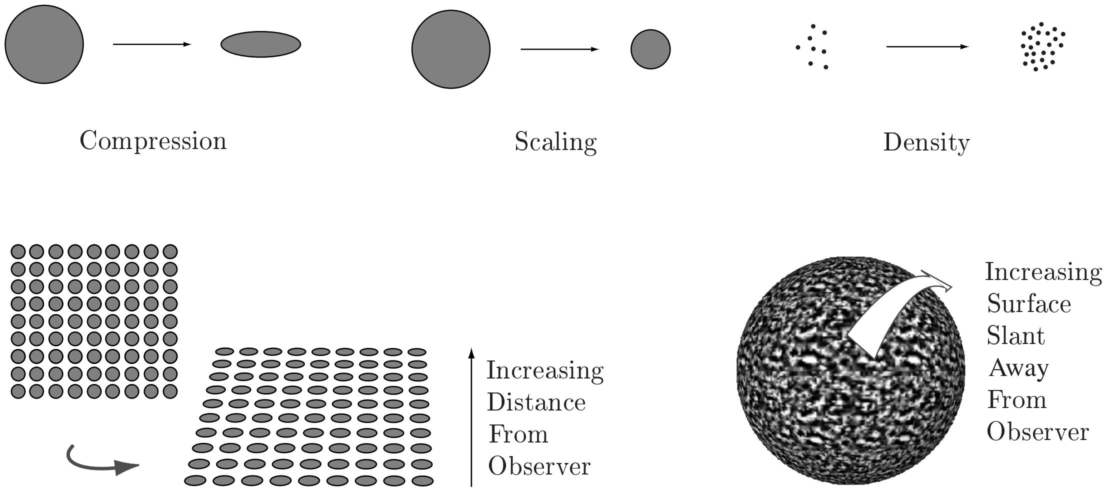


Figure 2.4: (Top row) Illustration of different types of texture distortions that can serve as cues to surface shape and orientation (after Blake et al., 1993). Texture *compression* involves a change in the perceived ratio of texture element width to length; texture *scaling* involves a change in the overall size of texture elements; and changes in texture *density* involve a change in the number of perceived elements per unit viewing area. (Bottom row) Texture distortion cues are evident with both flat and curved surfaces.

texture *density* with increasing slant away from the viewer. (Density changes can occur due to both changes in surface orientation and changes in distance from the viewer.)

Being able to use these sources of information requires making assumptions about the nature of the texture on the physical surface. The most commonly-made assumption is that the surface texture is homogeneous in some way, such as in the density of texture elements covering the surface. Then changes in the characteristics of the texture between different points in the image projected onto the eyes can tell the observer something about the orientation and shape of the surface. Another type of assumption is directional isotropy, the assumption that all orientations of texture elements are equally likely on the physical surface. Different assumptions lead to different methods for estimating surface shape and orientation, and a number of assumptions have been used in research on texture-based estimation of shape (Rosenholtz & Malik, 1997). This issue is taken up again in Chapter 4.

In his original analysis, Gibson (1950a, 1950b) focused primarily on the case of planar surfaces, such as a flat region of the ground, assumed to be covered with texture elements at a uniform density. He observed that the perspective effects in natural viewing situations produce changes in the density of texture grains as perceived by the viewer, and these changes are directly related to the spatial orientation of a surface. He therefore argued that it is gradients of texture density that lead to the perception of surfaces receding in depth. Gibson's work stimulated considerable follow-on work in both computer vision and psychology.

Stevens (1981) noted that early psychological studies by Gibson and others were inconclusive because they did not adequately remove other potential textural sources of information about surface orientation and distance. He and Witkin (1981) also argued that, for theoretical reasons, density is a poor source of information about either surface orientation or distance-to-surface, and was unlikely to be used by the visual system. They separately proposed alternative methods based on arguments from computational considerations.

Subsequent psychological research (Blake et al., 1993; Cumming et al., 1993; Cutting, 1984; Cutting & Millard, 1984; Johnston et al., 1993; Todd & Akerstrom, 1987) further clarified the sources of information likely to be used by the human visual system. Cutting and Millard (1984) found that scaling is the most important source of information for estimating the flatness of surfaces, and is much less effective for curved surfaces; conversely, compression is the most important source of information when judging curved surfaces. Cutting and Millard (1984) and Blake et al. (1993) found experimentally that density can also be a useful, albeit weak, source of information, contrary to the theoretical arguments of Stevens (1981) and Witkin (1981). Todd and Akerstrom (1987) found that compression only provides useful shape information if the orientations of the compressed elements are consistent with the direction of slant; other things being equal, randomizing the orientations of compressed elements virtually eliminates the impression of curvature. Further, they found that subjects' judgments of surface shape were accurate even for surfaces covered with irregularly shaped, elongated elements. The latter result implies that the visual system does not rely on assumptions about the regularity of the shapes of the physical texture elements, and argues against approaches that rely on measuring actual changes in texture element shapes (e.g. Blostein & Ahuja, 1989; Marr, 1982; Stevens, 1981).

In the context of computer vision, two broad classes of approaches have emerged in reaction to Gibson's work: methods that work from image features, and methods that work directly from image intensity values.

1. Methods in the former category begin by extracting features from an image. The features may be whole texture elements or something derived from texture elements, such as short edge segments. These methods then infer surface orientation based on one of three approaches: (a) measuring the distortions in the shapes of texture

elements in the image (e.g., Ikeuchi, 1984; Kender & Kanade, 1980; Moerdler & Kender, 1987; Ohta, Maenobu, & Sakai, 1981); (b) measuring changes in the spatial distribution of elements or features derived from the elements (e.g., Aloimonos, 1988; Aloimonos & Swain, 1985; Blostein & Ahuja, 1989; Davis, Janos, & Dunn, 1983; Kanatani, 1984; Pentland, 1986; Rosenfeld, 1975; Stone & Isard, 1995); or (c) measuring deviations from statistical isotropy of edge segments (Blake & Marinos, 1990; Witkin, 1980, 1981). The first approach in this class (using distortions in the shapes of texture elements) requires being able to know or infer the true shapes of the elements on the physical surface. This limits the applicability of this approach primarily to regular textures, since natural textures tend to be so varied and irregular that the shapes of elements often cannot be described easily. The second approach (using element distributions rather than element shapes) is less restrictive, and also more compatible with evidence, mentioned above, that humans do not seem to rely on judging individual texture element shapes. An example is the work of Aloimonos (1988; Aloimonos & Swain, 1985). His method uses the changes in densities of texture edge elements in an image as a basis for estimating the orientations of planar surfaces, under the assumption that element edges are distributed with uniform density on the physical surfaces. The third approach (measuring deviations from statistical isotropy) can estimate surface orientation at individual locations, without having to compare multiple locations, but it requires textures to be isotropic. Many natural and artificial textures do not satisfy this requirement (Aloimonos, 1988).

2. Approaches in the second category—working directly from the image intensity—do not rely on first extracting texture elements from an image. Nearly all examples in this class employ some variation on spatial-frequency analysis, and are the topic of the next section.

The neurophysiological bases of texture-based estimation of shape have not been studied extensively. Some workers have searched for neural correlates of shape processing, for example in the form of neurons selectively responsive to slant (Gallant, Van Essen, & Nothdurft, 1995). The results to date have been largely negative—neurons selectively tuned for surface orientation have not been found in the cortical areas studied. However, this does not rule out the possibility of their existence in brain areas not yet investigated. Theories about the mechanisms used in visual estimation of shape from texture thus remain mostly speculative, informed primarily by psychological studies and computer vision experiments.

2.3.2 Spatial Filtering Models of Texture-Based Shape Estimation

Bajcsy and Lieberman (1976) had the important insight that the effects of compression and other texture distortions can be related to variations in spatial-frequency and orientation content across an image. In other words, texture distortions can be analyzed not only in the spatial domain (in terms of the effects of image projection on individual texture elements), but equivalently in the spatial-frequency domain. Bajcsy and Lieberman developed the first computer vision algorithm that estimated the slants of surfaces based on these principles.

The relationships between the spatial-frequency spectrum in the image and texture compression, scaling, and density are relatively straightforward:

- Compression leads to flattening of texture elements, as shown in Figure 2.4 on page 29. This causes changes in both the dominant spatial frequencies and the dominant orientations in different parts of the image. The changes are as follows. In image areas where texture elements are made smaller due to compression, there is a commensurate increase in the proportion of high spatial frequencies. In addition, as elements are increasingly flattened, the orientations of the elements tend towards a direction perpendicular to the direction of the gradient of compression. This also means that less-compressed regions contain a greater range of orientations, whereas more-compressed regions tend toward one predominant orientation.

- Scaling (and in fact, density gradients, too) also causes changes in the dominant spatial frequencies in different parts of the image. The changes are as follows. Portions of a surface closer to the observer appear to have larger texture elements and greater inter-element spacing. These qualities translate into lower spatial frequencies in the image. Portions of the surface farther away from the observer appear to have smaller elements spaced more closely together, and this translates into higher frequencies in the image.

These effects make it possible to analyze the spatial-frequency spectrum in an image in order to infer orientation and shape properties of physical surfaces. It is an approach that can be performed without having to detect or extract individual texture elements from the image—it allows texture to be characterized through regional, aggregate properties. An additional advantage of this type of approach is that the effects listed above co-occur in most natural situations: for example, slanting a surface induces gradients of compression, size and density all at the same time. Methods based on variations in the spatial-frequency spectrum may thus lead to a more robust estimate of surface shape without having to measure each type of texture distortion individually.

In their seminal paper, Bajcsy and Lieberman (1976) estimated the spatial-frequency content in small windows at different locations in an image using the spectrogram (Koenig, Dunn, & Lacey, 1946), a local Fourier transform method. For their test cases, they examined surfaces with homogeneously-sized texture elements. They derived texture descriptors based on the spatial-frequency spectrum, and correlated values of these descriptors to the slant of a flat surface in the image.

Despite this promising start, research attention for a time seemed to shift almost exclusively to methods based on extracting image features (Stevens, 1981; Witkin, 1981). Eventually, however, the spatial-frequency-based methods saw renewed interest. An important development was the realization that texture compression and other kinds of information could be extracted by using spatial filters as an approximation to computing a spectrogram, thus giving this class of models a possible biological grounding. Many models have been developed based on this idea, in addition to models based on the spectrogram and related measures (e.g., Black & Rosenholtz, 1995; Brown & Shvaytser, 1988, 1990; Gårding, 1992a, 1992b, 1995; Jau & Chin, 1988; Jones & Malik, 1992; Krumm & Shafer, 1994, 1995; Malik & Rosenholtz, 1993, 1994a, 1994b, 1997; Reed et al., 1990; Reed & Wechsler, 1990; Sakai & Finkel, 1994, 1995, 1997; Stone & Isard, 1995; Super & Bovik, 1992, 1995; Turner, Gerstein, & Bajcsy, 1991).

The most technically advanced of these approaches are the recent computational models of Gårding (1992a, 1992b, 1995) and Malik and Rosenholtz (1994a, 1994b, 1997). Gårding (1992b) developed a rigorous framework relating surface orientation and curvature to the distortions that texture undergoes in perspective projection onto an image. He used the tools of differential geometry (O'Neill, 1966) to express the mapping between a surface and its image, then derived relationships between texture distortions in the image and the shape parameters of the surface. Malik and Rosenholtz (1993, 1994a, 1994b, 1997) extended Gårding's (1992b) analysis, and proposed that texture distortions are appropriately modeled as an affine (i.e., linear) transformation between neighboring patches of texture. This formulation of *affine texture distortion* subsumes the texture gradients studied previously by other researchers, and is powerful enough to recover both surface shape and local orientation. Malik and Rosenholtz demonstrated their algorithm with a computer implementation that finds the shape and orientation of a surface using these principles. Their approach begins by computing the spectrogram of the image, then estimates the affine texture distortion from the differences in oriented spatial-frequency content between neighboring patches in the image. Gårding (1995) further improved the approach of Malik and Rosenholtz (1994b) by finding a simpler form for the relationships between texture distortion and surface shape.

The approaches developed by Gårding (1995) and Malik and Rosenholtz (1994b, 1997) are powerful, but complex and mathematically intensive. It is unclear whether a biological visual system can implement some of the computations required, though it is true that there is also no evidence against it. On the other hand, psychophysical evidence from work by Reichel, Todd, and Yilmaz (1995) suggests that humans do not seem to obtain a precise representation of shape from texture—people's estimates of surface shape from texture are rather coarse. This suggests that the mechanisms of texture-based shape estimation in humans may be approximate in nature.

In this context, there is intriguing recent work by Sakai and Finkel (1995) that suggests that qualitative shape estimation can be performed using relatively simple principles. Sakai and Finkel examined the question of whether the human visual system uses the whole spatial-frequency spectrum across a surface in order to detect texture compression, or whether the system characterizes the spectrum in some simpler form and only tracks the changes in that characterization. Using psychophysical experiments, they demonstrated that the latter seems to be the case—the human visual system does not appear to use the whole spectral distribution in its estimation of surface shape. This makes sense as an adaptive strategy, because tracking a small set of measurements instead of the whole spectrum is less demanding computationally. Sakai and Finkel proposed a particular quantity that the visual system may use in judging surface shape: the local average of the dominant (peak) spatial frequencies. They demonstrated with human subjects that the degree of perceived surface curvature is correlated with the magnitude of the change in the average peak frequency across a surface. In addition, they implemented a neural network simulation to estimate surface shape using this measure.

The work of Sakai and Finkel (1995) is a key element of this dissertation project, and it is discussed in more detail in Chapter 4.

2.3.3 A Limitation of Current Texture-Based Shape Estimation Models

Shape estimation based on spatial-frequency analysis is a promising approach to modeling this process in the visual system. Because simple and complex cells respond to limited ranges of spatial frequencies, a collection of such mechanisms can be used to detect changes in the spatial-frequency spectrum resulting from texture compression, scaling and density gradients. Because the cells are also sensitive to limited ranges of orientations, they can detect orientation changes resulting from the compression of texture elements. Work such as that of Sakai and Finkel (1995) even provides a candidate neural model of this processing.

A limitation of almost all existing computational models of texture-based shape estimation is that they are designed with the assumption that the input contains only a single surface [e.g., Figure 2.1(e) on page 17]. This is understandable given the research goal of focusing on the shape estimation problem. However, real-world scenes usually contain multiple regions of texture, and most current models cannot cope with such inputs because they assume that texture inhomogeneities in an image are related only to changes in the shape of a single surface. If there are multiple surfaces in the input, the changes in textures at region borders confuse the shape estimation process.

Thus, the majority of existing models of texture-based shape estimation cannot be applied directly to general inputs. They must either be paired with a segmentation method, or else modified in some way to allow inputs having multiple texture regions.

2.4 Integrating Processes of Texture-Based Segmentation and Shape Estimation Into a Common Architecture

In the sections above, I summarized a number of models of texture-based segmentation and shape estimation. I also pointed out problems in the most popular existing models of each process. The problems are due to certain underlying assumptions:

1. Almost all existing models of texture-based segmentation assume that image regions are characterized by homogeneous textures, and that *texture inhomogeneities indicate region boundaries.* This assumption is violated when the input contains a textured surface that is curved with respect to the observer. In that case, even if the physical surface itself is homogeneously textured, the apparent texture in the image will be distorted and not homogeneous.

2. Texture distortions arising from curved or slanted surfaces are precisely what allows the visual system to recover information about surface shape and orientation. However, almost all existing models of texture-based shape estimation are designed with the assumption that *texture inhomogeneities indicate changes in surface shape or distance.* This assumption is violated when the input contains multiple textured surfaces. In that case, the apparent inhomogeneities in the image textures may not indicate changes in shape or distance, but instead may indicate transitions between surfaces.

As Krumm and Shafer (1994) have noted, these assumptions are incompatible. In order to perform both types of processes on general inputs containing multiple, curved or slanted textured surfaces, one or both of these assumptions must first be modified.

2.4.1 Attempts at Overcoming the Limitations of Existing Texture-Based Segmentation and Shape Estimation Approaches

An assumption made in many models of texture-based shape estimation is that segmentation should be performed first. Once the image is segmented, the shape estimation method should have no problem operating within each region separately, where presumably any texture inhomogeneities will indicate only changes in surface shape or orientation.

Although this may seem to be the most obvious solution to the problems discussed above, it is not the only possible one. A disadvantage of this approach is that it puts the onus entirely on the segmentation process, which must be able to handle curved or slanted surfaces in the input. Another disadvantage of this approach is that it ignores any possible contributions to the segmentation process that might be made available by the shape estimation component. Information arising from estimates of surface orientation and shape can potentially be a powerful aid to the segmentation process.

There is in fact a space of three possible avenues to explore in attempting to overcome the problems of existing texture-based processing schemes:

1. *Modify the segmentation model.* If the segmentation approach can be made to cope with texture distortions on its own, then other processes such as shape estimation can be performed independently within each segmented region.

2. *Modify the shape estimation model.* If the form estimation approach can be made to cope with multiple regions in the input on its own, then it does not need to rely on a segmentation process.

3. *Couple both processes into an integrated model.* Information from the shape analysis process can potentially contribute to the segmentation process. Rather than being separate, the two could be joined in some fashion.

All three approaches have been investigated, primarily from the vantage point of computer vision. I discuss each alternative in turn below.

Modifying the Segmentation Model

The central problem with most existing texture-based segmentation models is their reliance on the assumption that regions should be segmented based on the homogeneity of spatial filter responses. Overcoming this limitation requires allowing for gradients in the responses of the spatial filters. The issue, then, is how to best use the filter outputs.

Although a number of alternatives to the standard energy model of texture-based segmentation have been pursued in computer vision, few alternatives have been developed in the context of biologically-reasonable modeling. One of the few is the work of Lee (1993, 1995; Lee et al., 1991, 1992), who developed a model of texture-based segmentation that can cope with distorted textures and can potentially be implemented in a neural network. They achieved this by adapting a class of techniques known as *weak membrane models.* The idea behind the basic, two-dimensional weak membrane model is to fit a function to a set of data measurements. The function expresses segmentation as a competition between growing regions and forming boundaries. The membrane model treats the function as a thin, elastic sheet (a membrane), and stretches it across the set of data values. Where the data vary smoothly, the elastic sheet bends and conforms to the landscape. Where there are abrupt changes in the data, the elastic sheet breaks. The sharp breaks in the fitted function indicate borders between regions.

The basic weak membrane approach has been used for texture-based segmentation in the context of computer vision (e.g., Mumford & Shah, 1985; Blake & Zisserman, 1987; Geman, Geman, Graffigne, & Don, 1990). The contribution of Lee (1993, 1995; Lee et al., 1991, 1992) lies in extending the model to use as input data the responses of spatial filters resembling cortical complex cells. Their model begins by computing the responses of spatial filters tuned to different spatial frequencies and orientations at every point in the input. This produces a four-dimensional space of measurements (two dimensions in space, one in frequency, one in orientation). Lee's model then applies multiple weak membranes to these measurements with couplings in all four dimensions, allowing breaks only in the spatial x, y dimensions. In this way, the membranes are mutually constrained, and gradients in spatial frequency and orientation are considered in the segmentation process. The model can thereby accommodate changes in spatial frequencies and orientations that result from image texture distortions due to shape changes on surfaces. They dubbed their approach the *coupled-membrane model.*

Weak membrane approaches are somewhat involved mathematically, but they are a key component of this dissertation. Lee's (1993, 1995; Lee et al., 1991, 1992) work is revisited below and discussed in more detail in Chapter 5.

Modifying the Shape Estimation Model

A few researchers have attempted to overcome the one-surface-at-a-time limitation of models of texture-based shape estimation. One such attempt is the work of Moerdler and Kender (1987; Moerdler, 1989). They examined the problem of combining multiple methods for texture-based shape estimation. Their model uses several shape estimation methods as sources of constraints for estimating the orientation of each texture element in the input image. The methods are structural in nature, based on extracting individual texture elements (cf. Section 2.3.1). Their system derives multiple estimates for the most likely orientation of each texture element, then combines the results. The final result is a description of the estimated orientation of each texture element in the image.

More recently, Black and Rosenholtz (1995) extended Malik and Rosenholtz's (1994b) approach to handle inputs containing multiple regions of texture. Black and Rosenholtz proposed treating multiple textures as being approximately additive, in the sense that at each point where a spectrogram is measured, it contains the sum of frequency components originating from possibly multiple textured surfaces. Their approach represents the image conceptually as a set of layers, where each layer contains a region of texture. Within each individual layer, their method can estimate the affine texture transformation in the manner used by Malik and Rosenholtz (1994b). In this way, their approach can handle multiple textured surfaces and recover surface orientation for each one.

Coupling Both Processes Into an Integrated Model

A third approach to overcoming the problems of existing texture-based processing models is to modify both the segmentation method and the shape estimation method, to combine them and allow them to interact. The most advanced attempt so far to do this is the computer vision work by Krumm and Shafer (1994, 1995; Krumm, 1993).

Krumm and Shafer's (1994, 1995; Krumm, 1993) model begins by computing a spectrogram of the input using the Fourier transform with a fixed-size window. The transform is applied to each small image patch, and a small number of spatial-frequencies peaks are extracted from each patch. The change in the dominant frequencies between neighboring patches is used to compute an estimate of the surface slant. This measure of the slant is used to determine what the frequencies would be if the pair of patches were viewed frontally; in effect, the procedure undoes the effects of the 3-D image projection of a slanted surface. By comparing the "frontalized" peak spatial frequencies of each patch, Krumm and Shafer's model can group regions of the image that are likely to have come from the same physical surface. The final result is a segmented image and an estimate of the surface orientation of each region. Their formulation is designed for planar surfaces.

The work of Moerdler and Kender (1987; Moerdler, 1989), mentioned above in the context of modifying the shape estimation component, also represents an attempt to combine segmentation and shape estimation. The segmentation is implicit, in the form of an estimate of which texture elements are likely to be part of the same region. Although their method appears to be capable of combined segmentation and shape estimation, the published reports do not provide sufficient detail to judge the effectiveness of the segmentation scheme.

Beyond the works of Krumm and Shafer (1994, 1995; Krumm, 1993) and Moerdler and Kender (1987), there do not appear to have been any other attempts to date at integrating texture-based segmentation and shape estimation into a combined framework.

2.4.2 Evidence From Psychology and Neuroscience

The discussion above shows that several avenues have been investigated to overcoming the limitations of existing models of texture-based segmentation and shape estimation. With the exception of Lee's (1993, 1995; Lee et al., 1991, 1992) work, these attempts have come from the field of computer vision, without regard to biological modeling. There has been surprisingly little work on this problem in psychology and neuroscience.

One of the few relevant results comes from work by Kingdom and Keeble (1996). They performed psychophysical experiments using stimuli composed of a dense array of small, oriented elements (specifically, Gabor patches). The arrays contained either abrupt variations in element orientations or smooth variations in element orientations across regions. Kingdom and Keeble examined their subjects' ability to detect the patterns in brief presentations. They were able to account for their subjects' performance using a common model for both types of stimuli. This suggests that the detection of abrupt variations in textures (useful in texture-based segmentation) and smooth variations in textures (useful for shape analysis) may be subserved by a common visual mechanism. Kingdom and Keeble (1996) concluded that this mechanism acts on the outputs of spatial filters in the early visual cortex.

The work of He and Nakayama (1994b), discussed in Section 2.2.3, is also relevant here. Their results suggest a strong influence of surface interpretation on the processes of preattentive texture segregation. It seems that the mechanisms of texture-based segmentation do not operate solely and in isolation at the level of spatial filtering; rather, even at the earliest levels, the visual system attempts to interpret the results in terms of relationships to the three-dimensional layout of surfaces in the visual input.

Additional evidence from a number of other psychophysical and neurophysiological experiments also support this view of preattentive, surface-oriented processing in the early stages of the visual system. These include the phenomenon of filling-in of visual surfaces (Mattingley et al., 1997), the fact that rapid visual search can be influenced by three-dimensional interpretation of the elements being searched (Enns & Rensink, 1990, 1991, 1992), and experiments on the responses of V1 neurons to textured surfaces (Zipser, Lamme, & Schiller, 1996).

Taken together, these results suggest that texture-based segmentation and estimation of three-dimensional shape may be more closely related than is usually assumed. It may therefore be worth investigating a combined framework for texture-based segmentation and shape estimation.

2.4.3 A New Approach

The goal in this research is to develop a biologically-reasonable model of texture-based segmentation and shape estimation in a combined architecture. Of the work reviewed in this chapter, that of Krumm and Shafer (1994) and Moerdler and Kender (1987) are closest to achieving this goal. But although these models demonstrate that it is possible to combine segmentation with shape estimation, they are not intended nor suitable as models of biological processing. They use complex algorithms and data structures that are not easily mapped to neural mechanisms. Therefore, it seems worthwhile to investigate alternative approaches.

As discussed in Section 2.2.3, the energy model of segmentation cannot account for the visual system's ability to segment general texture inputs. The most promising alternative currently is the model developed by Lee (1993, 1995; Lee et al., 1991, 1992). Weak membrane methods offer a powerful framework, one that is capable of uniting several segmentation methods that have been developed in computer vision research (Geiger & Yuille, 1991; Zhu & Yuille, 1996). Also, the weak membrane approach is a parallel method that can be implemented in terms of local interactions between computational units—in other words, it is amenable to implementation in neural circuits. In fact, Lee (1993, 1995) has proposed how parts of his coupled-membrane model could map to parts of the primary visual cortex.

More interesting from the perspective of the present research is that the couple-membrane model already incorporates constraints coming from texture gradients. I show in Chapter 5 that the model can be modified slightly to use an intermediate result generated by the shape estimation model of Sakai and Finkel (1995). By doing so, the four-dimensional function used by Lee (1993, 1995; Lee et al., 1991, 1992) can be reduced to a three-dimensional function. This simultaneously reduces the computational requirement for the segmentation procedure, and also connects the shape estimation process to the segmentation process. The result has some of the flavor of an integrated architecture such as that of Krumm and Shafer (1994, 1995; Krumm, 1993).

The full details of this are left to Chapter 5, because they require having more information about both Sakai and Finkel's, and Lee et al.'s models. The end result, however, is a new model of combined segmentation and shape estimation.

2.5 Summary

To function in their environments, natural and artificial agents must be able to "perceive a wide three-dimensional field near the body, and must differentiate the solidity, continuity, spatial separation and mobility of objects in it" (Trevarthen, 1968, p. 302). Vision is the most important source of information for this purpose in sighted organisms. But the task of interpreting the spatial patterns of light entering the eyes is astonishingly difficult—so difficult, in fact, that a staggering 40% of the human cortex is estimated to be involved in visual processing in one way or another (Van Essen, Anderson, & Felleman, 1992; Van Essen & Gallant, 1994).

Successful visual processing in higher animals is the result of a great many mechanisms acting in concert to analyze the stream of visual inputs. The use of many parallel mechanisms is important:

> Why use multiple mechanisms when a single one will suffice on computational grounds? There are at least two reasons. First, by using multiple strategies for any one problem, the system can get away with each of them being relatively crude and, therefore, easy to implement in real neural hardware. As an analogy, consider two drunks, neither of whom can walk unsupported but, by leaning on each other, they manage to stagger along towards their goal! Second, the simultaneous use of multiple parallel short-cuts allows more rapid processing of images and a greater tolerance for noise than what would be possible with a single sophisticated algorithm (Ramachandran, 1990, p. 23).

In this chapter, I have focused on only two of the many mechanisms likely to be used by the visual system. Both mechanisms are based on exploiting patterns of visual texture, and both produce approximate results. One of the mechanisms seeks to segregate the visual input into regions. The second mechanism seeks to extract the rough shapes of surfaces based on patterns of texture.

If a visual region is found to contain texture, a reasonable default assumption is that what gave rise to the region in the image was a surface in the scene, or a collection of objects (bushes, rocks, etc.) that as a first approximation appears to constitute an implicit surface. This implies that a texture-based analysis is inherently a surface-oriented analysis. Thus, segmenting an input based on texture can provide an estimate of where surfaces are likely to be located in the image.

Most surfaces in the world are neither flat nor always viewed from the front, so the system must be able to cope with textures that are systematically distorted due to 3-D effects. At the same time, for a visual system attempting to estimate the spatial layout of surfaces in a scene, it would be advantageous to exploit texture distortions as a way of estimating surface orientations, rather than throwing away the information contained in the distortions. Unfortunately, most existing models of each type of process make assumptions that prevent them from being used on general visual inputs. Most models of texture-based segmentation assume that the visual input contains only flat textures viewed from the front; most models of texture-based shape estimation assume that the input contains only one surface.

A few researchers have developed models designed to overcome each of the limitations separately. Only two models (Krumm & Shafer, 1994, 1995; Moerdler & Kender, 1987) have addressed the question of how to combine the two processes of segmentation and shape estimation into a system that can handle inputs containing multiple regions of slanted, textured surfaces. However, both models are engineering efforts and neither is intended as a model of biological visual processing. A biologically-reasonable model of combined texture-based segmentation and shape estimation has remained an elusive goal.

The model I present in the remainder of this dissertation is an attempt at solving this problem. It begins with the outputs of complex cells in the primary visual cortex. Work on models of both texture-based segmentation and shape estimation has, in recent years, converged on the idea of using spatial filters resembling cortical cells as the initial processing stage. The filter outputs can be used as a basis for detecting different regions in the input for segmentation, and for detecting shape-induced distortions in texture for shape estimation.

The shape estimation mechanism used here is based on relatively simple principles developed by Sakai and Finkel (1995). The approach relies on tracking changes in a certain quantity, the average peak frequency, across a region of texture in an image as a way of estimating changes in the shape of a surface. The method assumes that surface textures are statistically homogeneous—that is, everywhere the same—and currently has the limitation that it works only with orthographic image projections. The mechanism is described in detail in Chapter 4.

Unlike the majority of existing texture-based segmentation models, the segmentation mechanism used here does not assume that textures in the input image are flat and viewed from the front; it can work equally well with slanted, curved and distorted textures, which is an essential quality for a model of visual processing of natural scenes. The mechanism is based on an approach developed by Lee (1993, 1995; Lee et al., 1991, 1992) and is described in detail in Chapter 5.

Although the shape estimation mechanism itself only works within a single region of texture, it is combined in Chapter 6 with the segmentation mechanism to allow the complete system to handle inputs containing multiple regions.

Chapter 3

A Model of Complex Cells in the Primary Visual Cortex

The model of spatial layout processing developed in this dissertation involves an architecture that can be broadly divided into three components: a model of cortical complex cells, a model of texture-based estimation of surface form, and a model of texture-based segmentation. This chapter describes the first component, a computational model of V1 complex cells.

The early stages of visual processing in the brain can be conceptualized as consisting of a bank of neural analyzers, each of which looks at a tiny part of the world through the small window that is its receptive field. Each such neural circuit is tuned to respond best when the stimulus falling within the receptive field has certain characteristics such as degree of contrast, color, or spatial organization. The last characteristic is especially important for purposes of spatial vision: each neural circuit acts as a filter that produces a signal indicating how well the local image structure resembles the receptive field profile. Because of the duality between spatial and spatial-frequency descriptions, the local image structure can be described equivalently in spatial-frequency terms. That is, a cortical cell produces a response when the image structure contains a certain range of spatial frequencies and orientations.

The role of the processing component described in this chapter is to compute a representation that resembles, in a highly simplified way, the transformation of a visual image performed by one type of filter, the cortical complex cell. The output of this part of the overall architecture is a set of numbers, each number representing a particular neuron's steady-state signal in response to a given visual input. The model of complex cells used here is based on combining the outputs of multiple simple cells, so the simulation also encompasses a model of simple cells.

The rest of the overall architecture relies on the outputs of the simulated complex cells, so it is important to use an adequate and well-grounded model of the responses of these neurons. Otherwise, the rest of the architecture may be developed under false assumptions about their properties. The model used here is based on a combination of two lines of research into the neurophysiology of V1 neurons: the *contrast-normalization model* of cortical simple and complex cells, and the *Gaussian derivative model* of simple cell receptive fields. The two models are complementary; each offers a computational theory of a specific aspect of the structure and function of cortical cells.

The particular variety of complex cell being modeled here is the most ordinary type, distinguished from various other known types of complex cells such as hypercomplex cells and end-stopped cells (e.g., Heitger, Rosenthaler, von der Heydt, Peterhans, & Kübler, 1992; Hubel, 1988; Zeki, 1993). This part of the overall architecture is generic and not task-specific; that is, although the outputs of the simulated complex cells are used for texture-based visual processing, there is nothing specific to detecting texture in their design. Presumably the same neural responses could be used as a starting point for other visual processes in a more elaborate model.

This chapter is organized as follows. Section 3.1 describes the contrast-normalization model of simple and complex cell responses, while Section 3.2 summarizes the Gaussian derivative model of simple cell receptive fields. Section 3.3 describes my software simulation of these simple and complex cells, including the justifications for the choices of various parameters. Section 3.4 then describes experimental testing of the simulation. Finally, Section 3.5 summarizes the key points of this chapter and discusses some of the novel aspects of this simulation as compared to other models.

3.1 The Contrast-Normalization Model of Simple and Complex Cells in Cortical Area V1

Appendices A.4 and A.5 present a summary of some of the basic characteristics of the simple and complex cells located in area V1, the cortical area that receives (via the LGN) the bulk of the signals produced by the retinas. Heeger has developed an elegant and plausible computational model of the simple and complex cells in V1 (Heeger & Adelson, 1989; Heeger, 1990, 1991). This *contrast-normalization model* appears to account for many empirical results about the behavior of these cortical neurons (Heeger, 1992a, 1992b, 1993; Tolhurst & Heeger, 1997a, 1997b). It is also straightforward to implement in a simulation.

The full contrast-normalization model includes a time dimension, but for the purposes of this research I will focus only on the two dimensions of space and ignore time. This is a significant simplification, because the receptive field properties of cortical cells often change over a short period of time—indeed, this temporal behavior is at the crux of what allows a cell to detect motion (Adelson & Bergen, 1985; Emerson, 1997; Emerson & Huang, 1997; Heeger, 1987; Gaska, Jacobson, Chen, & Pollen, 1994; Simoncelli & Heeger, 1997; van Santen & Sperling, 1985; Watson & Ahumada, 1985; Young & Lesperance, 1993). In the present research, all of the stimuli are static texture patterns, for which motion detection is irrelevant.

3.1.1 The Model of Simple Cells

The key difference between contrast-normalized complex cells and other models lies in the characteristics of the underlying simple cells. There are four components to a contrast-normalized simple cell, with the first three being common to many other models.

The first component, the element that is responsible for detecting the distribution of light intensities in a stimulus, is a linear receptive field operator. It generates a response based on the degree of match between the operator's two-dimensional receptive field pattern, given by some function $F(x,y)$, and the portion of the retinal image falling within the receptive field. An operator is defined as *linear* if its output in response to a sum of two inputs is equal to the sum of its outputs to each input individually. The response is calculated as a weighted sum of the stimulus image $I(x,y)$ and the receptive field's positively weighted (excitatory) and negatively weighted (inhibitory) subregions. This can be expressed mathematically as cross-correlation,

$$L(x_0, y_0) = \iint_{-\infty}^{\infty} F(x + x_0, y + y_0) I(x, y) dx dy, \tag{3.1}$$

where (x_0, y_0) is the center of the receptive field.

In a typical computational model of cortical processing, a copy of the operator is assumed to be centered at each location in the visual field. The profile of the receptive field weighting function, $F(x,y)$, determines the operator's selectivity for spatial frequency and orientation. Different shapes of $F(x,y)$ lead to different properties. For example, a receptive field with closely spaced excitatory and inhibitory subregions will be selective for higher spatial frequencies than will a field that has larger subregions spaced farther apart. The different parameters affecting receptive field properties are discussed in more detail in Section 3.2 below, in the context of a specific model (the Gaussian derivative).

A purely linear operator can generate a signal ranging over both positive and negative values, but real cortical neurons cannot signal a negative response. The latter is a consequence of the fact that cortical neurons have low rates of spontaneous activity. (If the neurons had a moderate baseline activity rate, then they could signal negative quantities by firing less than the baseline, and positive quantities by firing more than the baseline.)

To model the positive-only quality of cortical cell responses, the second component of the contrast-normalization model uses *half-wave rectification*. This produces an output value of zero for those inputs that would otherwise result in a negative output value. Because there must still be a way for the brain to detect dark stimuli as well as bright stimuli, the model assumes that there are two half-wave rectified cells present at every image location: an *on-center* cell to encode the positive half of a signal, and an *off-center* cell with a complementary arrangement of its receptive field—where the excitatory and inhibitory subregions of the field are reversed—to encode the negative half of the signal.

The third component is squaring the result of the half-wave rectification stage. This makes the cell outputs more consistent with experimental results (Heeger, 1992a). Half-wave rectification followed by squaring are expressed as *half-squaring*:

$$R(x,y) = \{\max[L(x,y),0]\}^2 \tag{3.2}$$

where $L(x,y)$ is the response of a purely linear operator at some point x,y, and $R(x,y)$ is the resulting half-squared response. The use of a squaring component has important implications for how experimental data about the properties of cortical cells must be interpreted; more about this issue in Section 3.2.2.

Finally, the fourth component of the simple cell model is a *divisive normalization* stage after the half-squaring stage. In this stage, the cell's response to a stimulus is divided by a quantity proportional to the pooled activity of a large number of neighboring neurons. This set of neighboring neurons includes cells tuned to a range of spatial frequencies and orientations. The effect is that the activity of neighboring cells can partially inhibit the activity of a given cell in such a way that the ratio of the cell's responses to two stimuli is mostly independent of the contrast of the stimuli. It is this fourth component that gives the model its essential contrast normalization quality, and allows it to account for a number of experimentally-observed properties of real cells (Akutsu & Legge, 1995; Heeger, 1992b; Tolhurst & Heeger, 1997a).

All together, the response of a simple cell is defined in the model as:

$$S(x,y\,;f,\theta,\phi) = k_s \frac{R(x,y\,;f,\theta,\phi)}{\sigma^2 + \frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x,y\,;f,\theta,\phi)} \tag{3.3}$$

where σ is a constant; k_s is a scaling parameter; $R(x,y\,;f,\theta,\phi)$ is the half-squared output of a linear receptive field operator with preferred spatial frequency f, orientation θ, and phase ϕ; and the summation in the denominator, $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x,y\,;f,\theta,\phi)$, is taken over a large number of receptive field operators with different tuning properties and neighboring locations i. The sum includes the current cell; that is, each neuron participates

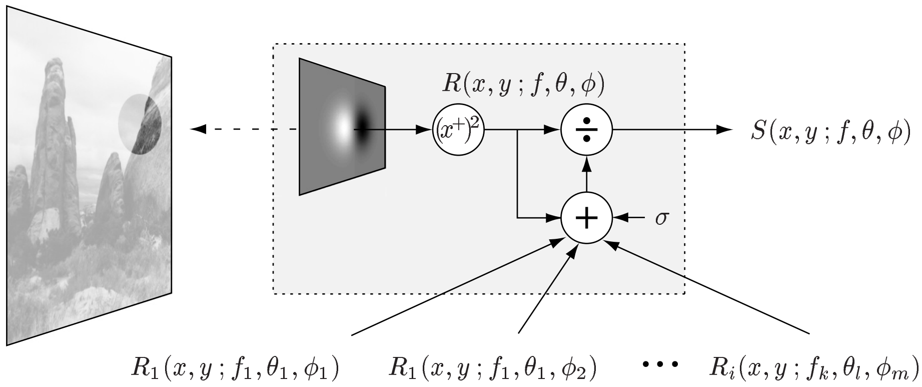

$R(x,y\,;f,\theta,\phi)$
$(x^{+})^{2}$
$\div$
$S(x,y\,;f,\theta,\phi)$
$+$
σ
$R_1(x,y\,;f_1,\theta_1,\phi_1)$
$R_1(x,y\,;f_1,\theta_1,\phi_2)$
$\cdots$
$R_i(x,y\,;f_k,\theta_l,\phi_m)$

Figure 3.1: Sketch of the components of a contrast-normalized model simple cell. The first component produces a signal calculated from the weighted sum of the linear receptive field and the portion of the stimulus that falls in the visual region covered by the receptive field. The receptive field establishes the simple cell's preferred frequency f, orientation angle θ, and phase angle ϕ. The second and third components are half-wave rectification followed by squaring, collectively known as half-squaring and abbreviated here as $(x^{+})^{2}$. This results in the nonlinear output $R(x,y\,;f,\theta,\phi)$ for a cell at a given location x,y in the retinal image. The fourth component is divisive normalization by a large pool of other receptive field operators $R_i(x,y\,;f_k,\theta_l,\phi_m)$ having different spatial frequencies f_k, orientations θ_l, and phases angles ϕ_m, at neighboring locations i surrounding x,y. The normalization sum includes the current cell.

in its own divisive normalization process. The use of $^1/_4$ in the summation term arises because the model uses simple cells at four different phases (that is, the index ϕ ranges in value from one to four). The components of the model are illustrated in Figure 3.1. The meanings and values of the different parameters are discussed in Section 3.1.4 below.

The value $S(x,y\,;f,\theta,\phi)$ is intended to represent the equivalent of a neuron's post-stimulus time histogram (PSTH), a measure of a neuron's firing rate. For the purposes of the present research, the values can be interpreted in a qualitative manner: high values indicate strongly activated neurons, low values indicate weakly activated neurons.

3.1.2 The Model of Complex Cells

Cortical complex cells bear many similarities to simple cells. For example, they are selective for similar ranges of spatial frequencies and orientations (De Valois & De Valois, 1990). However, they possess one property not shared by simple cells: a complex cell will respond to a bright or dark stimulus wherever it may appear in its receptive field—in effect, it abstracts away relative position (phase) within the receptive field. Experimental evidence suggests that this functional difference between the cell types represents a true dichotomy and not simply the ends of a continuum (De Valois & De Valois, 1990).

One way to achieve this insensitivity to phase is to combine the outputs of multiple simple cells whose receptive fields overlap in space. This is the approach taken in the contrast-normalization model. This type of arrangement was first put forth as an anatomical theory by Hubel and Wiesel (1959, 1962). Since the time of Hubel and Wiesel's original experiments, anatomical and physiological evidence has been uncovered suggesting that some complex cells of primate V1 actually receive direct input from LGN neurons (De Valois & De Valois, 1990). This implies that not all complex cells draw their inputs exclusively from simple cells. Despite this, it remains viable to regard complex cells *functionally* as deriving their inputs directly from simple cells.

In Heeger's (1992a, 1992b) model, four simple cells serve as input to one complex cell. The four simple cell receptive fields are assumed to be coincident in space, so that all four cover the same region on the retinal image, but with phases differing by 90°. The outputs of the four cells are averaged to produce a single complex cell output according to the equation

$$C(x,y\,;f,\theta) = \tfrac{1}{4}\sum_{\phi=1}^{4} S(x,y\,;f,\theta,\phi), \tag{3.4}$$

where $S(x,y\,;f,\theta,\phi)$ is the normalized response of a simple cell tuned to a preferred spatial frequency f, orientation θ, and phase ϕ. The complex cell model is illustrated in Figure 3.2 on the following page.

The average of the four half-squared mechanisms in the model is equivalent to the average of just two full-squared mechanisms whose phases differ by 90° from each other. As mentioned above, real cortical neurons cannot represent negative outputs, so each such full mechanism must be represented using two cells, the two being on-center and off-center partners of each other. This gives rise to the sum of four cells used in the model. The diagram in the lower left of Figure 3.2 on the next page provides an example of four receptive field profiles that have this arrangement.

3.1.3 Requirements for Proper Normalization

The goal of contrast normalization is to produce a set of mechanisms that collectively respond to visual patterns in a way that is roughly invariant with respect to stimulus contrast, despite that the responses of individual mechanisms saturate at high contrasts (Heeger, 1991). This is meant to account for our ability to perceive patterns as having similar spatial frequencies and orientations even if they have different contrasts (Albrecht & Hamilton, 1982; Graham, 1989). For the model to work properly, the receptive field operators used in the implementation of the simple and complex cell models must satisfy certain conditions.

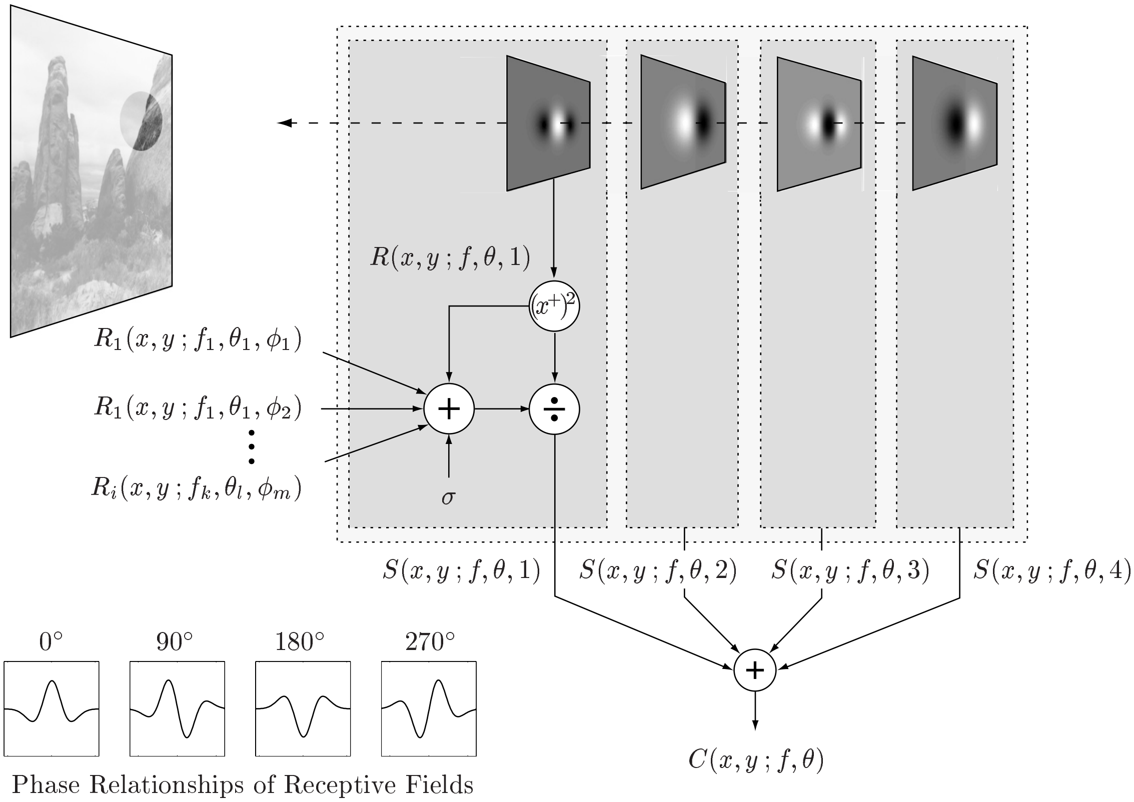

$R(x, y\,; f, \theta, 1)$
$(x^+)^2$
$R_1(x, y\,; f_1, \theta_1, \phi_1)$
$R_1(x, y\,; f_1, \theta_1, \phi_2)$
$R_i(x, y\,; f_k, \theta_l, \phi_m)$
$+$
$\div$
σ
$S(x, y\,; f, \theta, 1)$
$S(x, y\,; f, \theta, 2)$
$S(x, y\,; f, \theta, 3)$
$S(x, y\,; f, \theta, 4)$
$+$
$C(x, y\,; f, \theta)$
0°
90°
180°
270°
Phase Relationships of Receptive Fields

Figure 3.2: Sketch of the components of a contrast-normalized complex cell. The output of a complex cell with a preferred frequency f and orientation angle θ is produced by the sum of four contrast-normalized simple cells. Each simple cell receptive field has a different phase $\phi = \{1, 2, 3, 4\}$. An example of this kind of arrangement is shown in the lower left of the figure.

Following common practice, an underlying assumption in the model is that information about a stimulus is represented by the relative responses of a collection of neurons. When the stimulus contrast changes, the responses of individual neurons change, but as long as the ratio of their responses remains the same, the collective result will be independent of stimulus contrast.

The numerator in the simple cell equation (Equation 3.3 on page 46) is proportional to the squared output of the underlying linear receptive field. To insure that the ratio of a simple cell's outputs to two stimuli is independent of stimulus contrast, the summation term in the denominator of the equation must be proportional to the squared stimulus contrast. This is achieved in the model by making the summation term be proportional to the Fourier energy of the stimulus (i.e., the squared magnitude of the Fourier transform of the stimulus). The reasoning behind this is involved, but the conditions that lead to this proportionality are worth elaborating because they have important implications for creating simulations of the model:

1. Recall that the four half-squared receptive field operators in the model are equivalent to two full-squared operators having only a phase difference of 90°. In the spatial-frequency domain, these two operators will have identical amplitude profiles. Suppose that the maximum amplitude of each operator is A. Then, in analogy to the mathematical identity $\sin^2 x + \cos^2 x = 1$, the sum of these two squared operators will equal A independent of the phase of the input. Thus, the two operators are complementary and act as *quadrature partners*, a term borrowed from signal analysis and optics. A collection of quadrature partners in which each pair is tuned to a different spatial frequency and orientation will respond with a magnitude of A at each point in the spectrum.

2. For the next step, another concept is needed from signal analysis, that of the energy of a signal. For signal function $Z(x)$, the energy over some interval $a \leq x \leq b$ is defined as $\int_a^b |Z(x)|^2 dx$. The Parseval-Rayleigh theorem (Brigham, 1974; Bracewell, 1986) provides the crucial relationship that $\int_{-\infty}^{\infty} |Z(x)|^2 dx = \int_{-\infty}^{\infty} |\mathcal{Z}(f)|^2 df$, where $\mathcal{Z}(f)$ is the Fourier transform of $Z(x)$. This theorem states that energy can be computed either by summing the energy per unit space ($|Z(x)|^2$) or else by summing the energy per unit frequency ($|\mathcal{Z}(f)|^2$).

3. The summation term $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)$ in the simple cell formula (Equation 3.3 on page 46) takes the sum of the squared responses of the receptive field operators at all phases over a set of frequencies, orientations, and locations. Because all phases are summed, this is equivalent to the sum of a pair of squared operators in quadrature at each frequency, orientation and location. By the reasoning above, the sum of a pair of quadrature partners is independent of phase—it is equivalent to the squared modulus $|\mathcal{Z}(f_k, \theta_l)|^2$ of some function at a frequency f_k and orientation θ_l. The rest of the summation $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)$ in effect takes the sum of $|\mathcal{Z}(f, \theta)|^2$ at many frequencies f and orientations θ. In other words, it computes an approximation to the sum of energy per unit frequency, which, by the Parseval-Rayleigh theorem, is equivalent to the sum of energy per unit space. Therefore, the summation term is *proportional to the energy of the Fourier spectrum of the stimulus* in a region of space and limited range of spatial frequencies and orientations (Heeger, 1987, 1991, 1992b).

A mechanism that sums the squared outputs of a pair of operators in quadrature is also known as an *energy mechanism* (Adelson & Bergen, 1985), a concept mentioned in Section 2.2.2. The requirements for normalization imply that the spatial-frequency spectrum should, as much as possible, be entirely covered by the collection of energy

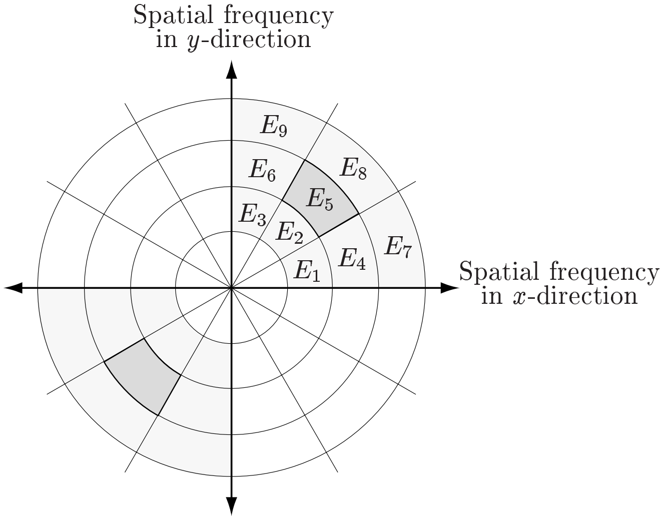

Spatial frequency in y-direction
Spatial frequency in x-direction
E_9
E_6
E_8
E_5
E_3
E_2
E_7
E_4
E_1

Figure 3.3: Desired tiling of the spatial-frequency domain by energy mechanisms (Heeger, 1991). The abscissa and ordinate of the graph represent spatial frequencies in the x and y directions, respectively. Each energy mechanism (the sum of squared outputs of a quadrature pair of receptive field operators) covers some region of the spatial-frequency domain, represented here by E_i. If the underlying receptive field operators are chosen properly, the energy mechanisms can be made to evenly tile the spatial-frequency domain as shown here (Heeger, 1991, 1992b). To normalize the output of a given mechanism, say the one tuned for region E_5 above, the normalization sum should include the outputs of mechanisms tuned to neighboring spatial frequencies and orientations E_1–E_9.

mechanisms. Figure 3.3 illustrates the idea. The abscissa and ordinate of the graph represent spatial frequencies in the x and y directions, respectively. If the entire spatial-frequency spectrum of a stimulus were plotted, it would show up as a set of points scattered across this graph. A portion of the graph area in Figure 3.3 has been subdivided into small schematic regions that each represent the amplitude response of one energy mechanism. Contrast normalization of each mechanism is achieved by dividing its output by the total "energy" at all orientations and nearby spatial frequencies. If, for example, all orientations are summed in a range of frequencies encompassing three octaves, then the normalization signal will be proportional to the Fourier energy of the stimulus in a three-octave wide annulus around the preferred spatial frequency of a given energy mechanism. (An octave is a two-to-one ratio.)

This feature of the model therefore imposes a requirement on the properties of the receptive fields: *the receptive field operators should tile the spatial-frequency spectrum* in a manner illustrated by Figure 3.3. That is, the sum of squared quadrature operators should add up to a constant value at all points in the spectrum, so that the result is proportional to the Fourier energy of the stimulus. How this is accomplished in the present simulation is discussed in Section 3.3.2. To foreshadow the results of that section, I note here that Gaussian derivative receptive fields have a drawback with respect to this aspect of the contrast-normalization model: a collection of Gaussian derivative receptive fields at multiple scales does not evenly tile the spatial-frequency spectrum. Nevertheless, it is possible to achieve an approximate tiling by making effective use of the scaling parameter k_s in the model of simple cells, and by exploiting the proportionality implied by the Parseval-Rayleigh theorem between the sum of receptive field operators and the Fourier energy of the stimulus.

3.1.4 Parameters of the Contrast-Normalization Model

The parameters of a complex cell $C(x, y\,; f, \theta)$ are the same as those of the underlying simple cells given by Equation 3.3 on page 46. There are five parameters in that equation, along with three more parameters that control the normalization sum in the denominator of the equation. How should values be chosen for them?

The Semisaturation Constant σ

The constant σ in Equation 3.3 is known as the semisaturation constant (Heeger, 1992b). In physiological experiments involving simple cells, it is set to the contrast of a grating pattern stimulus that evokes half the maximum attainable response of the cell being tested. As long as σ in the equation is nonzero, the response $S(x, y\,; f, \theta, \phi)$ will be a value between 0 and k_s. For this project, the value of σ is not critical because the nature of the simulations render its value arbitrary. Typical values range from 0.1 to 0.01 (Nestares & Heeger, 1997), but the purposes of this simulation, I set it to 1.0. The reason for this particular choice is clarified in Section 3.3.2 below.

The Scaling Factor k_s

In physiological experiments, the scaling factor k_s in Equation 3.3 can be used to rescale the maximum value of $S(x, y\,; f, \theta, \phi)$ to equal the experimentally-derived response value of a simple cell. For the simulations in this research, I used k_s to scale the simple cell outputs to have a value between 0 and 1 in a manner described in Section 3.3.2.

The Width of Spatial Averaging in the Normalization Sum

The locations i encompassed by the summation $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)$ in the denominator of Equation 3.3 comprise a local spatial neighborhood surrounding a given simple cell. In his formulation of contrast-normalized cells, Heeger left the size and shape of the spatial neighborhood unspecified (Heeger, 1992a, 1992b). For a realistic model, it should be set to a size and shape that is commensurate with neurophysiological data on the averaging neighborhoods of actual cortical cells (e.g., Hirsch & Gilbert, 1991).

In the present simulation, I set the spatial neighborhood to the size of the entire image. This is undoubtedly too large to be biologically realistic, but it has the advantage of allowing considerably faster computation than if the spatial average of a limited neighborhood about each cell is computed. The latter is equivalent to performing a convolution, which is an extremely slow operation for the kinds of large neighborhoods that would be required in a realistic version.

In an effort to verify that a neighborhood equal to the image size does not significantly alter the outputs compared to smaller neighborhoods, I performed a small experiment in which I tested the results of using neighborhoods of different widths. A comparison of the pattern of simulated cell activations for several different neighborhood sizes revealed that the outputs were substantially the same. Therefore, for the purposes of this research, I chose the simpler approach of setting the spatial averaging width of the normalization signal to the full width of the input image.

The Spatial-Frequency and Orientation Ranges in the Normalization Sum

What should be used for the values of frequencies f and orientations θ in the rest of the summation term $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)$ of Equation 3.3? The choices for these two parameters are determined by the requirements for proper normalization. The requirement is that, ideally, the summation term be approximately proportional to the Fourier energy of the stimulus. This ideal, however, is unlikely to be achieved in the nervous system. Rather than using the entire spectrum, a more realistic model is to have each cell be normalized within a limited portion of the spectrum by choosing a subset of spatial frequencies and orientations (see Figure 3.3 on page 51).

The easiest issue to settle is which particular orientations to use in the summation. A simple and reasonable choice is all possible orientations. Then each cell becomes normalized by the responses in an annulus of spatial frequencies, with a different annulus applying to cells tuned to different preferred spatial frequencies. It then remains only to choose how wide the annulus of frequencies should be.

In this implementation, I chose to use frequencies in a three-octave wide annulus centered on the preferred frequency of each cell. This is the normalization width used in some of the simulations implemented by Heeger (1993; Nestares & Heeger, 1997).

3.1.5 Discussion

The contrast-normalization model is an attempt to account for several functional properties of simple and complex cells in V1. These properties include nonlinearities in the cell responses with respect to spatial summation of stimuli, suppression of responses to a preferred stimulus by superimposed stimuli having other spatial frequencies and orientations, and saturation of responses at high contrasts (Heeger, 1991, 1992a, 1992b). The different components of the model, such as half-squaring and normalization, combine to produce behavior that is consistent with the properties of actual cortical neurons.

The primary difference between a simple and a complex cell is the latter's insensitivity to the relative position (phase) of a stimulus within its receptive field; in the model, this is accomplished by taking the sum of multiple simple cells whose receptive fields overlap on the retina. The full contrast-normalization model also accounts for sensitivity to temporal change (i.e., motion), but as mentioned above, in this work I restrict consideration only to static stimuli and ignore the time component.

The contrast-normalization model leaves open the question of what shapes the receptive fields should have. The two-dimensional receptive field profile of a simple cell determines its particular tuning properties with respect to the spatial, spatial-frequency, and orientation qualities of a stimulus. To implement a working simulation of complex cells using the contrast-normalization model, it is necessary to choose appropriate profiles for the simple cell receptive fields. The profiles I chose for the present work are discussed in detail in Section 3.2 below.

Motivations for Using the Contrast-Normalization Model

The ultimate goal of this research effort is to study the processing of texture by the visual system. In this context, the present choice of starting mechanism may seem odd. The contrast-normalization model of complex cells is a generic model, one that describes the basic behavior of all complex cells. There is nothing about the model that makes it specialized for responding to texture. Nevertheless, as will be seen in the experiment described below, these generic complex cells do respond well to visual texture.

In reality, cortical complex cells do not comprise a single, homogeneous class of neurons; one can classify them into subsets based on specific response properties. It is possible that some of these other types of complex cells may better serve as mechanisms for texture-based visual analysis (cf. Petkov & Kruizinga, 1997; von der Heydt, Peterhans, & Dürsteler, 1992). Despite this, there are a number of motivations for using a generic model of complex cells as a starting point:

- This class of neurons is one of the most basic and ubiquitous in the primary visual cortex. There is no question that these neurons exist, and they have been reported in hundreds of experiments.

- The properties of typical complex cells have been reasonably well-documented and accepted. Using complex cells thus puts the present research on firm empirical ground.

- In keeping with the research methodology adopted in this work (described in Section 1.2.3), it is more appropriate to begin with relatively basic mechanisms that may ultimately be found inadequate than to begin with more complicated mechanisms from the very beginning.

- Similar models of complex cells are used in the initial stages of many other models of texture-based visual processing, as summarized in Chapter 2. Using the same starting point improves one's ability to draw comparisons to other work.

Comparison of the Contrast-Normalization Model to Other Models of Complex Cells

There are some differences between the contrast-normalized complex cells and models of complex cells used by some other researchers. The type of model used by Bergen and Landy (1991), described in Section 2.2.2, is perhaps the most commonly-used alternative in research on texture-based visual processing. To simulate a complex cell, this model takes the sum of two linear, full-squared operators at each location, then pools the responses of many such pairs over a small spatial area, and finally applies a normalization operation to the result. This is similar to the contrast-normalization model, but a major difference is that it averages the responses of linear operators in a small spatial neighborhood centered at each location. The contrast-normalization model as formulated by Heeger (1992a, 1992b) simply takes the average of four simple cells with coincident receptive fields. Taking a spatial average, as is done in other models, is roughly equivalent to smoothing the responses of the simple cells in producing the complex cell responses. (In fact, this is exactly how Bergen and Landy implemented this component of their model.)

The implications of taking a spatial average of this sort are not clear for the present research. On the one hand, there is empirical research suggesting that some real complex cells may do this sort of averaging (De Valois, Thorell, & Albrecht, 1985; Emerson, Bergen, & Adelson, 1992). For instance, in a sample of monkey cortical cells, De Valois et al. (1985) found many complex cells that preferred periodic stimuli, implying their receptive fields contained many subregions—which in turn implies that they should have narrow frequency bandwidths—and yet they had broad spatial-frequency bandwidths. (Bandwidth is defined by Equation A.1 on page 326.) They hypothesized that these cells worked by summing the outputs of broadly-tuned simple cells in a spatial neighborhood. On the other hand, De Valois et al. also found complex cells with narrow spatial-frequency bandwidths. This implies that these particular cells did *not* sum their inputs over a neighborhood, because doing so would likely lead to broader spatial-frequency bandwidths. These experimental findings probably argue for differences within the class of complex cells: it may be that there exist subtypes of cells having different input pooling behaviors.

A model that could encompass both of these extremes—significant spatial averaging of simple cells versus highly localized averaging—could be created by extending the complex cell model of Equation 3.4 on page 48 to use more than four simple cells. In a recent model of motion-selective neurons, Simoncelli and Heeger (1997) used exactly this type of approach.

For the present research, I have chosen to stay with the contrast-normalization model as originally formulated (i.e, taking the sum of four simple cells with coincident receptive fields). The properties of this variant are simpler to analyze, and there is reasonably good support for this type of model from physiological research.

Realism of the Requirements for Specific Phase Relationships Between Simple Cell Receptive Fields

The contrast-normalization model, like many others, assumes the existence of simple cells in the visual cortex that are arranged with coincident receptive fields and a specific phase difference. Is there evidence for this kind of receptive field organization in the brain?

In fact, there is empirical work showing that some simple cells apparently do have receptive fields with this phase relationship. Pollen and Ronner (1981) examined the responses of adjacent simple cells recorded simultaneously from the same microelectrode in the visual cortex of cats. They located pairs that had the same preference for spatial frequency, orientation and direction of motion. Then they used cross-correlation techniques to estimate the spatial phase *difference* in the responses of the pairs. They found that the phase difference between cells in their sample ranged from 82° to 105°, or approximately 90° on average. They verified that the difference was not due to the receptive fields being shifted in space with respect to one another. Similar findings have been reported by others (Liu, Gaska, Jacobson, & Pollen, 1992). Additional studies have found pairs of cells with opposite polarities (i.e., differing in phase by 180°), which suggests that the contrast-normalization model's requirement for four simple cells of different polarity and symmetry is supported by physiological data (Pollen & Ronner, 1983).

It is worth noting, however, that real cortical cells do not only have purely even or odd symmetric receptive fields; their absolute phases tend to be distributed over all possible values (De Valois & De Valois, 1990). Although the implementation of complex cells developed in this chapter uses receptive fields with phases of exactly 0°, 90°, 180°, and 270°, there is nothing intrinsic to the model of complex cells that requires the use of those particular phases. All that the contrast-normalization model requires is that the difference in phases between component receptive fields is 90°.

3.2 The Gaussian Derivative Model of Simple Cell Receptive Fields

The contrast-normalization model of cortical simple cells does not specify the profiles that simple cell receptive fields should have (Heeger, 1991, 1992a, 1992b). The receptive fields of real cortical simple cells usually have an elongated form with separate excitatory and inhibitory subregions. But what, exactly, should the shapes of the profiles be? How many subregions should they have? What are the spatial-frequency properties of particular receptive field profiles? These are the kinds of questions that must be answered by a model of the receptive field shape.

For this research, I have chosen to use the *Gaussian derivative model* developed by Young (1978, 1985, 1986, 1991; Young & Lesperance, 1993). In addition to matching a number of features of real simple cells, this model's mathematical form has several convenient properties, and the theory behind it has already been elaborated by Young and others (e.g., Koenderink, Kappers, & van Doorn, 1992; Koenderink & van Doorn, 1992; Lindeberg, 1994).

3.2.1 Spatial Properties of the Gaussian Derivative Model

A given simple cell in cortical area V1 will increase or decrease its rate of firing neural spikes in response to a stimulus falling within its receptive field. The sensitivity profile of the receptive field determines the two-dimensional pattern of light contrast for which the cell is selective. Simple cells can also be selective for direction of motion, speed of motion, wavelength, and other properties, but for simplicity, these other properties are ignored in this research.

Hubel and Wiesel's (1959, 1968) classic observations of the spatial properties of receptive fields in V1 resulted in a model composed of two or three subregions. However, subsequent work led to the discovery that simple cells often have receptive fields composed of more than just two or three subregions—some cells have six or more (De Valois et al., 1985; Kulikowski & Bishop, 1981; Young, 1985). This sparked much speculation about how receptive field shapes can be described mathematically.

Different functional forms make different predictions about the precise shape of a receptive field and therefore its tuning properties in the space and spatial-frequency domains. The Gaussian derivative model formulated by Young (1985) begins with the product of two Gaussian functions in orthogonal directions:

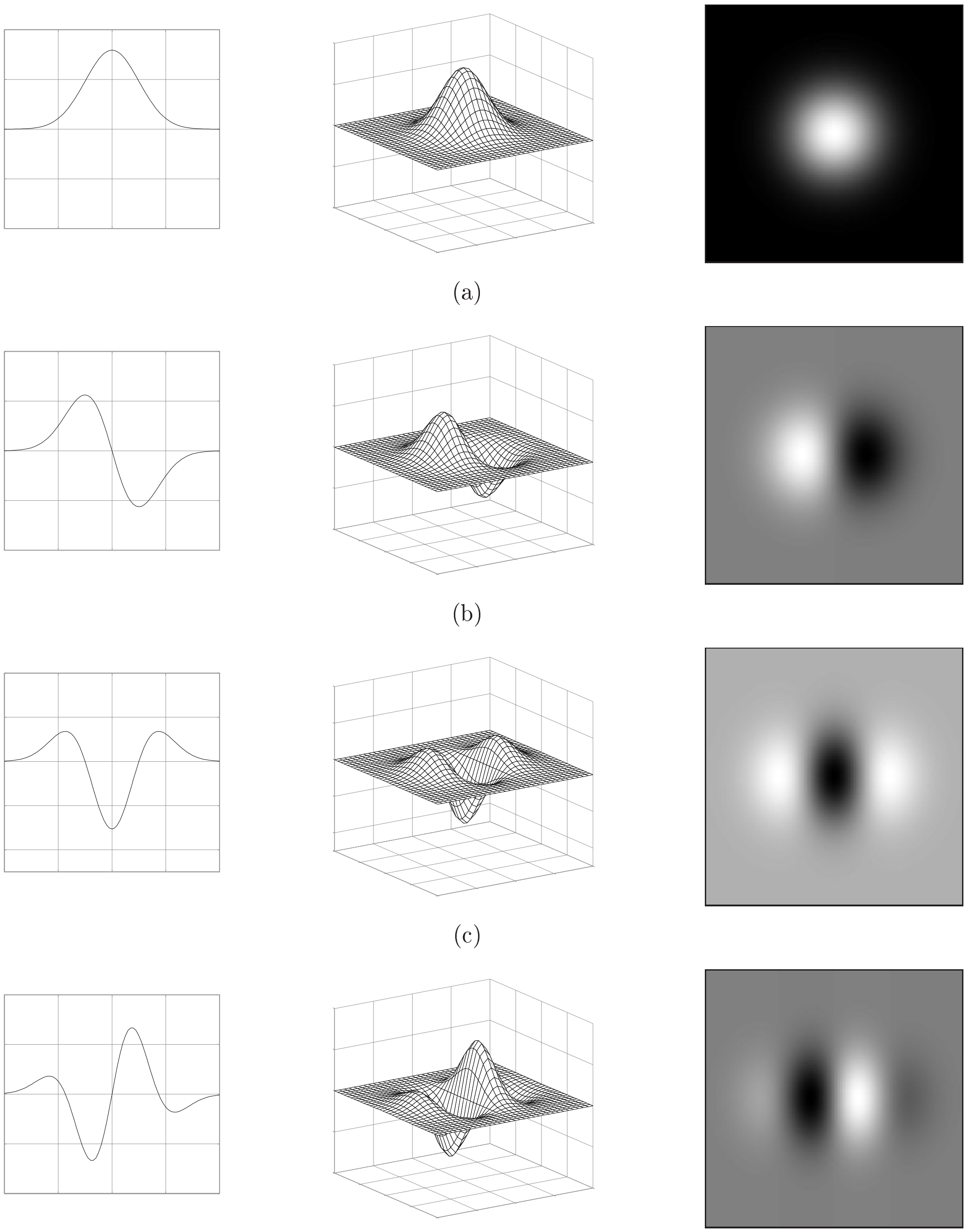
$$\begin{aligned} G(x, y\,; \sigma_x, \sigma_y) &= G(x\,; \sigma_x) G(y\,; \sigma_y) \\ &= \frac{1}{\sigma_x \sqrt{2\pi}}\, e^{-x^2/(2\sigma_x^2)} \times \frac{1}{\sigma_y \sqrt{2\pi}}\, e^{-y^2/(2\sigma_y^2)} \\ &= \frac{1}{\sigma_x \sigma_y 2\pi}\, e^{-x^2/(2\sigma_x^2) - y^2/(2\sigma_y^2)} \end{aligned} \tag{3.5}$$

where σ_x determines the spread of the function along the x-axis, σ_y determines the spread along the y-axis, and x and y are distances in space from a center point. This function has the shape of a “hump” when plotted against values of the Cartesian coordinates x and y, as illustrated in Figure 3.4 on the following page. By taking the derivative of this equation along one axis (conventionally the x-axis), a new feature is introduced: positive and negative lobes in the direction of the derivative, as illustrated in Figure 3.4(b–d). Young (1978) discovered the remarkable fact that when suitable values of the derivative order and σ's are chosen, the Gaussian derivative function describes a profile that can closely approximate the receptive field sensitivity profile of a cortical simple cell.

More precisely, the Gaussian derivative model of a receptive field is the two-dimensional function given by

$$\begin{aligned} G_n(x, y\,; \sigma_x, \sigma_y) &= G_n(x\,; \sigma_x) G(y\,; \sigma_y) \\ &= \left[\frac{d^n}{dx^n} G(x\,; \sigma_x) \right] G(y\,; \sigma_y) \\ &= \left[\frac{d^n}{dx^n} \frac{1}{\sigma_x \sqrt{2\pi}}\, e^{-x^2/(2\sigma_x^2)} \right] \frac{1}{\sigma_y \sqrt{2\pi}}\, e^{-y^2/(2\sigma_y^2)} \\ &= \frac{1}{\sigma_x \sigma_y 2\pi} e^{-y^2/(2\sigma_y^2)} \frac{d^n}{dx^n} e^{-x^2/(2\sigma_x^2)}. \end{aligned} \tag{3.6}$$

This describes a function that is centered at $(0, 0)$ in the x-y plane, and with the major and minor axes parallel to the coordinate axes. To allow for arbitrary orientations of the Gaussian derivative function in Cartesian coordinates, the coordinate axes of can be



(a)

(b)

(c)

(d)

Figure 3.4: The basic idea behind the Gaussian derivative model of receptive field profiles. Each row shows graphs of a Gaussian derivative of different order n but otherwise identical values of $\sigma_x = 1$, $\sigma_y = 1$ and $\theta = 1$. The left-hand column shows cross-sections along the x-axis, the middle column shows three-dimensional plots of $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$ against x and y, and the right-hand column depicts intensity plots of the functions $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$, with higher function values represented by brighter pixels. (a) A basic Gaussian function in two-dimensions. (b) The first derivative of a Gaussian in the x direction multiplied by a plain Gaussian in the y direction. (c) The second derivative of a Gaussian in x multiplied by a plain Gaussian in y. (d) The third derivative.

rotated by substituting the x and y parameters with

$$\begin{aligned} x_r &= x\cos\theta - y\sin\theta \\ y_r &= x\sin\theta + y\cos\theta \end{aligned} \tag{3.7}$$

where x_r and y_r are the coordinates in the rotated axes and θ is a counter-clockwise rotation, with respect to the x-axis, of the receptive field. It is also convenient to provide a magnitude scaling factor, k_n. The overall form of the Gaussian derivative function then becomes:

$$G_n(x, y\,; \sigma_x, \sigma_y, \theta) = k_n \frac{1}{\sigma_x \sigma_y 2\pi} e^{-y_r^2/(2\sigma_y^2)} \frac{d^n}{dx_r^n} e^{-x_r^2/(2\sigma_x^2)} \tag{3.8}$$

This function has five parameters: k_n is the magnitude scaling factor; n is the derivative order; σ_x and σ_y are the spatial variances in the x_r and y_r directions, respectively; and θ is the angle of rotation of the coordinates on which the function is given with respect to the original coordinate system.

The function $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$ can be substituted for $L(x, y)$ in Equation 3.2 to build a contrast-normalized simple cell. Gaussian derivatives G_n with different parameters can be used to create simple cells with different spatial-frequency and orientation tunings. For present purposes, the following are several useful properties of the Gaussian derivative receptive field model (Young, 1985):

- The total number of subregions (excitatory or inhibitory) for a given Gaussian derivative is always one more than the order of the derivative. For example, G_2 has three subregions and G_3 has four subregions, as shown in Figure 3.4 on the page before. This qualitatively relates the form of a Gaussian derivative function to a corresponding receptive field profile.

- The even-numbered derivatives possess even symmetry about the y-axis, and the odd-numbered derivatives possess odd symmetry. Further, Gaussian derivatives have the convenient feature that the function of order $n + 1$ is an approximation to the quadrature partner of the function of order n. This allows easy construction of complex cells, as discussed in Section 3.2.3.

- The function G_n is *zero-balanced*: the integral of $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$ over all x and y equals zero. This is desirable for the receptive fields of contrast-normalized cells, because it means that the response of such a receptive field to an even illumination will be zero. This corresponds to the behavior of most cortical simple cells (De Valois & De Valois, 1990; Hubel, 1988).

Equation 3.8 is a two-dimensional function in space only. However, the Gaussian derivative model can easily be extended to describe receptive fields whose profiles are time-dependent, such as those of cells that are selective for direction of motion. This can be accomplished by multiplying Equation 3.8 by another Gaussian in time (Young & Lesperance, 1993), producing a three-dimensional function. Such a three-dimensional receptive field function will have a spatial sensitivity profile that changes across time, allowing it to detect movement across its receptive field (Adelson & Bergen, 1985). This is another example of the Gaussian derivative model's elegance.

3.2.2 Spatial-Frequency Properties of the Gaussian Derivative Model

The four most important spatial-frequency parameters of a receptive field are: (1) a cell's preferred spatial frequency; (2) its selectivity for spatial frequency, quantified in terms of its spatial-frequency bandwidth; (3) its preferred orientation; and (4) its selectivity for orientation, quantified in terms of its orientation bandwidth. (Appendix A.5 provides some background on this topic.) Creating a software simulation of cortical cells requires being able to match a receptive field model's parameters to these response parameters of actual cortical cells. This in turn requires transforming the space-domain model into the spatial-frequency domain by means of the Fourier transform, and deriving expressions relating the model's various parameters to the spatial-frequency quantities.

The Fourier transform of the Gaussian derivative function $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$ has a surprisingly simple form (Young, 1985):

$$\begin{aligned}\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta) &= \iint_{-\infty}^{\infty} G_n(x, y\,; \sigma_x, \sigma_y, \theta) e^{-j2\pi(ux+vy)} dx dy \\ &= k_n (j2\pi u)^n e^{-2\pi^2(\sigma_x^2 u^2 + \sigma_y^2 v^2)}. \qquad (3.9)\end{aligned}$$

In this equation, $\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta)$ represents the Fourier transform of $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$; n is the order of the derivative; j is the imaginary term $\sqrt{-1}$; σ_x and σ_y are the spread terms as before; k_n is the Gaussian derivative's scaling term as before; and u and v index the spatial frequencies in the directions of the x- and y-axes, respectively.

To ease the task of understanding how the parameters in the function above affect its spatial-frequency properties, it is useful to begin by making some straightforward simplifications. First, it is enough to focus on a receptive field with a preferred orientation of zero degrees (i.e., aligned perpendicular to the x-axis in the space domain). Other preferred orientations can be had by rotating $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$ through an angle θ; the other parameters are independent of the preferred orientation. Second, the imaginary term j in the equation above affects only the phase of the function, under the control of the derivative order n. Thus, it is generally enough to study the spatial-frequency amplitude spectrum $|\mathcal{G}_n|$, given by the modulus of $\mathcal{G}_n$.

The amplitude spectrum of $\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta)$ for $\theta = 0$ has the shape of two humps arranged symmetrically along the u-axis on either side of the origin. The parameters n, σ_x and σ_y affect the shapes and positions of these humps, and this in turn determines the preferred spatial frequency, spatial-frequency bandwidth, and orientation bandwidth of $\mathcal{G}_n$. Figure 3.5 on the following page illustrates the effects of changing the parameter values. To find the specific values of n, σ_x and σ_y that result in a desired preferred spatial frequency, frequency bandwidth, and orientation bandwidth, Equation 3.9 on the page before must be solved for appropriate relationships. The mathematical details of this are left to Appendix B.1; here I simply summarize the results that will be needed in Section 3.3 for the implementation of a simulation of cortical complex cells.

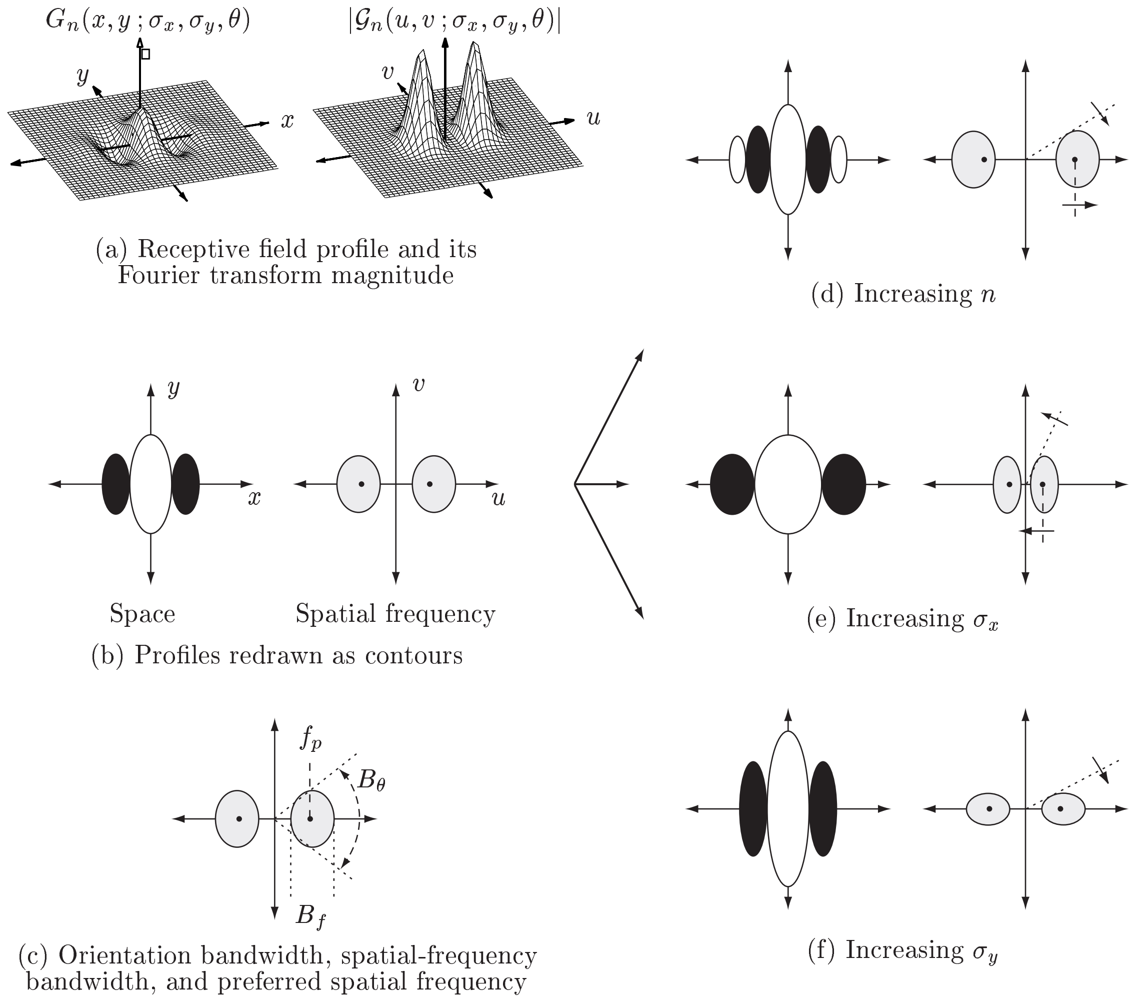
Equation 3.9 can be solved to yield a direct relationship between the order n of a Gaussian derivative receptive field, the preferred spatial frequency f_p, and σ_x:

$$\sigma_x = \frac{\sqrt{n}}{2\pi f_p}. \tag{3.10}$$

When modeling cortical neurons, the preferred spatial frequency f_p is usually determined by experimental data. Determining n, however, requires additional analysis.

The derivative order n of $\mathcal{G}_n$ is directly related to the spatial-frequency bandwidth, but it is not possible to obtain an analytical expression between the two. Fortunately, the relationship is one-to-one, so it is possible to determine n in a variety of ways. One approach is to use an approximation function developed by Young (1985), which states that $n \approx 2\pi/b^2$, where b is the full bandwidth measured at half-height in octaves of spatial frequency. Another approach is to exploit the fact that the spatial-frequency bandwidth depends solely on n and not on the other parameters of $\mathcal{G}_n$. This means that it is possible to set σ_x and σ_y to arbitrary values, then numerically calculate bandwidth versus n and tabulate the results, and finally use the resulting table to look up the value of n that gives the closest bandwidth to a desired bandwidth value. Young (1985) provides such a table; I have used the same approach to generate the values in Table 3.1.

Using the contrast-normalization model of simple and complex cells introduces an additional complication. As explained in Section 3.1.1, the underlying receptive field operator in a contrast-normalized cell is a half-squared function. This has an important implication for interpreting experimental data about cortical cells: it means that experiments which measure bandwidths are doing so on the *squared* responses of the underlying


$G_n(x, y\,; \sigma_x, \sigma_y, \theta)$
$|\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta)|$
y
x
v
u
Space
Spatial frequency
f_p
B_θ
B_f

(a) Receptive field profile and its Fourier transform magnitude

(b) Profiles redrawn as contours

(c) Orientation bandwidth, spatial-frequency bandwidth, and preferred spatial frequency

(d) Increasing n

(e) Increasing σ_x

(f) Increasing σ_y

Figure 3.5: Parameters of the Gaussian derivative receptive field. (a) Example Gaussian derivative function plotted in both the spatial and spatial-frequency domains (not to scale). The amplitude spectrum of the function has the form of two lobes spaced equally apart from the center. The receptive field shown has a preferred orientation of $\theta = 0$. (b) The spatial-frequency profile is typically plotted as the contour at one-half the maximum amplitude. The Gaussian derivative's frequency-domain cross-section is asymmetric, and so the peaks of the lobes, shown here as black dots, are slightly off-center within the contours. (c) The peaks of the lobes dictate the preferred spatial frequency, f_p. The spatial-frequency bandwidth B_f is the width of the frequency-domain contour. The orientation bandwidth B_θ, when measured using stimuli at the preferred frequency f_p, is the polar angle subtended by the outer edges of the profile at f_p. (d) The effect of increasing the derivative order n while holding the other parameters constant is to increase the separation between the lobes in the frequency domain. This moves the preferred frequency to a higher value and decreases the orientation bandwidth. (e) The effect of increasing σ_x while holding the other parameters constant is to narrow the widths of the lobes and decrease the separation between them. This simultaneously moves the preferred frequency to a lower value, produces shaper tuning for spatial frequency (i.e., narrower frequency bandwidth), and broader tuning for orientation. (f) The effect of increasing σ_y while holding the other parameters constant is to decrease the heights of the lobes in the frequency domain. This sharpens the sensitivity to orientation.

Table 3.1: Spatial-frequency bandwidths as a function of derivative order

n	b for $\mathcal{G}_n$	b for $\mathcal{G}_n^2$
1	2.590	1.765
2	1.765	1.224
3	1.423	0.993
4	1.224	0.858
5	1.091	0.765
6	0.993	0.698
7	0.918	0.646
8	0.858	0.604
9	0.808	0.569
10	0.765	0.539

b for $\mathcal{G}_n$: the spatial-frequency bandwidth as measured using the amplitude spectrum of a Gaussian derivative function.
b for $\mathcal{G}_n^2$: the spatial-frequency bandwidth as measured using the squared amplitude spectrum.

receptive field operators. In other words, *the physiological experiments must be interpreted as informing us about the squared spatial-frequency amplitude spectra of cortical cells, rather than the amplitude spectra themselves* (Heeger, 1992a). This in turn means that when calculating spatial-frequency and orientation bandwidths for purposes of a simulation, one must use the squared function $|\mathcal{G}_n|^2$ rather than $|\mathcal{G}_n|$ itself.

Using the squared amplitude $|\mathcal{G}_n|^2$ does not affect the preferred spatial frequency of the Gaussian derivative, but it does change the relationship between n and the spatial-frequency bandwidths. Table 3.1 includes a list of the frequency bandwidths for the squared amplitude function of the Gaussian derivative. The effect of using $|\mathcal{G}_n|^2$ is that a receptive field with a given n maps to a measured frequency bandwidth that is narrower than if $|\mathcal{G}_n|$ is used.

The final parameter, σ_y, is determined jointly by the desired orientation bandwidth, the frequency at which the orientation bandwidth is measured, and the values of n and σ_x. For the ordinary amplitude spectrum $|\mathcal{G}_n|$, the relationship is

$$\sigma_y = \sqrt{\frac{\ln 2 + n \ln(\cos \theta_{1/2})}{2\,\pi^2 f^2 \sin^2 \theta_{1/2}} + \sigma_x^2}. \tag{3.11}$$

For use with the squared amplitude spectrum $|\mathcal{G}_n|^2$, the $\ln 2$ term changes so that the function becomes

$$\sigma_y = \sqrt{\frac{\ln \sqrt{2} + n \ln(\cos \theta_{1/2})}{2\,\pi^2 f^2 \sin^2 \theta_{1/2}} + \sigma_x^2}. \tag{3.12}$$

Here, $\theta_{1/2}$ is the angle (measured from $u = 0$) at which the cell response drops to one-half of its maximum, and f is the frequency at which the orientation bandwidth is measured.

Usually, the frequency used to measure orientation bandwidth in neurophysiological experiments is the preferred frequency; that is, $f = f_p$.

3.2.3 Using the Gaussian Derivative in the Contrast-Normalization Model of Simple and Complex Cells

The contrast-normalization model places two requirements on the form of the underlying receptive fields. First, the four cells that are used as the primary inputs to a complex cell should have phase offsets of 90°. Second, the entire collection of receptive fields should have the property that their spatial-frequency tunings evenly tile the spatial-frequency domain. In this section, I describe how the Gaussian derivative receptive field model can be made to satisfy to these requirements.

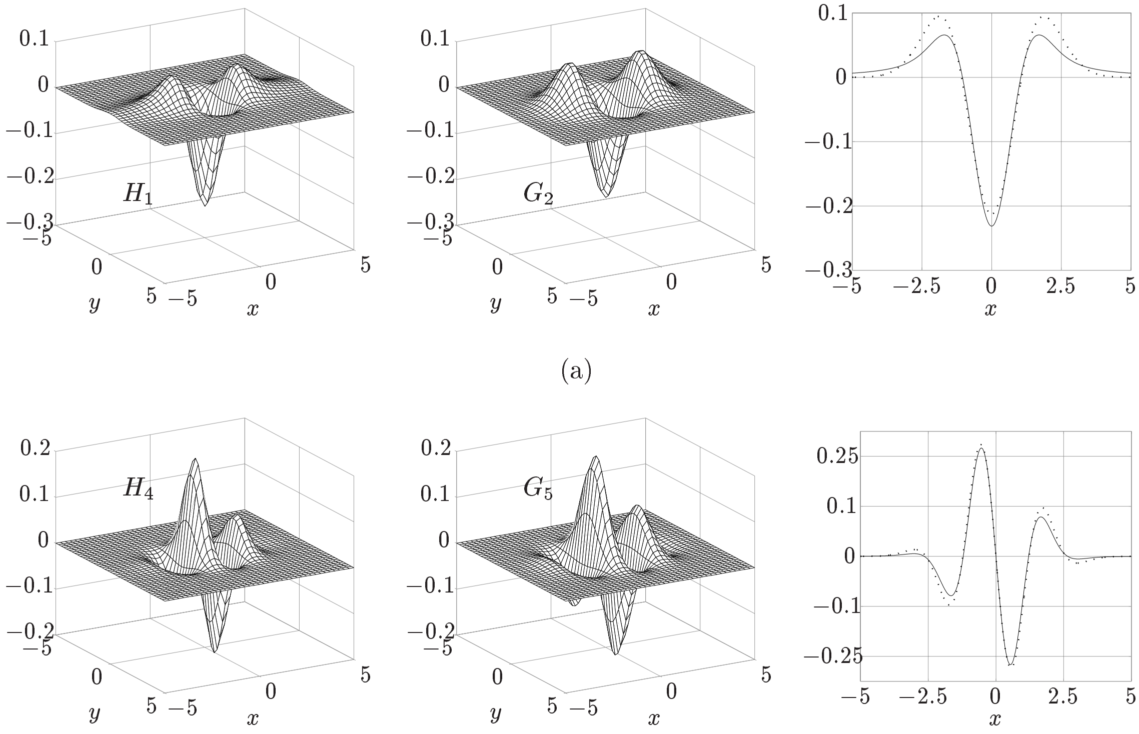
Obtaining Quadrature Pairs of Gaussian Derivative Receptive Fields

The simplest of the two requirements concerns the phase relationships of simple cells. The four underlying half-squared receptive field operators should combine to form the equivalent of two full-squared operators that are in quadrature. A quadrature pair is also known known as a *Hilbert transform pair*, because one member of the pair is the Hilbert transform of the other (Bracewell, 1986). As mentioned above, the Gaussian derivative functions have the property that the even-order derivatives have even symmetry, and the odd-order derivatives have odd symmetry. A number of researchers (e.g., Adelson & Bergen, 1985; Freeman & Adelson, 1991) have noted that the Hilbert transform of the nth-order Gaussian derivative function has a shape remarkably similar to the Gaussian derivative of order $n+1$. This suggests that it is possible to take the order $n+1$ derivative as an approximation to the true quadrature partner of the nth-order derivative.

The Hilbert transform of a function $F(x)$ is defined (Bracewell, 1986) as the Cauchy principal value of the integral

$$H(x) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{F(x')dx'}{x' - x}. \tag{3.13}$$

Figure 3.6 on the following page shows some examples of the true Hilbert transforms of Gaussian derivative functions G_n and their approximations using G_{n+1}. The functions plotted in the left-hand column of Figure 3.6 are the result of performing this integral on nth-order Gaussian derivative functions. The functions plotted in the middle column are the result of performing a least-squares numerical fit of the Gaussian derivative function of order $n + 1$ to a discrete representation of the Hilbert transform in the left-hand column. (Details of the procedure I used are presented in Appendix B.2.) The right-hand



(a)

(b)

Figure 3.6: Examples of approximations to true Hilbert transforms of Gaussian derivatives. The left column of each row displays a plot of the true Hilbert transform $H_n(x, y\,; \sigma_x, \sigma_y, \theta)$ of the Gaussian derivative function $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$. The middle column shows the closest-fitting Gaussian derivative function of order $n+1$ to the Hilbert transform function. The right column of each row shows a plot of the cross-section of each function. The solid line in the plots represents the actual Hilbert transform, and the dotted line represents the fitted function. (a) The Hilbert transform of $G_1(x, y\,; \sigma_x, \sigma_y, \theta)$ with $\sigma_x = 0.8$, $\sigma_y = 0.8$, and $\theta = 0$, and the closest-fitting approximation using $G_2(x, y\,; \sigma_x, \sigma_y, \theta)$. Parameters of fit: $\sigma_x = 1.0793$, $\sigma_y = 0.9434$ (b) The Hilbert transform of $G_4(x, y\,; \sigma_x, \sigma_y, \theta)$ with $\sigma_x = 0.8$, $\sigma_y = 0.8$, and $\theta = 0$, and the closest-fitting approximation using $G_5(x, y\,; \sigma_x, \sigma_y, \theta)$. Parameters of fit: $\sigma_x = 0.9038$, $\sigma_y = 0.9939$. For all cases, k_n was fixed at 1.0. All fits were performed using a nonlinear least-squares fitting procedure on discrete representations of the functions as described in Appendix B.2.

column shows the superimposed cross-sections of each function. The figure shows that the resemblance is not perfect, but it is close. The primary differences are in the ratios of the positive and negative regions and in the tails of the Hilbert transforms. The differences are most noticeable in the low order derivatives. For example, H_1 in Figure 3.6(a) has significant tails extending in the x-direction, but the Gaussian derivative approximation does not have this feature. The tails are progressively weaker in the Hilbert transforms of higher derivatives.

The implication of this similarity is that the four cells serving as input to a complex cell can be designed as follows. First, one chooses the desired preferred frequency and spatial-frequency bandwidth of one of the cells. By Table 3.1 on page 64, this establishes the derivative order n for G_n. The 90° partner of this cell would ideally be the Hilbert transform of the function; an approximation to this ideal can be found using the function G_{n+1}. This establishes the second receptive field. The remaining two receptive fields are simply the negatives of the first two. It then remains to find values for the parameters σ_x and σ_y for the partner of G_n.

The best approximation to the true quadrature partner of G_n would achieve the following goals:

1. The quadrature partner should have identical preferred spatial-frequency tuning, spatial-frequency bandwidth, and orientation bandwidth.

2. The sum of the squares of G_n and its partner H_n (i.e., $G_n^2 + H_n^2$) should be a smooth function in both the space domain and the spatial-frequency domain.

Unfortunately, it turns out that the first goal cannot be achieved completely. One difficulty is that the derivative order n directly determines the spatial-frequency bandwidth of a Gaussian derivative function; thus, G_{n+1} *cannot* truly have the same bandwidth as G_n. A second difficulty presents itself in practice. Given G_n and specific values for σ_x and the preferred spatial frequency, it should in theory be possible to use Equation 3.10 on page 62 to determine σ_x for the quadrature partner. The equation establishes a direct relationship between preferred frequency, n and σ_x. However, I have found that, although using Equation 3.10 in this way does indeed work, the sum of the resulting quadrature pair in the space domain is not always a smooth function: ripples sometimes appear in the shape of the sum.

To balance the different aspects of achieving a good approximation, I have found it more effective to perform a numerical fit of G_{n+1} to the space-domain representation of the Hilbert transform of G_n. The procedure I used is summarized in Appendix B.2. Using a fitting procedure in this manner creates approximate quadrature partners that produce fairly smooth sums $G_n^2 + H_n^2$ in the space domain. The main error introduced by this approach is that the preferred spatial frequency of the partner G_{n+1} is sometimes not identical to that of G_n.

Achieving Spatial-Frequency Tiling

The second requirement for the contrast-normalization model is that the receptive fields should have spatial-frequency tunings that evenly tile the spatial-frequency domain (Section 3.1.3). As mentioned already, it turns out to be difficult to make the Gaussian derivative model fulfill this goal.

To model a range of preferred frequencies and spatial-frequency bandwidths, it is necessary to use multiple derivative orders. Unfortunately, the maximum Fourier amplitude of the Gaussian derivative function changes with increasing derivative order, a consequence of the fact that each additional derivative multiplies the basic Fourier transform by an additional ramp function (see Equation 3.9 on page 61). If the responses in different frequency bands are normalized to have the same amplitudes in the spatial-frequency domain, then the amplitudes in the *space* domain end up different for filters tuned to different preferred frequencies. This is undesirable because then the amplitudes of the complex cells of different preferred frequencies are not on the same scale, and thus not comparable.

However, it is possible to achieve an *approximate* tiling that is adequate for the purposes of this simulation. The idea is based on exploiting the scaling parameters on the individual Gaussian derivatives (k_n in Equation 3.8 on page 60) as well as the simple cells (k_s in Equation 3.3 on page 46). A procedure for doing this is described in Section 3.3.2, in the context of the simulation architecture. The procedure must be performed numerically on the set of Gaussian derivatives actually implemented in the simulation, after other parameters have been established.

3.2.4 Discussion

The Gaussian derivative model described in this section provides a model of receptive fields that is in reasonable agreement with physiological findings. In this research, I use Gaussian derivatives of many different orders in order to simulate receptive fields having different sizes and tuning properties. Each different kind of receptive field is used to construct a particular kind of contrast-normalized complex cell, using the formulas described in Section 3.1.

Motivations for Using the Gaussian Derivative Model

Three models of simple cell receptive fields have been popular in research on cortical visual processing. The most popular has been the *Gabor* model (Daugman, 1985; Kulikowski, Marčelja, & Bishop, 1982; Jones & Palmer, 1987a, 1987b, 1987c; Marčelja, 1980; Watson, 1983). It has the form of a sine wave multiplied by a Gaussian function in each orthogonal direction. The primary motivations for using the Gabor are that it provides a fairly simple functional form, and that, in its full mathematical expression, it minimizes a certain quantity (the joint uncertainty in space and spatial frequency) that was initially thought to be significant for visual processing by cortical cells. However, empirical work suggests that the Gabor is not as accurate a model of real receptive fields as certain other models (Hawken & Parker, 1987; Young, 1985), and there is doubt about the utility of some of its theoretical properties as well (Stork & Wilson, 1990; Yang, 1992; Young & Lesperance, 1993). Despite this, it continues to be a popular model, especially in work concerning image representation (Lee, 1996; Watson, 1983; Xiong & Shafer, 1994).

Another popular model—actually, a class of models—is based on taking differences of multiple Gaussians (e.g., Hawken & Parker, 1987, 1991; Mesrobian & Skrzypek, 1995; Parker & Hawken, 1988; Wilson, 1991a, 1991b; Young, 1986). This difference-of-multiple-Gaussians model appears to be the most accurate currently available, at least for the cross-sections of monkey simple cell receptive fields (Hawken & Parker, 1987). It also suggests a fairly simple architecture for how LGN neuron outputs could be combined to produce the receptive fields of simple cells (Hawken & Parker, 1991). Its main disadvantage is that it requires an increasing number of parameters to describe receptive fields with increasing numbers of subregions, which makes it more difficult to use in modeling.

The third popular model of receptive field profiles has been the Gaussian derivative model. Certain aspects of the structure of cortical receptive fields (for example, the spacing between the zero-crossings in the profile) suggest that the Gaussian derivative model captures the structure better than the Gabor model (Young, 1985; Webster & De Valois, 1985). Despite this, some detailed neurophysiological studies in monkeys have found the Gaussian derivative to be less accurate than either the difference-of-multiple-Gaussians or the Gabor (Hawken & Parker, 1987). On the other hand, the Gaussian derivative model is actually closely related to the difference-of-multiple-Gaussians. Young (1985) has pointed out that, because a derivative is mathematically defined in terms of offset differences, the Gaussian derivative is therefore mathematically defined by a difference of multiple, offset Gaussian functions. In other words, the difference-of-multiple-Gaussians model can be considered to be an approximation to the discrete form of the Gaussian derivative model. Young has suggested that the former may be the brain's way of implementing an approximation to the derivatives of a Gaussian function (Young, 1985; Young & Lesperance, 1993). So although the former model may provide a more accurate fit to real cortical receptive fields, we can reasonably take the Gaussian derivative as being a mathematical idealization of the difference-of-multiple-Gaussians model.

Using the Gaussian Derivative Model of Receptive Fields

There are several parameters in the model that must be given values before a simulation can be implemented. The approach that I used to set these parameters is described in the next section.

3.3 Software Simulation of Simple and Complex Cells

The contrast-normalization model and the Gaussian derivative model have so far been described in the abstract. In order to create a simulation of visual processing using these models, it is necessary to settle on an architecture and choose values for the different parameters in the models. Many different questions need to be answered, such as: How many cells should be simulated? What should be their characteristics? How should they be organized in the simulation? In an effort to maintain a degree of biological relevance, I have attempted to adhere to a number of neurophysiological results while designing the simulation architecture. This section details the approach that I used.

The discussion that follows is divided into three parts. First, I describe the physiological and practical constraints on the model, and the particular set of parameter values I have chosen to use for the simulation. Second, I describe an approach for achieving approximate tiling of the spatial-frequency spectrum with Gaussian derivative receptive fields. Third, I summarize the basic architecture of the simulation and its implementation.

In the discussions that follow, I describe spatial-frequency quantities in units of cycles per degree of visual angle (c/deg). Physiological and psychophysical results on sensitivity to visual stimuli are usually reported in terms of cycles per degree, so using these units here is convenient for discussing tuning properties. However, in implementing a computer simulation of the spatial filtering performed by cortical cells, the units of visual angle must be translated into a specific number of pixels in a digital image. For all the simulations described in this research, I assume that cell tuning properties are expressed relative to an observer whose eyes are located approximately 92 cm in front of a computer screen having a resolution of 4 pixels/mm. This means that a visual angle of 1.0 deg equals 64 pixels in an input image.

3.3.1 Choosing Parameter Values Based on Physiological and Practical Constraints

The following sets of issues need to be resolved before implementing a simulation using the contrast-normalization and Gaussian derivative models:

1. What are the lowest and highest preferred spatial frequencies that should be included?

2. Which preferred frequencies in particular within this range should be included?

3. What should be the spatial-frequency bandwidth of each simulated cell?

4. What should be the orientation bandwidth of each simulated cell?

5. Which particular orientations should be included?

6. What should be the spatial arrangement of the model cells?

The choices of spatial-frequency tunings—the preferred spatial frequencies, frequency bandwidths and orientation bandwidths—drive the values of Gaussian derivative parameters n, σ_x and σ_y for each cell. The other issues—the sampling in frequency, orientation and space—drive the design of the rest of the architecture, such as the number of simulated cells required.

Unfortunately, the six issues listed above interact with each other, which makes choosing values difficult. For example, a cell's orientation bandwidth varies in proportion to its spatial-frequency bandwidth, and because orientation bandwidth determines the number of orientations that are needed in order to cover the range of possible orientations, spatial-frequency bandwidth indirectly affects orientation sampling. The relevant constraints for the current project are summarized in Table 3.2 on the following page. In the paragraphs that follow, I discuss each of these issues in turn. Where possible, I have tried to make choices informed by the relevant evidence from neurophysiology and psychophysics.

It is worth noting that available physiological data come almost entirely from research on animals and not humans. This means it is necessary to assume that the results carry over to the human visual system. I have tried as much as possible to emphasize data from research on monkeys, because their visual systems appear to be close in structure and property to the visual system of humans. Nevertheless, it is important to keep in mind that the data may not accurately reflect the human visual system.

Range of Preferred Spatial Frequencies

When measured using sine wave stimuli, simple and complex cells can be shown to exhibit a preference for a particular spatial frequency. This is the cell's *preferred spatial frequency.* Different cells respond best to different preferred frequencies.

De Valois, Albrecht, and Thorell (1982) performed quantitative analyses of the spatial-frequency tuning characteristics of simple and complex cells in V1 of macaque monkeys. They examined cells whose retinal eccentricities ranged from the foveal center to 5° away from the foveal center. They found that cells in their sample were tuned to preferred frequencies ranging from 0.5 c/deg to 16.0 c/deg. They also found that simple and complex cells had similar preferred frequency sensitivities, although complex cells were on average tuned to slightly higher frequencies than simple cells. A plot based on their results is shown in Figure 3.7 on page 73. Other studies have found similar ranges of preferred frequencies (Foster, Gaska, Nagler, & Pollen, 1985; Schiller, Finlay, & Volman, 1976).

Table 3.2: Parameters for the simulation and constraints affecting their values.

Parameter	Physiological Constraints	Simulation Constraints
Range of preferred spatial frequencies	• Lowest and highest values of preferred frequency found in V1 simple and complex cells • Negative correlation between preferred frequency and frequency bandwidth • Positive correlation between spatial-frequency bandwidth and orientation bandwidth	• Nyquist sampling limits on digital images used as inputs • Maximum number of orientations for a practical simulation
Sampling in spatial frequency	• Continuous sampling of spatial frequencies across the entire range of frequencies • Decreased sampling of highest frequencies with increasing eccentricity from foveal center	• Sufficient number of spatial-frequency bands for dense covering of frequency spectrum • Limited number of frequency bands for a practical simulation • Complexity of implementing varying sampling in spatial frequency
Spatial-frequency bandwidths	• Lowest and highest values of spatial-frequency bandwidths found in simple and complex cells • Negative correlation with preferred spatial frequency • Positive correlation with orientation bandwidth	• Sufficiently narrow spacing between frequency bands for adequate covering of spectrum • Limited number of frequency bands for a practical simulation • Limited number of orientations for a practical simulation
Orientation bandwidths	• Lowest and highest orientation bandwidths found in V1 simple and complex cells • Positive correlation with spatial-frequency bandwidth	• Limited number of orientations for a practical simulation
Sampling in orientation	• Continuous sampling of orientations over full 180° range	• Limited number of orientations for a practical simulation • Sufficiently narrow spacing between orientations for adequate coverage of spectrum
Sampling in space	• Extremely fine sampling of all parameters at every location in visual field	• Limited number of spatial samples (pixels) in digital images

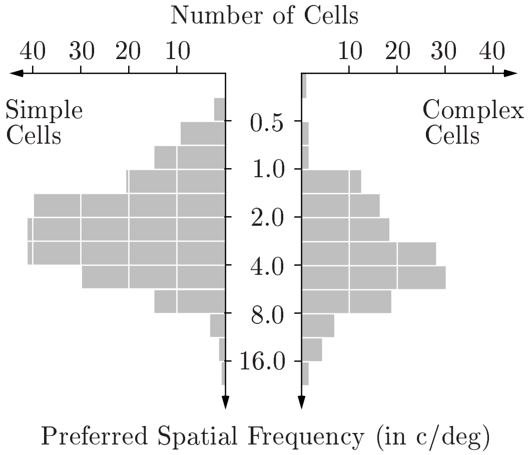


Figure 3.7: Distribution of the spatial-frequency sensitivities for a sample of over 370 simple and complex cells from the primary visual cortices of macaque monkeys. This distribution includes cells within 5° of the fovea. This graph shows that simple and complex cells have roughly the same range of preferred frequencies, with complex cells tending to prefer slightly higher frequencies on average. Data taken from De Valois and De Valois (1990), their Figure 4.17.

In humans it appears that there are cortical cells that can respond to frequencies higher than 16.0 c/deg. Several lines of evidence support this conclusion. First, the size of the eye is larger in humans than in monkeys; consequently a greater number of receptors on the retina is subtended by a given visual angle (Hawken & Parker, 1991). Second, psychophysical measurements of contrast sensitivity point to the need for cells with preferred tunings higher than 16.0 c/deg in order to explain the contrast sensitivity results (Watson, 1982). And third, under optimal conditions, human observers have been shown able to resolve frequencies as high as nearly 60 c/deg (De Valois & De Valois, 1990). Assuming that this ability is directly related to the tuning properties of simple and complex cells in the visual cortex, it implies that in the human visual system there are cells tuned to preferred frequencies of at least 32 c/deg and possibly higher. (Because cells have a certain breadth of response for spatial frequency, the highest perceivable frequency does not need to correspond to the highest preferred frequency of cortical neurons. For example, a cell with a peak at 32 c/deg and a bandwidth of 0.8 octaves would have a half-amplitude point at around 42 c/deg, and would show some response to even higher frequencies. Typical cell bandwidths range from 0.5 to 3.0 and higher, as discussed below.)

These results suggest that, for the purposes of simulating the simple and complex cells of the human visual system, a reasonable range of preferred frequencies would be 0.5 c/deg to about 32 c/deg and possibly higher. In this simulation, I have chosen the low and high preferred frequencies to be 1.0 c/deg and 22.6 c/deg, respectively. From preliminary tests, I determined that preferred frequencies lower than 1.0 c/deg are not useful for the kinds of input images used in this work. And although it would be desirable to include spatial frequencies higher than 22.6 c/deg, in practice I have found that higher frequencies are difficult to include in the present simulation, for the following reasons:

1. Using higher spatial frequencies in a simulation requires images sampled at higher resolutions, in order that the images actually contain the desired frequencies. This introduces a number of problems, the chief one being that if the field of view is to be kept reasonably wide, then increasing resolution implies the need for larger images. In the present simulation, images are of size 512×512 pixels; I have found larger image sizes too time-consuming to work with on current serial-processing digital computers.

2. As discussed below, the orientation bandwidth of a cell is positively correlated with its spatial-frequency bandwidth, and the spatial-frequency bandwidth is negatively correlated with preferred frequency. This means that cells tuned to higher frequencies tend to exhibit narrower orientation bandwidths. The narrower the bandwidth, the more cells must be included in a simulation in order to cover the spectrum of possible orientations. The present simulation uses eight orientations and even this involved a small deviation from the desired orientation bandwidth for the 22.6 c/deg frequency band. Using cells tuned to higher preferred frequencies would require increased sampling in orientation.

Sampling in Spatial Frequency

Does the visual system sample spatial frequencies continuously, or are only particular values of preferred frequencies represented in the cortex?

Some early models derived from psychophysical experiments suggested that cells are grouped into a handful of distinct classes tuned only to certain frequencies (Wilson & Bergen, 1979; Wilson, 1991b). However, it seems clear from neurophysiological experiments that there is a continuous distribution of preferred frequencies within the total range (De Valois et al., 1982; De Valois & De Valois, 1990). One can find cells tuned not only to preferred frequencies of, say, 1.0 c/deg, 2.0 c/deg, etc., but to every fractional value that has been tested as well.

The implication for a simulation is that, ideally, every possible preferred frequency within the entire range of spatial frequencies should be represented in the implementation. But this is obviously impractical; it is necessary instead to sample the space of possible values. The most commonly used alternative, dating from the earliest computational models of cortical processing (e.g., Watson, 1982, 1983), is to sample the frequency spectrum at equally-spaced intervals.

Watson (1982) examined the question of how many samples are needed to smoothly cover the average contrast sensitivity curve derived from human psychophysical experiments. The interval between samples must not be greater than a certain proportion of the spatial-frequency bandwidth, otherwise the overall sensitivity curve that results from the collection of spatial filters will have dips where there are none in the human data. Watson concluded that approximately one full bandwidth is the maximum reasonable spacing between preferred frequencies, with one-half bandwidth being a better choice.

To simplify the model here, I chose to sample the range of preferred frequencies using a fixed interval rather than with an interval that varies according to bandwidth of each class of cortical cell. I chose to sample at 0.5-octave intervals. This means that at the lower spatial frequencies, the preferred frequencies of cells are spaced closer than one-half bandwidth apart, whereas at the highest spatial frequency, the spacing is slightly less than would be given by one-half bandwidth. This seems to be a reasonable compromise, and provides a reasonably dense covering of the spatial-frequency spectrum for the shape-from-texture method used in Chapter 4.

Spatial-Frequency Bandwidths

When their sensitivities to different spatial frequencies are measured, cortical neurons differ in their degree of selectivity for their preferred spatial frequency. This is measured physiologically in terms of the *spatial-frequency bandwidth.*

De Valois et al. (1982) found that neurons in their sample of macaque monkey V1 cells ranged in bandwidths from 0.5 octaves to over 3.0 octaves of frequency. They found a median bandwidth for simple cells of approximately 1.4 octaves, and a somewhat higher median for complex cells, indicating that complex cells are on average slightly more broadly tuned. Other physiological studies in both monkeys (Foster et al., 1985; Schiller et al., 1976) and cats (Movshon, Thompson, & Tolhurst, 1978) have found similar values; studies using psychophysical methods have also estimated similar ranges and medians for humans (e.g. Blakemore & Campbell, 1969; Sachs, Nachmias, & Robson, 1971; Watson, 1982).

An important finding by De Valois et al. (1982) is the existence of a negative correlation between preferred spatial-frequency tuning and spatial-frequency bandwidth. That is, the spatial-frequency bandwidths of cortical cells (measured in octaves) decrease slightly with increasing preferred frequency. This means that cells with higher preferred frequencies are generally more narrowly tuned for spatial frequency, on a logarithmic scale, than are cells with lower preferred frequencies. Similar findings have been reported for cat cortical cells as well (Kulikowski & Bishop, 1981), suggesting that this is a consistent result.

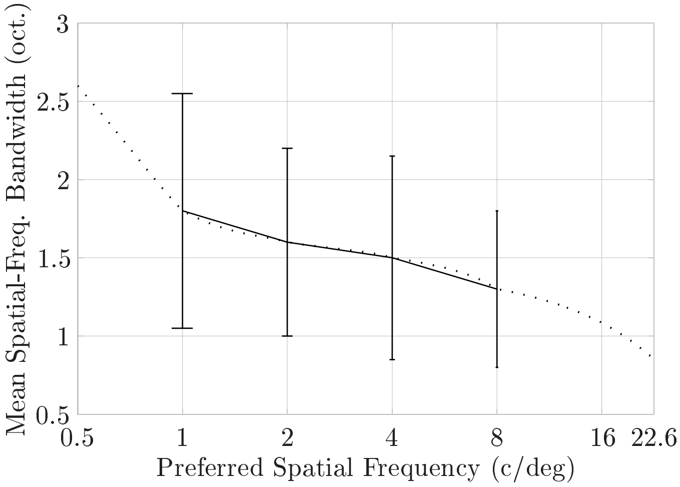


Figure 3.8: Distribution of spatial-frequency bandwidths of V1 simple and complex cells as a function of their preferred spatial frequencies. The solid line and error bars are redrawn from results presented by De Valois et al. (1982, their Figure 7); their distribution is based on over 360 cells located within 5° of the fovea of macaque monkey V1. The dashed line is an interpolated function based on De Valois et al. (1982)'s data, but extended to include points at 0.5 c/deg and 22.6 c/deg.

Figure 3.8 replots data estimated from one of De Valois et al.'s (1982) figures. Their data are shown here as the solid line with error bars. The plot shows the mean spatial-frequency bandwidths for cells tuned to different preferred frequencies.

One question about this data concerns the frequency bandwidths that should be chosen for cells tuned to frequencies lower than 1.0 c/deg and greater than 8.0 c/deg. De Valois et al. (1982) did not provide in their plots specific bandwidths for cells with preferred frequencies in these ranges; they lumped together all cells preferring frequencies less than 0.7 c/deg and, similarly, cells tuned for frequencies greater than 11.0 c/deg. To find bandwidths for cells tuned to preferred frequencies of 0.5, 0.7, 11.0, 16.0 and 22.6 c/deg for the present simulation, I used the following procedure to extend the available data:

1. The mean bandwidth given by De Valois et al. (1982) for cells tuned to less than 0.7 c/deg is approximately 2.2 octaves. The trend in their data at the low frequency end is for cells to have significantly broader bandwidths, with some cells having bandwidths of 3.0 octaves or more. Therefore, to anchor the low spatial-frequency end, I estimated a value of 2.6 octaves for cells tuned to a spatial frequency of 0.5 c/deg.

2. The mean bandwidth given by De Valois et al. (1982) for cells tuned to preferred frequencies greater than 11.0 c/deg is approximately 1.1 octaves, with a standard

deviation that includes bandwidths as narrow as 0.5 octaves. In order to anchor the high-frequency end and have a value for the preferred frequency of 22.6 c/deg used in this simulation, I selected a bandwidth of 0.85 for cells tuned to 22.6 c/deg.

3. I used cubic interpolation to fit a curve to a combination of the values provided by De Valois et al. (1982) and the additional data points just described. The result is shown as the dotted line in Figure 3.8 on the page before. I used this curve to estimate frequency bandwidths for cells tuned to 0.5, 0.7, 11.0, 16.0 and 22.6 c/deg.

The choice of 0.85 octaves bandwidth for cells at 22.6 c/deg is consistent with De Valois et al.'s data, and it also satisfies another, more practical constraint. Based on the joint relationship between spatial-frequency bandwidth and orientation bandwidth (see below), 0.8 octaves is the narrowest frequency bandwidth for which cells will have an orientation bandwidth of at least 22.5°. This is the narrowest orientation bandwidth that can be supported with eight orientation bands. A frequency bandwidth narrower than 0.8 octaves would require more orientations to be sampled in the simulation, increasing the computational requirements. I chose 0.85 instead of 0.8 to provide an extra margin of safety.

It is important to note that these bandwidth values are the experimentally-measured data. How they are interpreted depends on the model of cortical cells being assumed. As noted above in Section 3.2.2, in the framework of the contrast-normalization model, the values should be interpreted as representing measurements on the squared amplitude spectra of the receptive fields, rather than on the (unsquared) amplitude spectra. This affects how the bandwidth values are used for computing the parameters of the Gaussian derivative receptive fields in the simulation below.

It is also worth noting that most other texture-based visual processing models that use spatial filtering assume a fixed spatial-frequency bandwidth for all filters. As explained in this section, the available neurophysiological data (De Valois et al., 1982; De Valois & De Valois, 1990; Kulikowski & Bishop, 1981) suggest that this assumption is unjustified. For Gaussian derivative receptive fields, it is straightforward to allow cells tuned for different preferred frequencies to have different frequency bandwidths. Because of this, and because the neurophysiological data supports the idea that there is a correlation rather than a single median bandwidth, I have chosen to include this aspect in my simulation.

Orientation Bandwidths

Perhaps the most important constraint regarding orientation tuning in simple and complex cells is that the orientation bandwidths are positively correlated with the spatial-frequency

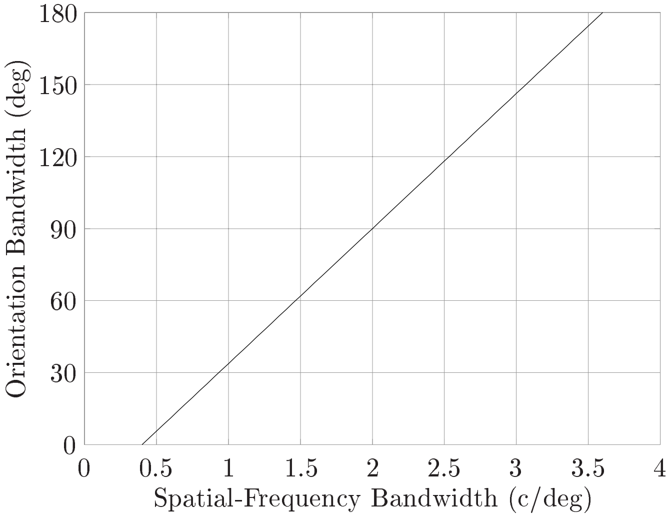


Figure 3.9: Orientation bandwidth as a function of spatial-frequency bandwidth for simple and complex cells of area V1. This corresponds to the regression line provided by De Valois et al. (1982, their Figure 8). Their data are based on a sample of 168 simple and complex cells recorded from within 5° of the fovea of macaque monkey V1.

bandwidths (De Valois et al., 1982; De Valois & De Valois, 1990). In other words, cells that are more narrowly tuned for spatial frequency are also more narrowly tuned for orientation on a logarithmic scale. De Valois et al. (1982) found that the positive correlation between spatial-frequency bandwidth and orientation bandwidth was the largest between any of the variables they measured in their study.

Note that this correlation also implies that preferred spatial frequency is negatively correlated with orientation bandwidth. This follows from the fact that preferred spatial frequency is negatively correlated with spatial-frequency bandwidth. Thus, cells tuned to higher spatial frequencies have narrower frequency bandwidths (on a logarithmic scale) as well as narrower orientation bandwidths (again, on a logarithmic scale).

The data reported by De Valois et al. (1982) can easily be used to derive a rough guide to the orientation bandwidths of cortical cells tuned to different frequencies. In Figure 3.9, I have replotted the regression line provided by De Valois et al. (1982) in their scatter plot of 168 simple and complex cells measured in area V1 of macaque monkeys. The figure shows the average relationship between orientation bandwidth and spatial-frequency widths. It shows that, for instance, cells with frequency bandwidths of over 2.8 octaves typically have orientation bandwidths of about 140°. Conversely, cells with spatial-frequency bandwidths of, say, 1.0 octave, usually have orientation bandwidths of about 35°.

To choose values for the cells in this simulation, I simply estimated the slope and intercept of the regression line from De Valois et al.'s (1982) data to deduce a linear relationship between spatial-frequency bandwidth and orientation bandwidth. I then used this to calculate an approximate value of orientation bandwidth for the cells of each frequency band based on their spatial-frequency bandwidths. The chosen values are summarized in Table 3.3 on page 82.

As is the case with spatial-frequency bandwidths, most other models of texture-based visual processing that use simulated simple and complex cells assume a fixed orientation bandwidth for all cells. Because the neurophysiological data point to a strong correlation between spatial-frequency bandwidth and orientation bandwidth, I have chosen to include this aspect in my simulation.

Sampling in Orientation

Is every orientation around the clock represented in the cortex, or are only particular orientations sampled by simple and complex cells?

Some early physiological studies reported finding orientation preferences at roughly 10° intervals (Hubel & Wiesel, 1959). That would imply a discrete sampling of orientations in the cortex, rather than a continuous one. However, subsequent studies found that every orientation around the clock appears to be sampled, with no tendency to cluster at discrete intervals (De Valois et al., 1982; De Valois & De Valois, 1990).

A simulation would ideally also provide cells tuned to every orientation, but that is impractical, just as it is impossible to include every possible preferred spatial frequency. It is necessary to sample the space of possible orientations covering 180°. (It is not 360° because, for cells that are not sensitive to direction of motion, the two orientations θ and $\theta + 180°$ are equivalent.) The sampling interval is determined by the orientation bandwidths of the cells with the narrowest tuning. In order to provide adequate coverage of 180°, some amount of overlap between neighboring orientations is necessary; the minimum separation needed is equal to the narrowest orientation bandwidth. Because frequency bandwidth is correlated with orientation bandwidth, the cells tuned to the highest spatial frequencies also have the narrowest orientation bandwidths. As already mentioned, in this simulation the narrowest frequency bandwidth of any simulated cell is 0.8 octaves, which leads to an orientation bandwidth of 22.5° based on the plot shown in Figure 3.9 on the page before. This implies eight orientations are needed to cover 180° adequately.

I mentioned above that the narrowest spatial-frequency bandwidth in the simulation was chosen in part with the intention of keeping the number of orientations moderately small. An alternative to maintaining a fixed number of orientations per frequency band is

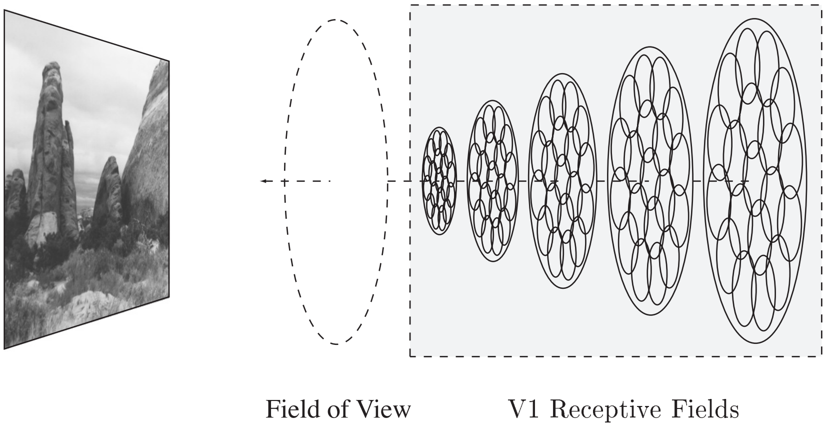


Figure 3.10: Sketch of the general trend of spatial-frequency tunings in the cortex. If simple and complex cells tuned to each spatial frequency could be laid out on separate layers, it might lead to an arrangement such as that shown here. The central, foveal region is analyzed by neurons tuned to all spatial frequencies, from high to low. Progressively more peripheral regions, however, have fewer of the highest-frequency cells. The effect is that peripheral regions of the visual field are only analyzed by neurons tuned to the lower spatial-frequency ranges.

to vary the number represented, by using more orientations at higher spatial frequencies. This is certainly possible, but it would increase the complexity of later stages in the simulation; therefore, for simplicity, I chose to maintain a fixed number of orientations for every frequency.

Sampling in Space

What is the spatial pattern of sampling of the visual input by simple and complex cells? Specifically, is every location in the input processed by cells tuned to every frequency and orientation, or is there a different sampling scheme in the visual cortex?

Some prominent, early studies reported that receptive field sizes increase with increasing eccentricity. This would imply that cells in the fovea are tuned to higher frequencies than cells in the periphery, and that low frequency cells are absent from the foveal area. However, more recent studies using both electrophysiological recordings (De Valois et al., 1982) and radioactive labeling have shown instead that cells in the fovea are tuned to all spatial frequencies, including low frequencies. Peripheral cells are also tuned to both low and high frequencies, although there is a trend for a decreasing number of cells tuned to high frequencies to be present with increasing eccentricity. This suggests a model like that shown in Figure 3.10, where the middle (foveal) range of the visual field is sampled by cells tuned to all frequencies, and peripheral regions have progressively fewer high-spatial-frequency cells.

For this simulation, I chose to sample every spatial location in the input (i.e., every pixel) with every type of simulated cell. The reasoning behind this choice is that the field of view being simulated here is likely to be limited enough that, in a real visual system, nearly all the high frequencies would be found within it. Intuitively, the simulation is intended to take a cylindrical slice through the layers shown in Figure 3.10. A secondary, more practical consideration for putting the same set of cells at every pixel location is that the simulation is made simpler. The alternative, using different spatial-frequency bands at different eccentricities from the center, would unnecessarily complicate the segmentation and form estimation processes that use the outputs of the complex cells.

Summary of Chosen Parameter Values

Table 3.3 on the next page summarizes the parameter values for the model simple and complex cells in this simulation. The first column of the table lists the specific preferred frequencies simulated; each frequency differs from the next by 0.5 octave. For each pair of rows in the table, column two lists the estimated frequency bandwidth for a cortical cell with the given preferred spatial frequency. These values were estimated from Figure 3.8 on page 76 above. The third column of the table lists the estimated orientation bandwidths for simple cells with the given preferred spatial frequencies, based on Figure 3.9 on page 78. As required by the contrast-normalization model, both the spatial-frequency and orientation bandwidths are assumed to be based on the squared amplitude spectrum of the underlying receptive fields. The fourth column lists the derivative orders used for the Gaussian derivative receptive fields. The next two columns of Table 3.3 list the values chosen for the variances σ_x and σ_y, respectively, for each Gaussian derivative. The last three columns show the calculated preferred spatial frequency, frequency bandwidth and orientation bandwidth for a Gaussian derivative receptive field having the given values of n, σ_x and σ_y.

Each desired preferred frequency in Table 3.3 is covered by two types of receptive fields in order to provide the even- and odd-symmetric fields needed by the contrast-normalization model (Section 3.1.2). This yields a pair of rows for each frequency band. I chose the value of n using Table 3.1 on page 64, selecting the derivative order corresponding to the squared amplitude bandwidth that is closest to the desired spatial-frequency bandwidth. The second receptive field of each pair is simply given a value of $n + 1$, because the Gaussian derivative of order $n + 1$ is being used here as an approximation to the quadrature partner of the derivative of order n.

I calculated the values of σ_x and σ_y as follows. For Gaussian derivatives of order n, Equations 3.10 and 3.12 on page 64 directly give the values of σ_x and σ_y, respectively. In

Table 3.3: Parameter values for Gaussian derivative receptive fields in the simulation

Desired Pre. Freq.	Desired Freq. B/W	Desired Orient. B/W	n	σ_x	σ_y	Actual Pre. Freq.	Actual Freq. B/W	Actual Orient. B/W
1.0	1.8	79	1	0.159	0.191	1.0	1.8	79
			2	0.201	0.153	1.1	1.2	79
1.4	1.7	73	1	0.114	0.149	1.4	1.8	73
			2	0.144	0.124	1.6	1.2	73
2.0	1.6	68	1	0.0796	0.114	2.0	1.8	68
			2	0.101	0.0967	2.2	1.2	68
2.8	1.6	68	1	0.0568	0.0814	2.8	1.8	68
			2	0.0718	0.0691	3.1	1.2	68
4.0	1.5	62	1	0.0398	0.0624	4.0	1.8	62
			2	0.0502	0.0538	4.5	1.2	62
5.7	1.4	56	2	0.0395	0.0472	5.7	1.2	56
			3	0.0462	0.0441	6.0	1.0	56
8.0	1.3	51	2	0.0281	0.0377	8.0	1.2	51
			3	0.0329	0.0355	8.4	1.0	51
11.0	1.2	45	2	0.0205	0.0309	11.0	1.2	45
			3	0.0239	0.0293	11.5	1.0	45
16.0	1.1	39	3	0.0172	0.0242	16.0	1.0	39
			4	0.0194	0.0235	16.4	0.9	39
22.6	0.85	25	4	0.0141	0.0267	22.6	0.86	25
			5	0.0155	0.0262	23.0	0.77	25

Pre.: preferred. *B/W*: bandwidth. *Orient.*: orientation.

order to find the appropriate values for the Gaussian derivative of order $n+1$, I used a least-squares numerical fitting procedure to match as closely as possible the shape of the $n+1$ order function to the actual Hilbert transform of the nth-order function having the listed values of σ_x and σ_y. The values of σ_x and σ_y shown in Table 3.3 for the derivatives of order $n+1$ are the values that provided the best fit. The fitting procedure is described in more detail in Appendix B.2.

The calculated preferred frequencies shown in the third-to-last column of the table are the results of using the relationship $f_p = \sqrt{n}/(2\pi\sigma_x)$, derived from Equation 3.10. All of the values for the nth-order derivatives are identical to the desired preferred frequencies listed in the first column, but the calculated peaks for the $n+1$ order derivatives differ slightly from the desired values. These differences arise from using a fitting process to generate the parameters for the quadrature partners. As explained in Section 3.2.3, it is possible to use an alternative method and match the preferred frequencies exactly; unfortunately, doing so generally leads to quadrature pairs whose sum in the space domain is not as smooth as when the pairs are generated using the fitting process. It is difficult to balance all of the constraints; the result is that some small differences in tuning characteristics remain. For the purposes of this project, the small differences are inconsequential.

The frequency bandwidths shown in the second-to-last column are simply the values corresponding to a particular n read from Table 3.1 on page 64. The discrepancies between these values and the desired spatial-frequency bandwidths shown in the first column arise because the Gaussian derivative order n and the frequency bandwidth are in a one-to-one relationship. Setting n necessarily sets a particular bandwidth, and the best value of n may not necessarily lead exactly to the desired frequency bandwidth.

The last column of the table presents the actual orientation bandwidths for cells with the given n, σ_x, σ_y and actual preferred frequency (from the second-to-last column). These values were calculated using Equation 3.12 on page 64.

3.3.2 Achieving Frequency-Domain Tiling and Normalization With Gaussian Derivative Receptive Fields

The contrast-normalization model requires the energy mechanisms underlying the simple and complex cells to tile the spatial-frequency domain in a manner discussed in Section 3.1.3. I mentioned above that it is not possible to achieve this exactly using a collection of Gaussian derivative receptive fields at multiple derivative orders (cf. Simoncelli & Heeger, 1997), but that it *is* possible to achieve an approximation. In this section, I describe a procedure that I developed for this purpose.

The procedure is divided into two parts. The first part is concerned with the actual tiling of the spatial-frequency domain by the energy mechanisms. It can be accomplished in an approximate fashion by adjusting the scaling parameter k_n in $\mathcal{G}_n$ (Equation 3.9) for each receptive field. However, doing so introduces an undesirable side-effect: by the linearity property of the Fourier transform, scaling $\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta)$ by any factor means that the space-domain form, $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$, is rescaled by the same value. If the magnitude of $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$ differs for different spatial-frequency bands, then the maximum amplitudes of simple cells tuned to different frequencies will not be the same, making it impossible to compare the relative strengths of cell responses. Therefore, the second part of the procedure involves readjusting the simple cell output magnitudes using the scaling parameter k_s.

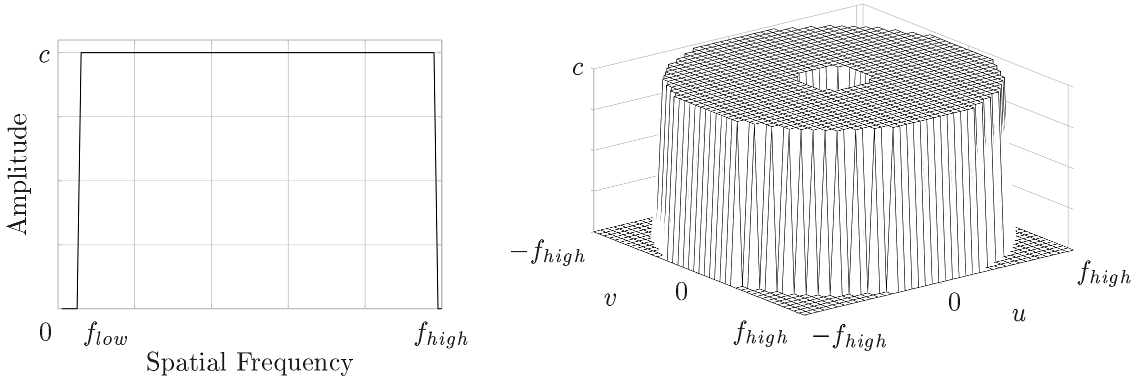


Figure 3.11: Illustration of ideal spatial-frequency tiling. The sum of the squared Fourier amplitude spectra of all the energy mechanisms (the sum of the amplitude spectra of squared quadrature partners) should add up to a constant, c, over the range of spatial frequencies implemented. The absolute value of c is actually arbitrary, because the linearity property of the Fourier transform means that the entire collection of mechanisms can be multiplied by a common factor. Thus, without loss of generality, c can be assumed to be a convenient value such as one.

Adjusting the Gaussian Derivative Magnitudes to Achieve Spatial-Frequency Tiling

Each energy mechanism composing the summation term in the denominator of the simple cell equation [i.e., $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)$ in Equation 3.3 on page 46] is equivalent to a sum of the squares of two operators in quadrature. Such a sum of squares covers a hump-shaped region of the spatial-frequency domain, with a shape similar to that of a single Gaussian derivative's amplitude profile. (See the spatial-frequency profiles in Figure 3.5 on page 63.) The goal of tiling the spatial-frequency domain can be reduced to the problem of making the sum of these hump-shaped profiles be a constant at all frequencies and orientations. The absolute value of the constant is arbitrary, so for convenience, we can use a constant value of one. Ideally, the tiling should produce a cylinder-shaped profile in the spatial-frequency domain, such as that illustrated in Figure 3.11.

The impediment to achieving perfect tiling is that the sum of the hump-shaped regions created by the Gaussian derivative operators does not lead to a smooth shape like that in the figure. However, if there are sufficiently many operators covering the spatial-frequency domain densely enough, then if the individual operator magnitudes are scaled appropriately, the overall sum can be made approximately constant over a range of frequencies.

To simplify the problem, let the scaling parameter k_n in the Gaussian derivative equations (G_n and $\mathcal{G}_n$) be decomposed into the product of two factors,

$$k_n = K_{\mathcal{G}_n}(n, \sigma_x) K_f(f_p) \tag{3.14}$$

where n is the derivative order, σ_x determines the spread of the receptive field function along the x-axis, and f_p is the preferred spatial frequency of a given receptive field. For the moment, let $K_f(f_p) = 1$. The factor $K_{\mathcal{G}_n}(n, \sigma_x)$ can be designed so that the maximum amplitude of $\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta)$ equals the square root of one-half. Then the square of each quadrature partner will equal one-half, and the sum of the squares of two operators in quadrature will have a maximum amplitude of one. The value of $K_{\mathcal{G}_n}(n, \sigma_x)$ that achieves this is the square root of the reciprocal of the maximum value of $\mathcal{G}_n$ (given by Equation B.4 on page 337) times the square root of one-half:

$$K_{\mathcal{G}_n}(n, \sigma_x) = \sqrt{\frac{1}{2}} \left(\frac{\sigma_x \sqrt{e}}{\sqrt{n}} \right)^n \tag{3.15}$$

With this simplification, both operators of the quadrature pair can together be rescaled by a common factor—$K_f(f_p)$—that depends only on the preferred frequency of the operators.

The problem is thus reduced to finding values of $K_f(f_p)$. The factors are the same for every orientation; the values only depend on the spatial-frequency bands implemented by the collection of receptive field operators. For example, in the present simulation, there are ten preferred frequencies, so there are ten values of $K_f(f_p)$.

One approach to finding the $K_f(f_p)$'s is to do a brute-force minimization on the two-dimensional sum of all squared amplitude spectra,

$$\sum_{f_p} \sum_{\theta} [K_{\mathcal{G}_n}(n, \sigma_x) K_f(f_p) (2\pi u)^n e^{-2\pi^2(\sigma_x^2 u^2 + \sigma_y^2 v^2)}]^2, \tag{3.16}$$

adjusting the values of the K_f's until the sum is a smooth function at all points in the spectrum. Fortunately, this brute-force approach is unnecessary because of symmetries in the problem. Instead, it is enough to consider a slice through $v = 0$ in the spatial-frequency spectrum; this is one of the directions along which the sum of the squared amplitude spectra reaches its maximum value. This is now a problem in one-dimensional nonlinear curve fitting. The goal of the curve fitting is that the sum of all the operator spectra that cross $v = 0$ should add up to a constant, one, everywhere from the lowest to the highest preferred frequency implemented. All possible orientations, frequencies and operator phases must be considered in the sum. Since the values of n, σ_x and σ_y are established for a given set of mechanisms, the only parameters that are adjusted during the fitting process are the values of $K_f(f_p)$.

Table 3.4: Values of $K_f(f_p)$ for this simulation.

f_p	$K_f(f_p)$
1.0	0.41976
1.4	0.11954
2.0	0.34859
2.8	0.34739
4.0	0.34544
5.7	0.39231
8.0	0.31200
11.0	0.55560
16.0	0.41077
22.6	0.82249

The curve-fitting process must be performed in the spatial-frequency domain, on a discrete grid corresponding to the frequency samples along $v = 0$. The procedure that I used is detailed in Appendix B.3. The result of applying the procedure is a set of ten values for the $K_f(f_p)$'s. Figure 3.12 on the next page shows the result of applying the procedure to the collection of energy mechanisms implemented in the present simulation. The top of the figure presents a cross-section at $v = 0$ through the squared amplitude spectra of the sum of all quadrature mechanisms; the bottom of the figure shows a three-dimensional rendering of the surface for all u and v. The figure shows that the result is not a smooth tiling—it does not resemble the ideal cylinder shape of Figure 3.11—but it is reasonably good. For reference, the actual values of the $K_f(f_p)$'s used in this simulation are listed in Table 3.4.

Readjusting the Simple Cell Output Magnitudes

The first part of the procedure, described above, adjusts the scaling factor k_n on the Gaussian derivative's Fourier transform, $\mathcal{G}_n$. This means that the space-domain Gaussian derivative G_n becomes scaled by the same value, and this in turn affects the magnitudes of the simple (and complex cell) cell outputs. Therefore, the second part of the procedure involves rescaling the simple cell outputs so that all simple cells have a similar maximum amplitude. More specifically, the goal of this step is to adjust the parameter k_s in the simple cell formula in Equation 3.3 on page 46,

$$S(x, y\,; f, \theta, \phi) = k_s \frac{R(x, y\,; f, \theta, \phi)}{\sigma^2 + \frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)},$$

so that the maximum output value of $S(x, y\,; f, \theta, \phi)$ is a constant. For simulation purposes, the absolute value of this constant is arbitrary; I chose a maximum amplitude of one, so that the simple and complex cell outputs lie between zero and one.

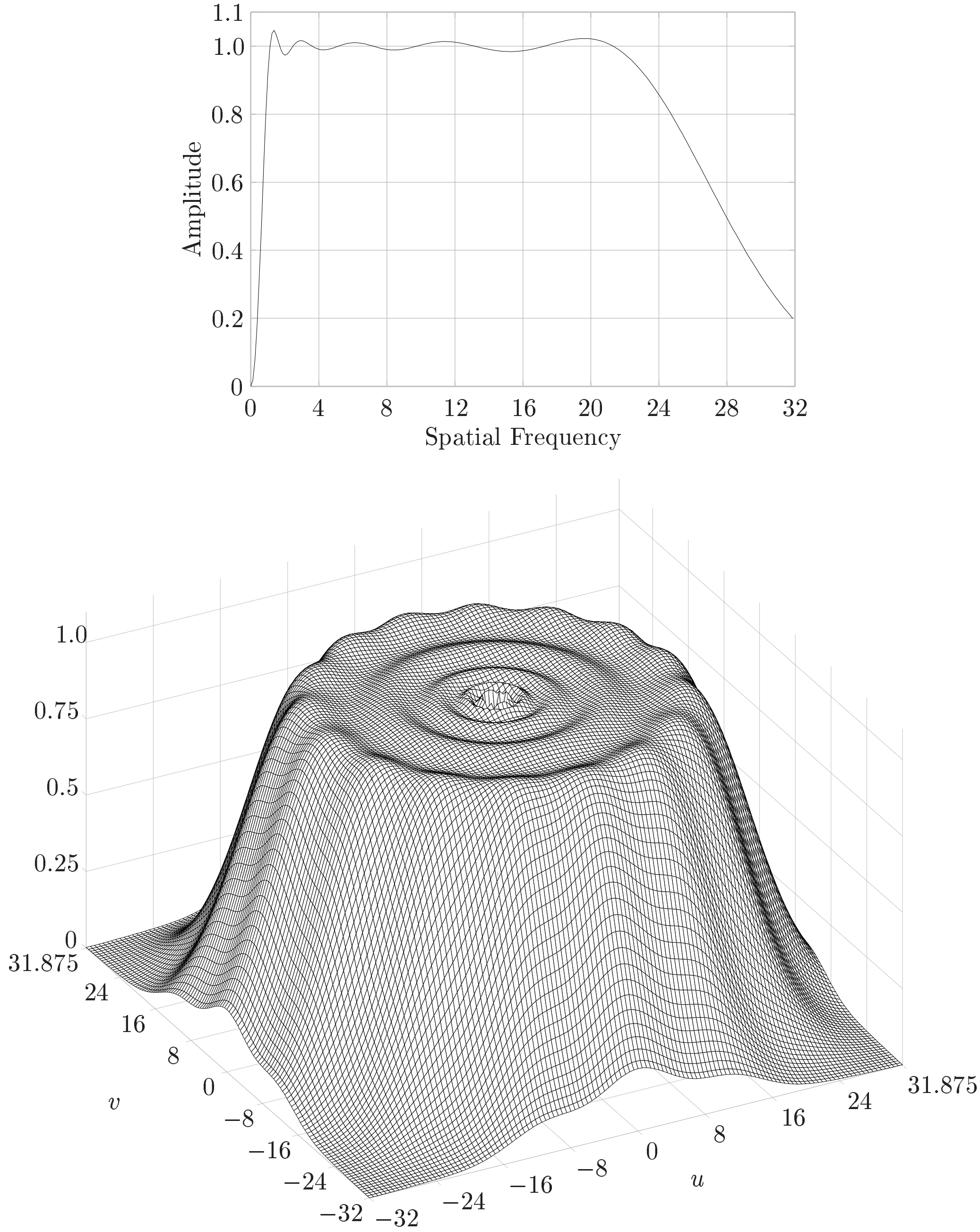


Figure 3.12: Results of adjusting the Gaussian derivative scaling factors to achieve approximate spatial-frequency tiling by the energy mechanisms of the simple and complex cells. (Top) Cross-section through the sum of the squared amplitude spectra at $v = 0$ of the set of Gaussian derivatives using the set of $K_f(f_p)$ values shown in Table 3.4 on the preceding page. The profile tapers to zero below a spatial frequency of 1.0 c/deg and above a frequency of approximately 22.6 c/deg. The shape of the taper at the low and high ends is determined by the bandwidths of the receptive fields tuned to the lowest and highest frequencies. (Bottom) A three-dimensional rendition of the sum of the squared amplitude spectra. This shows the quality of tiling when all orientations are considered. The result is not an entirely smooth surface; bumps and ripples remain.

In the discussion that follows, let $\hat{S}(x, y\,; f, \theta, \phi)$ represent the fraction in the simple cell equation; that is, let

$$S(x, y\,; f, \theta, \phi) = k_s \hat{S}(x, y\,; f, \theta, \phi), \tag{3.17}$$

where

$$\hat{S}(x, y\,; f, \theta, \phi) = \frac{R(x, y\,; f, \theta, \phi)}{\sigma^2 + \frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)}. \tag{3.18}$$

The objective is to find a value of k_s that, when used to multiply $\hat{S}(x, y\,; f, \theta, \phi)$, will rescale $S(x, y\,; f, \theta, \phi)$ so that it lies between zero and one. To achieve this, first let k_s be a quotient of two quantities:

$$k_s = \frac{K_\perp(f_p)}{K_\top(f_p)}. \tag{3.19}$$

The two components of this fraction are both functions of the preferred frequency f_p to which a given simple cell is tuned. The value of $K_\perp(f_p)$ for cells tuned to a particular frequency f_p ideally should be equal to the predicted value of the *denominator* of $\hat{S}(x, y\,; f, \theta, \phi)$, and the value of $K_\top(f_p)$ should ideally be equal to the maximum value of the *numerator* of $\hat{S}(x, y\,; f, \theta, \phi)$. The reasoning behind this is the following. The value of the summation term $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)$ will never be less than zero, and if the value of σ^2 is not less than 1.0, the denominator of the simple cell equation will never be less than 1.0. That means the value of $\hat{S}(x, y\,; f, \theta, \phi)$ will be bounded above by the maximum value of the numerator. By setting $K_\top(f_p)$ to the maximum value that the numerator can take on, and setting $K_\perp(f_p)$ to the expected value of the denominator of $\hat{S}(x, y\,; f, \theta, \phi)$, then multiplying $\hat{S}(x, y\,; f, \theta, \phi)$ by $k_s = K_\perp(f_p)/K_\top(f_p)$ ensures that the output of the simple cell function, $S(x, y\,; f, \theta, \phi)$, will be bounded by 1.0.

The success of this approach depends on finding the maximum values of the numerator and the "expected" value of the denominator of each type of simple cell implemented in the simulation. It is possible to do this analytically, but I have found that the values found in this manner are not useful in practice—the resulting $K_\perp(f_p)$'s and $K_\top(f_p)$'s tend to either overestimate or underestimate the actual values that result when running a simulation. For example, $K_\top(f_p)$ can be calculated directly from the fact that the maximum value of the numerator in the simple cell equation [i.e., $R(x, y\,; f, \theta, \phi)$] is the maximum squared value of a Gaussian derivative receptive field operator. However, the resulting value is a poor choice for $K_\top(f_p)$ because it usually overestimates the maximum obtained in a running simulation. The reason is simply that, when using real images and software simulations, no stimulus ever manages to stimulate the receptive fields in an optimal manner. Therefore, I used an approach that estimates appropriate values for the $K_{\perp}(f_p)$'s and $K_{\top}(f_p)$'s based on the outputs of the actual receptive field operators in the simulation.

To set $K_{\top}(f_p)$, I used the following procedure. For each preferred spatial frequency f_p implemented in the simulation:

1. Create an artificial image consisting of a sine wave having a spatial frequency equal to f_p. This is the optimal stimulus for a receptive field tuned to f_p.

2. Apply both the even- and odd-symmetric receptive field operators to the stimulus, and compute each operator's squared output using the simulation system;

3. Find the maximum output value of each of the two types of operators, and pick the largest of the two maxima as the value of $K_{\top}(f_p)$.

This produces a value for $K_{\top}(f_p)$ for each f_p implemented in the simulation. The result is close to the actual maximum value of the numerator of $\hat{S}(x, y\,; f, \theta, \phi)$ that can be produced in simulation runs.

Setting the second factor, $K_{\perp}(f_p)$, requires finding the expected value of the denominator of $\hat{S}(x, y\,; f, \theta, \phi)$ for a given f_p. To do this, first note that the denominator in the equation is dominated by the summation term $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)$, so the problem of finding the expected value of the denominator reduces to the problem of finding the expected value of the summation term. For this, I exploited the relationship between the summation term and the energy of the Fourier spectrum of a stimulus. The relationship was explained in Section 3.1.3; in essence, the summation term (for a given range of frequencies and orientations) is proportional to the energy of the Fourier spectrum of the stimulus in an annulus of frequencies and orientations. Using this relationship, one can find the value of $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)$ for a given input image by sampling the discrete power spectrum of the image. Then, by averaging the results over many example images, it is possible to estimate the typical value for the summation term, and thereby obtain a predicted value for the denominator of the simple cell equation.

The procedure I used to achieve this is the following:

1. For each preferred spatial frequency f_p implemented by the set of simple and complex cells, determine the appropriate upper and lower cut-off frequencies (f_h and f_l, respectively) for the annulus used in the summation $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)$. In the present simulation, there are ten preferred frequencies, spaced one-half octave apart. Let these frequency bands be signified by $f_1, f_2, \ldots, f_{10}$. Each frequency

band is to be normalized by the responses of cells in a three-octave-wide annulus of frequencies. The lower cut-off frequency, f_l, for a given f_i is the lower half-bandwidth point of frequency band f_{i-2}. The upper cut-off frequency, f_h, is the upper half-bandwidth point of frequency band f_{i+2}. (The terms are -2 and $+2$ because the spacing is one-half octave; an annulus of three octaves centered on f_i therefore encompasses two f_i's on either side of f_i.)

2. Select a sample of the images that will be used as inputs to the simulation. I used a set of twelve images from among the test cases used for Experiments One, Two and Three.

3. For each image in the set:

   (a) Calculate the discrete power spectrum of the image. This will be a two-dimensional grid of values. The index of each value is a pair of spatial frequencies, one frequency index for the x-axis direction, another for the y-axis direction.

   (b) For each frequency band f_i, take the sum of all the power spectrum values on the grid between the spatial frequencies f_l and f_h, at all orientations. This traces out a circular ring in frequency space bounded by $f_l \geq \sqrt{u^2 + v^2}$ and $f_h \leq \sqrt{u^2 + v^2}$, where u and v are the spatial frequencies in the x-axis and y-axis directions, respectively. Figure 3.13 illustrates this. Let $P(f_i)$ signify the value of this sum.

4. For each frequency band f_i, set $K_\perp(f_i)$ to the mean value of the $P(f_i)$'s collected from the set of images.

The result of this is an average value for the energy of the Fourier spectrum in each annulus of frequencies centered on the f_p's implemented in the simulation. Because it is an average obtained by measuring the spatial-frequency characteristics of actual input images, it tends to reflect more accurately the typical value of the summation term $\frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x, y\,; f, \theta, \phi)$ in $\hat{S}(x, y\,; f, \theta, \phi)$. This serves as a reasonable choice for $K_\perp(f_p)$.

One question in this procedure is what to do with the highest and lowest frequencies f_p implemented in the simulation. There are no frequency bands below f_1 and above f_{10}, so how should the f_{i-2} and f_{i+2} points described above be calculated for these bands? In the implementation described here, the low end can be handled easily: the lowest frequency is 1.0 c/deg, which leaves room for two imaginary frequency bands (at 0.7 c/deg

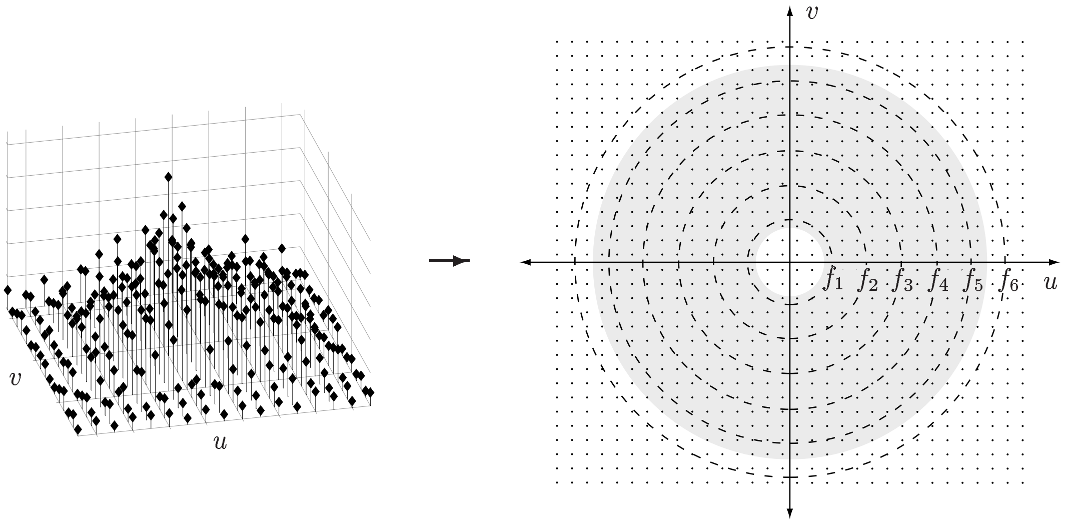


Figure 3.13: Illustration of how annuli of spatial frequencies and orientations are sampled in the process of finding the set of $K_{\perp}(f_p)$ values. First, the power spectrum (i.e., the squared Fourier amplitude spectrum) is obtained for an image. This has the form of a large number of values on a two-dimensional grid. Each of the preferred frequencies f_p implemented in the simulation will trace out a ring on the grid according to $f_p = \sqrt{u^2 + v^2}$, where u and v are the spatial frequencies in the x-axis and y-axis directions, respectively. Six example frequencies are shown as dotted lines and labeled $f_1, f_2, \ldots, f_6$. Let $f_i = f_3$ be a specific preferred frequency under consideration. The annulus used in the normalization sum of the simple cell equation consists of all points between frequencies f_l and f_h. The value of f_l is one-half octave lower than f_{i-2}, and the value of f_h is one-half octave higher than f_{i+2}. For $f_i = f_3$, this traces out a ring-shaped region enclosing all frequencies slightly below f_1 and slightly above f_5. This ring-shaped region is shown shaded in gray.

and 0.5 c/deg) that can be inserted into the calculations for purposes of computing the annulus sum. Unfortunately, the matter is more difficult at the high-frequency end. The viewing configuration being assumed here (see Section 3.3) implies that the highest spatial frequency in the input images is assumed to be 32 c/deg. The highest preferred frequency band that is implemented is 22.6 c/deg; one-half octave above this is 32 c/deg, which means that there is no higher frequency band to use for $f_h = f_{i+2}$. To work around this problem, I calculated the two highest annulus sums as follows:

- For f_9, I took the annulus sum to be from $f_l = f_7$ to $f_h = f_{10}$, and added an additional $^1/_5$ of this value to the result.
- For f_{10}, I took the annulus sum to be from $f_l = f_8$ to $f_h = f_{10}$, and added an additional $^2/_5$ of this value to the result.

The reasoning behind this is that each annulus sum consists of five frequency bands, and so the best approximation for the missing bands at the high end is using a fraction of the available data. This is not a perfect solution, but appears to work well enough in practice.

It is worth noting in passing that Heeger's (1991, 1992a, 1992b) contrast-normalization model does not address this issue (i.e., the problem of what to do with the normalization sum at the high-frequency end). The problem must necessarily be faced in any implementation of the contrast-normalization model, including the brain's. If, in a collection of simple cells, there are no receptive fields with preferred frequencies higher than some value $f_{\max}$, then there is no way to properly normalize the simple cells tuned to $f_{\max}$ and the nearby lower frequency bands. One has to use an alternative method, such as the approximation described above.

Summary: Achieving Frequency-Domain Tiling and Normalization With Gaussian Derivative Receptive Fields

This section described a procedure for achieving approximate tiling of the spatial-frequency domain with Gaussian derivative receptive fields of multiple derivative orders. The procedure consists of adjusting k_n in the Gaussian derivative formula and k_s in the simple cell formula. Factor k_n is first decomposed into a product of two terms, $K_{\mathcal{G}_n}(n, \sigma_x) K_f(f_p)$. For each preferred frequency implemented by the collection of receptive fields in the simulation, $K_{\mathcal{G}_n}(n, \sigma_x)$ is set using an analytical formula (Equation 3.15 on page 85), and $K_f(f_p)$ is set using a curve-fitting process. Similarly, factor k_s is decomposed into a quotient of two terms, $K_{\perp}(f_p)/K_{\top}(f_p)$. The term $K_{\perp}(f_p)$ is set using an empirical procedure for determining the maximum output value produced by the odd- and even-symmetric receptive field operators in the simulation. The term $K_{\top}(f_p)$ is set using another empirical procedure for estimating the typical value of the summation in the denominator of the simple cell formula. The procedure for setting $K_{\top}(f_p)$ exploits the proportionality, described in Section 3.1.3, between the summation in the simple cell formula and the Fourier energy of an input image. The resulting values for k_s turn out to be different for cells tuned to different spatial frequencies, but this is acceptable in the contrast-normalization model.

The method described in this section is designed to bound the simple cell outputs between zero and one. However, it is important to be clear that it is only an approximation. In particular, the upper bound is not guaranteed—if the value of $K_{\perp}(f_p)$ (Equation 3.19) is too low, it is possible that the simple cell outputs will exceed 1.0 on occasion. I have not found this to be a problem in the simulations described in this dissertation, but nevertheless it is possible.

3.3.3 The Simulation Architecture

I implemented the simulation of simple and complex cells described in this chapter using a mixture of code written in the language C (Kernighan & Ritchie, 1988) and the Matlab numerical computation and visualization environment (MathWorks, 1998a). All computations in the simulation are performed using double-precision floating-point arithmetic.

All input images are monochromatic with 256 gray levels and a size of 512 × 512 pixels. The simulation preprocesses each image to linearly rescale the pixel values in order to insure that the image contains the full range of values from 0 to 256. As mentioned at the beginning of this section, all spatial-frequency quantities are calculated under the assumption of a viewer located in front of a computer screen at a distance that leads to 1.0 degree of visual angle being equal to 64 pixels. Therefore, the 512 × 512 pixel images cover a field of view of $8° \times 8°$.

The simulation architecture is illustrated in Figure 3.14 on the next page. Complex cell responses are represented as two-dimensional maps of values, in which each point corresponds to the response of a cell centered at that location in the image. Consequently, each map is 512 × 512 units in size. There is one type of complex cell for each combination of preferred spatial frequency and orientation. Because this simulation includes ten preferred frequencies and eight orientations, there are 80 possible complex cell response maps. The simulation also includes the ability to generate simple cell responses; there is a total of 320 possible simple cell types (ten frequencies, eight orientations, four phases). However, as explained below, it is not necessary to compute the simple cell responses for the purposes of this research, since the complex cells can be more efficiently implemented in terms of full-squared operators.

Computing Cell Responses

The model simple and complex cells defined by Equations 3.3 and 3.4 on page 48, respectively, can be implemented in a direct, feedforward manner. That entails first computing the receptive field operator responses, then dividing the results by the normalization sums in the denominators of Equations 3.3 and 3.4. In neural circuits or limited-precision fixed-point arithmetic, this would not be possible because the intermediate values in the formulas would exceed the limited dynamic ranges of such mechanisms; instead, a feedback normalization architecture would be required (Heeger, 1992b; Carandini & Heeger, 1994). However, the use of floating-point arithmetic permits using the simpler, more direct approach of computing the quantities exactly as expressed in the equations. It is also possible to make certain simplifications.

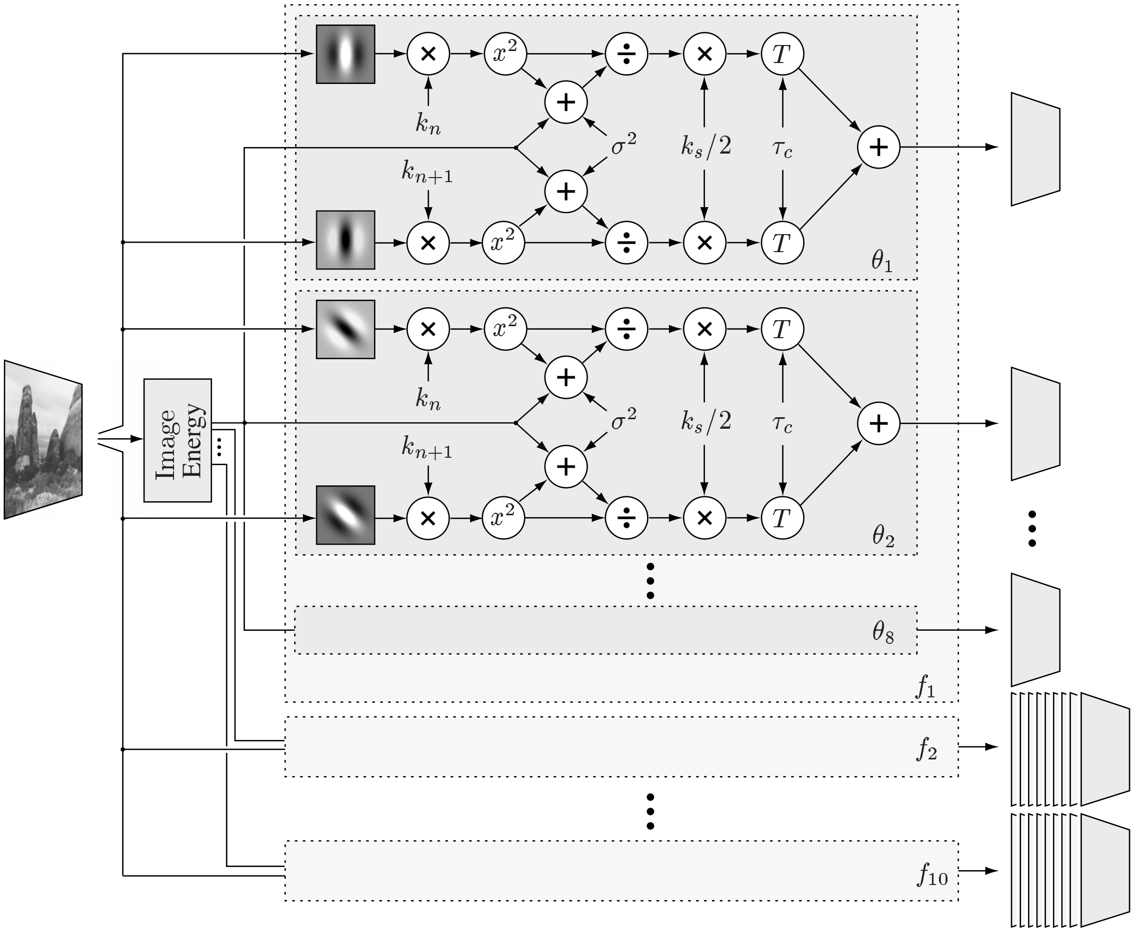


Figure 3.14: The simulation architecture. Although each complex cell is conceptually composed from four simple cells, in practice it can be simulated more efficiently using two full-squared receptive field operators, as described in the text. The receptive fields are Gaussian derivatives with amplitudes scaled by a factor k_n, where n is the order of the derivative. Each pair of full-squared operators is then normalized (divided) by the equivalent of the outputs of many other operators. But rather than use the outputs of many other cells directly, it is more efficient to exploit the relationship between the image's Fourier energy and the denominator of the complex cell equation (see Sections 3.1.3 and 3.3.2). The normalized output of each operator is multiplied by a scaling factor of $k_s/2$, thresholded using T_c, and finally added together to create the output of a complex cell. The simulated cell responses are represented as two-dimensional maps of non-negative numbers; the value at a given location in each map represents the activity level of a neuron centered at that location in the image. There is a total of 80 complex cell response maps, one for each combination of orientation angle and spatial frequency.

The formula for the complex cell (Equation 3.4) takes the average of four simple cells. Writing out the equation for a complex cell gives

$$\begin{aligned} C(x,y\,;f,\theta) &= \tfrac{1}{4}\sum_{\phi=1}^{4} S(x,y\,;f,\theta,\phi) \\ &= \tfrac{1}{4}\sum_{\phi'=1}^{4} k_s \frac{R(x,y\,;f,\theta,\phi')}{\sigma^2 + \frac{1}{4}\sum_i \sum_f \sum_\theta \sum_\phi R_i(x,y\,;f,\theta,\phi)} \end{aligned} \tag{3.20}$$

The numerator of each simple cell is the output of a half-squared operator. But as mentioned in previous sections, each pair of half-squared operators is equal to a full-squared operator. The full-squared operators here are simply the squares of the Gaussian derivative receptive field functions,

$$R(x,y\,;f,\theta,0^\circ) + R(x,y\,;f,\theta,180^\circ) = [G_n(x,y\,;\sigma_x,\sigma_y,\theta)]^2 \tag{3.21}$$

$$R(x,y\,;f,\theta,90^\circ) + R(x,y\,;f,\theta,270^\circ) = [G_{n+1}(x,y\,;\sigma_x,\sigma_y,\theta)]^2, \tag{3.22}$$

where n, σ_x and σ_y set the spatial-frequency tuning characteristics for a given receptive field according to Table 3.3 on page 82. So, instead of computing the response values of four simple cells to get the output of one complex cell, for simulation purposes it is more efficient to compute the responses of just two full-squared linear operators, then multiply that by $k_s/2$ divided by the term in the denominator of the complex cell equation.

By making use of the relationships between the summation term in the denominator and the Fourier spectrum of the stimulus, it is possible to gain additional savings in computations. As explained in Section 3.3.2, the normalization sum can be approximated directly from the spectrum of the stimulus rather than actually summing the outputs of many linear operators.

The response of a linear receptive field operator (a Gaussian derivative) is obtained by cross-correlating it with the input image. However, direct cross-correlation is an expensive computation for even moderately-sized receptive fields, and is prohibitively expensive for the sizes of receptive fields that result for the cells tuned to the lowest spatial frequencies. Instead of computing cross-correlation directly, I implemented the operation by using the well-known convolution theorem in signal processing (Bracewell, 1986). This allows performing cross-correlation by way of the Fast Fourier Transform (FFT) instead of by direct computation (Bracewell, 1986; Brigham, 1974).

Although an FFT-based algorithm is faster that performing direct cross-correlation for most receptive field sizes, it is still a relatively time-consuming operation. An even more efficient computational method for performing multiscale cross-correlation is the

Gaussian pyramid algorithm (Burt, 1981; Burt & Adelson, 1983; Burt, 1984). In fact, I used this method in an early implementation of the present simulation. However, a later modification—the introduction of cells tuned to preferred frequencies at one-half octave spacing—meant that the Gaussian pyramid could no longer be used. This is due to the fact that the levels of the pyramid correspond to one octave differences in frequency. Still, a possible future enhancement to this simulation would be to implement a hybrid scheme: using a Gaussian pyramid to compute the filter responses at one-octave frequency intervals (1.0, 2.0, 4.0, 8.0 and 16.0 c/deg), and using FFT-based cross-correlation for the in-between frequencies.

An FFT-based cross-correlation method also permits an additional simplification. Computing a filter's response in the space domain would entail first creating a discrete array representation of the receptive field. This brings with it a number of issues related to proper sampling of the filter function (cf. Bovik et al., 1990). For example, a filter tuned to a high spatial frequency needs to be implemented with enough samples to avoid frequency aliasing, but the need for many samples trades off against the resolution of the input image. It is often not possible to implement digital filters that encompass the highest frequencies that can actually be present in the input image. But in an FFT-based approach, it is the Fourier transform of the filter that is digitized rather than the space-domain representation of the filter. Implementing the filter by direct sampling of the Fourier transform avoids issues of minimum sampling frequencies and aliasing. Moreover, because the Gaussian derivative function has a simple analytical form in the spatial-frequency domain, the filter transform can be computed directly from $\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta)$ of Equation 3.9 on page 61. The transform function $\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta)$ simply needs to be sampled at discrete frequency points corresponding to the grid used in the Fourier transform of the input image.

Thresholding Low Output Values

The use of floating-point arithmetic sometimes results in extremely small, non-zero values that are, realistically, better set to zero. For this reason, I added a thresholding step on the output of simple and complex cells. The actual cell responses are therefore given by

$$\tilde{S}(x, y\,; f, \theta, \phi) = T_c[S(x, y\,; f, \theta, \phi)] \tag{3.23}$$

$$\tilde{C}(x, y\,; f, \theta) = T_c[C(x, y\,; f, \theta)] \tag{3.24}$$

where the thresholding function $T_c(x)$ is given by

$$T_c(x) = \begin{cases} x & \text{if } x > \tau_c \\ 0 & \text{otherwise.} \end{cases} \tag{3.25}$$

The parameter τ_c is an absolute threshold on the outputs of the simulated complex cells. I used a value of 0.001 for τ_c for all simulations described in this dissertation. I chose this value based on (a) observations of the mean value of the outputs produced by the simulated complex cells for different input images, and (b) the effects of different thresholds on the pattern of responses of the cells. The mean value of the cell outputs in this implementation ranges from 0.004 to 0.01 for most images; the peak response of strongly activated cells typically ranges from 0.1 to 0.8. A threshold of $\tau_c = 0.001$ eliminates the weakest responses but leaves the majority of other responses untouched.

Padding the Input Image

A perennial issue in implementing image filtering operations is the question of what to do at image borders. The lack of data beyond the edges of the image complicates the task of computing responses at image locations near the borders, especially for large receptive field operators such as those of cells tuned to low spatial frequencies.

The approach that introduces the fewest artifacts is to ignore the responses in the outer-most image region equal to one-half the width of the widest receptive field. Unfortunately, the filters tuned to 1.0 c/deg have receptive fields that are 137 pixels wide in the present simulation. Truncating the response maps by 68 pixels all around would significantly reduce the working size of the input image.

The alternative is to extend the input image by adding padding beyond its edges. The simplest approach is to pad with zeros, but when using receptive fields that have positive (excitatory) and negative (inhibitory) regions, zero-padding leads to severe distortions at the image borders. A better method is to reflect the image about the edge pixels, as is commonly done in image processing (Pratt, 1991). This is the method I used for the present simulation. Even though cross-correlation is implemented here by way of the FFT, it is still important to perform reflection-padding on the input image prior to computing the FFT-based cross-correlation.

This image padding approach works quite well in general. However, it does not completely eliminate edge effects in all cases. Some types of textures are more prone to producing edge effects than others, as shown in the results of Experiment One below. Regular textures are the most problematic; irregular textures, the least.

3.4 Experiment One: Testing the Complex Cell Simulation

This section describes the results of tests performed on the complex cell network described in the previous section. The tests center around three predictions about the expected behavior of the simulation. This section also serves to illustrate the kinds of outputs produced by the network of model complex cells.

Given the goals of this research and the function of the initial processing component—the network of complex cells—in the overall theory, three predictions arise naturally for the capabilities of this component:

1. Flat textures differing in average orientation, spatial frequency, or a combination of the two should lead to responses in different subpopulations of the total set of complex cells. The cells that respond are expected to have tuning properties that are closely matched to the characteristics of the underlying textures. The spatial extent of the responses should be roughly equal to the extent of the particular texture area in the image, with little or no response from those cells over image areas that have significantly different types of textures. The rest of the set of complex cells (those that have significantly different spatial-frequency or orientation tunings) are expected not to show strong responses.

2. Gradients in textures arising from changes in surface orientation should lead to responses in different subsets of cells within the same region of texture. In other words, whereas flat textures are expected to produce relatively homogeneous responses in some subset of complex cells, curved or slanted textures are expected to produce responses in varying subsets of cells at different locations over the region of texture. This is expected from the discussion in Section 2.3 of how changes in surface orientation generally affect local spatial-frequency content over a surface.

3. Regions devoid of texture in an image should not lead to responses in the simulated cells. This follows from the fact that simple and complex cells are selective for localized changes in contrast, and those only occur where there are lines, edges, texture grains or other markings in the image.

The results of testing each prediction are discussed below.

3.4.1 Methods

Each of the three predictions above was tested using a common methodology. This consisted of running the simulation using different sample images as input, and examining the complex cell response maps for patterns of activity that confirmed or denied the prediction. This is admittedly a qualitative approach with no objective criteria for success or failure, but the nature of the questions are such that it is simple to evaluate the results based on a visual inspection of the response maps.

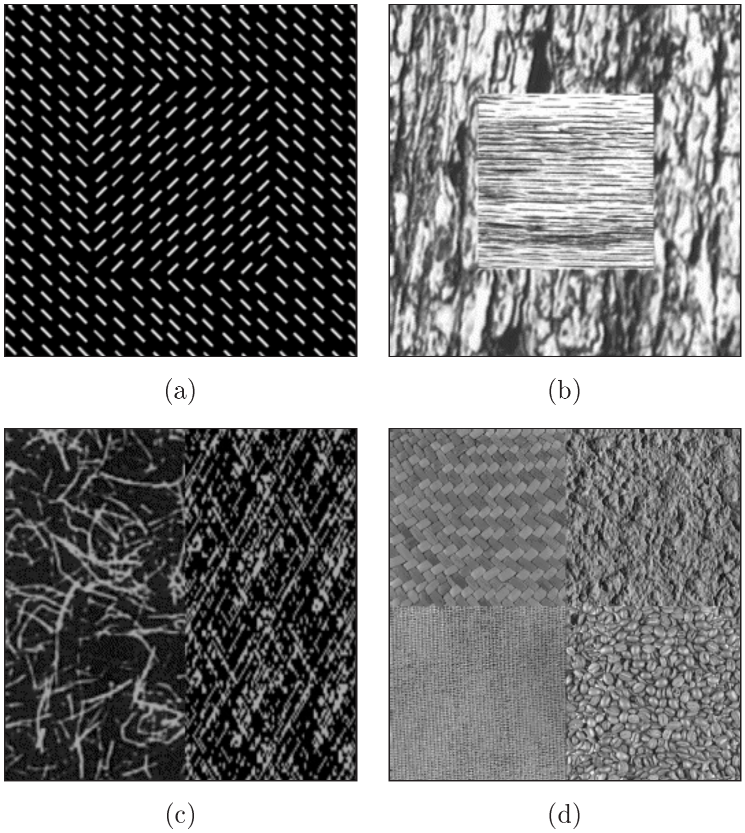
The set of images used for testing in this experiment came from a variety of sources, and included five classes of images: single patches of textures viewed frontally, single patches of textures viewed from an angle, collages of multiple patches of textures, images of objects with textured surfaces, and natural, outdoor scenes. Images of flat texture patches were generally obtained from freely available image databases on the Internet, such as the VisTeX database (Picard et al., 1995). Some images were created by texture-mapping artificial textures onto simulated three-dimensional objects. All images were 512 × 512 pixels in size.

3.4.2 Results for Prediction One: Responses to Flat Textures

The first prediction involves images of flat textures viewed from a frontal orientation. The prediction is simple: textures with different characteristics should lead to activity in different subpopulations of the simulated complex cells, based on differences in dominant orientation, dominant spatial frequency, or both. Intuitively, this means that it should be possible to see, in a visual inspection of the outputs, the locations and extent of different texture regions in the images. This is not expected to be evident in all response maps, because not all complex cell types will respond equally to different inputs, but at least some subset of the cells should display this pattern of results for each input image.

I used four images in this first test. The images were collages of different textures and are shown in Figure 3.15 on the next page for reference. The responses to each of the four input images are displayed in Figure 3.16 on page 101 through Figure 3.19 on page 107. Note that the full set of responses to each input contains 80 complex cell response maps; in order to conserve space, I only show the full 80 maps for the first input image, in Figure 3.16. Subsequent figures only show one-half of the full set, using the outputs of cells tuned to every other orientation.

Figure 3.16 shows maps of the complex cell responses when the simulation is presented with the first input image of Figure 3.15(a). Each response map shown in the figure is 512 × 512 elements in size, with each point representing the activity of a simulated complex cell whose receptive field is centered at that location in the input image. Degree of activity is coded as intensity in the maps, with black representing little or no activity, white representing the highest activity, and shades of gray representing varying degrees of activity in-between. Spatial frequency increases left-to-right in columns, beginning with



(a)

(b)

(c)

(d)

Figure 3.15: The images used for testing the simulated complex cells' responses to flat textures. (a) Two artificial textures with orthogonally-oriented texture elements. (b) Two natural wood bark textures, one patch inset inside another. (c) Closeup of grass in the left half and a textile in the right half. (d) Four natural textures; in clockwise order from the upper left, they are: raffia weave, poured concrete, coffee beans, and woven fabric. Sources: (b,c) Brodatz (1966) album; (d) VisTeX database (Picard et al., 1995).

the lowest frequency (1.0 c/deg) in the leftmost column and going through to the highest (22.6 c/deg) in the rightmost column on the second page of the figure. Each row shows a different orientation, ranging from vertical (0°) at the top and turning counterclockwise going down the rows. The contrast in each of these maps has been readjusted to cover the full span of possible gray-levels, in order to highlight differences in the responses of cells within each map.

The figure shows that the response maps do display the predicted patterns of activity. In particular, the strongest responses to the middle texture patch occur in one set of cells while the strongest responses to the outer region of texture occur in a different set of cells (those tuned to the orthogonal orientation). This is evident, for example, in the responses of cells tuned to the frequencies 4.0–8.0 c/deg, and orientations 22.5°–67.5°. These strong responses occur in those cells whose spatial-frequency and orientation selectivities are

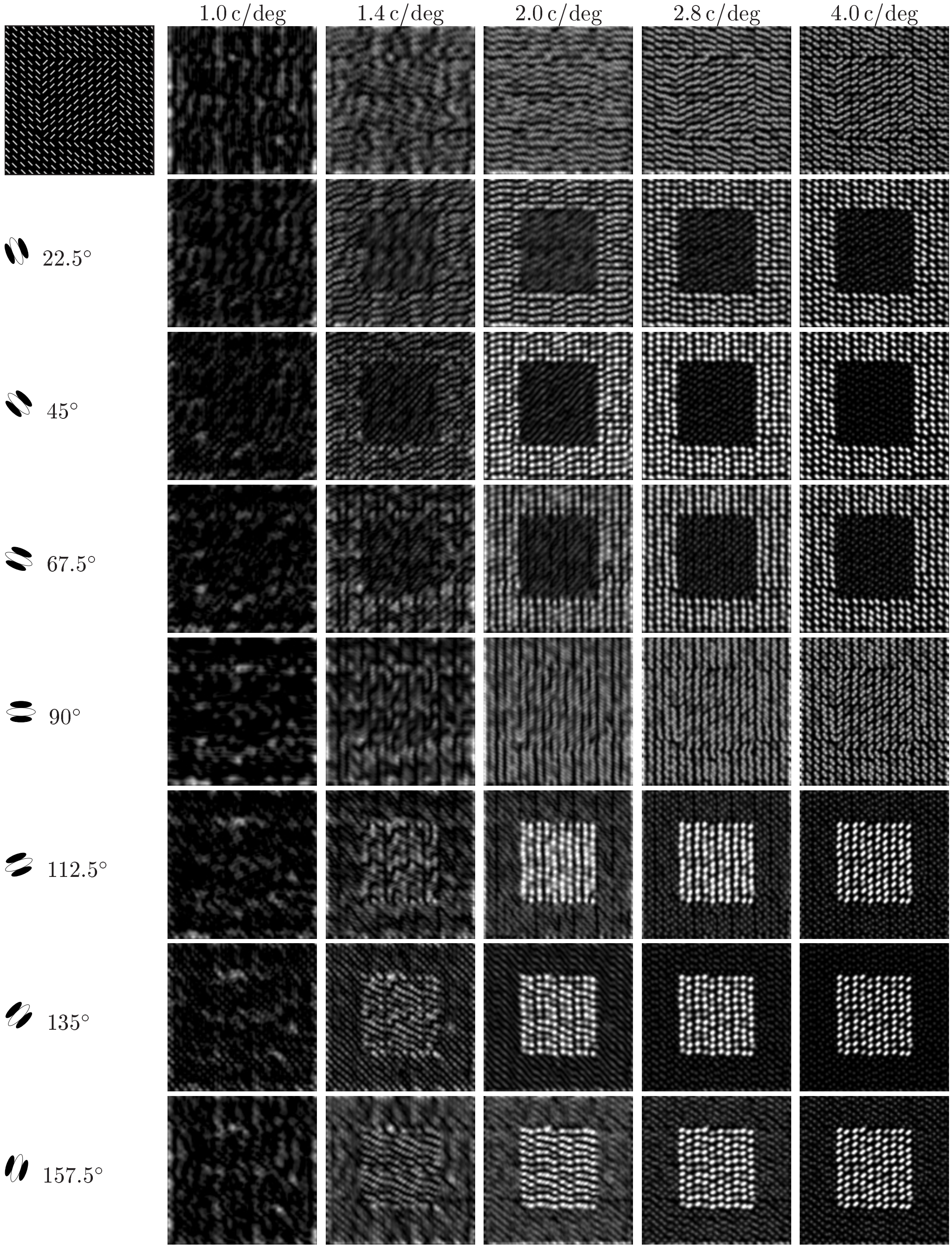


Figure 3.16: Maps of responses of simulated complex cells to the image shown in Figure 3.15(a) on the page before. Each point represents the level of activity of a simulated cell with its receptive field centered on the corresponding image location. Brighter points indicate greater activity. The input image is shown at the top left; the rest of the top row contains response maps for cells at 0° orientation. (Continued on the next page.)

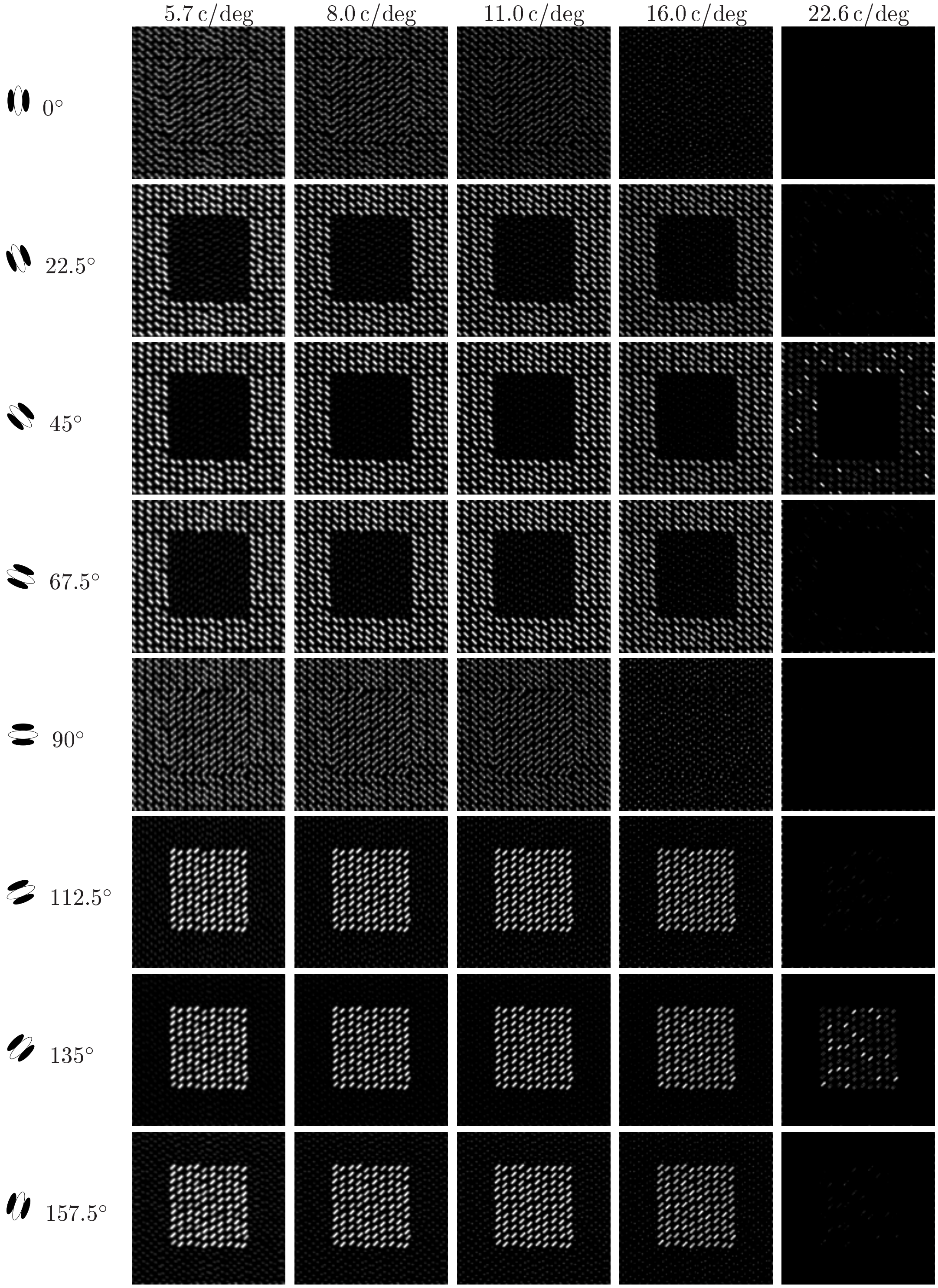


Figure 3.16: (Continued from the previous page.) Maps of responses of simulated complex cells to the image shown in Figure 3.15(a) on page 100.

closest to the sizes and orientations of individual texture elements. The responses are relatively homogeneous over the texture regions. Cells tuned outside of these ranges of spatial frequencies and orientations do not respond well to the outer region of texture. Cells tuned to orthogonal orientations respond most strongly to the middle square texture region.

The texture elements in the input image are oriented at exactly 45° and 135°. But the simulated responses are not localized to complex cells tuned only for these orientations. For example, the middle texture square evokes activity in a number of types of complex cells. This happens because the cells have fairly broad tuning in spatial frequency and orientation. To take a concrete example, cells tuned to 2.8 c/deg have an orientation bandwidth of 68° (see Table 3.3 on page 82), which means that cells tuned to a preferred orientation of 157.5° will respond to stimuli oriented approximately 124° through 191°. This explains why the cells tuned to 2.8 c/deg and 157.5° respond to the center texture square even though the texture elements are actually oriented at 135°. On the other hand, cells tuned to the same orientation at 22.6 c/deg have a much narrower orientation bandwidth of about 25°, which means that they respond best to elements oriented from about 145° through 170°. This explains why the cells tuned to 22.6 c/deg at 157.5° do not respond at all to the middle texture, even though the cells at 135° and 22.6 c/deg do show some response.

The responses of the simulated complex cells to the remaining images are shown in Figures 3.17 on the next page through 3.19 on page 107. As mentioned above, for these remaining cases, I only present the responses to every other orientation. The responses of model cells that are not shown follow the same general trends as those shown here.

Figure 3.17 on the next page presents the outputs in response to the second test image. The outer texture region in this case produced responses in model cells tuned to many orientations, especially at the lower spatial frequencies. This is especially visible in the response maps of cells with preferred frequencies of 1.0–4.0 c/deg; these cells display roughly comparable responses in the outer texture region at most orientations. This is because the texture itself is fairly coarse, and the grains have a certain degree of roundness that leads to responses in cells of many orientation preferences. On the other hand, the central texture square in this case produced relatively localized activity in cells tuned to the near-horizontal orientations (90°) and the higher spatial frequencies (e.g., 5.7–22.6 c/deg).

Figure 3.18 on page 105 shows the outputs in response to the third test case, the image shown in Figure 3.15(c) on page 100. This set of responses follows the same trends as those of the previous test cases. Once again, there is fairly localized activity across

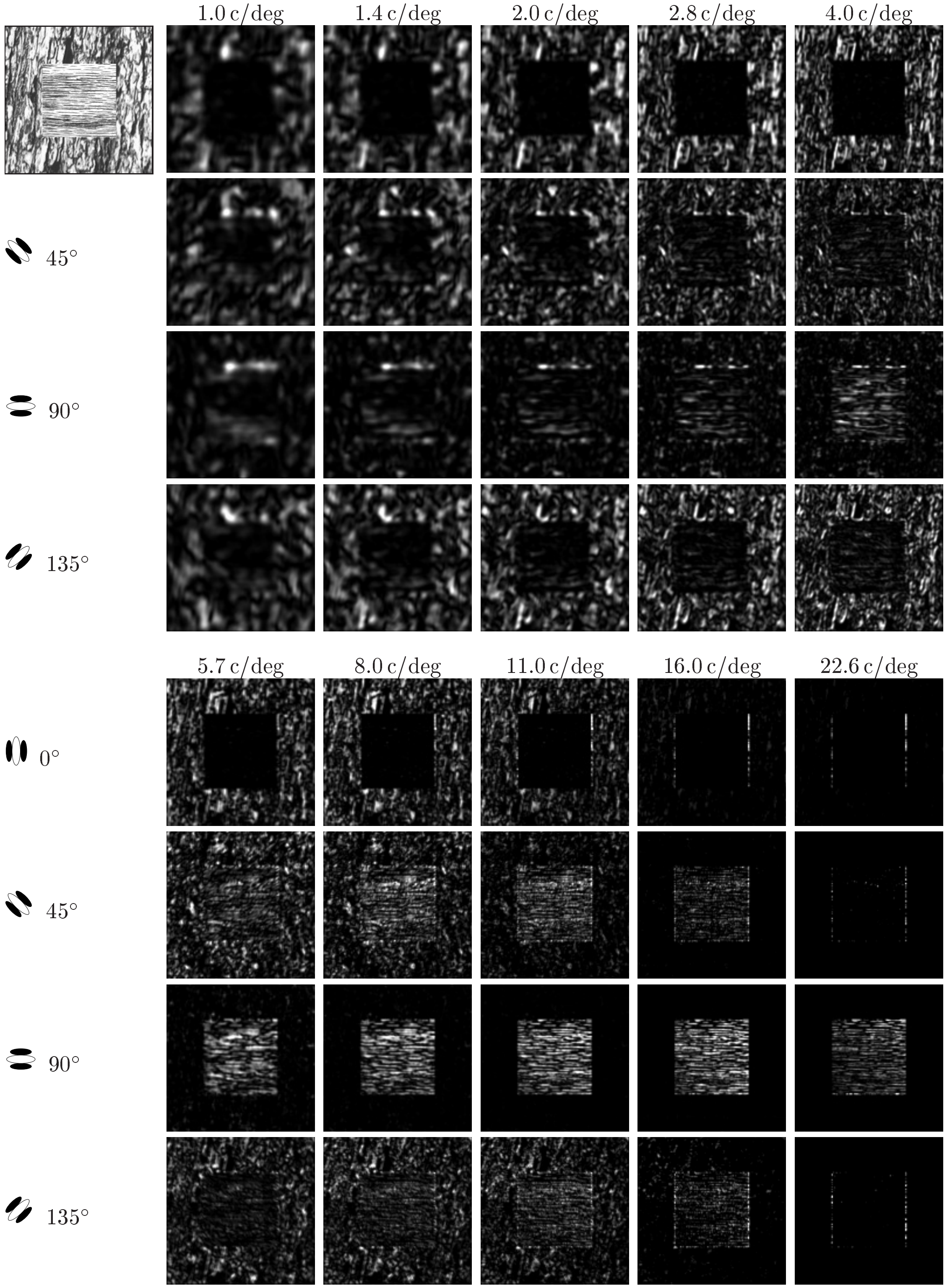


Figure 3.17: Maps of responses of simulated complex cells to the image shown in Figure 3.15(b) on page 100. As in the previous figure, all maps have 512×512 elements, and each point represents the level of activity of a simulated cell with its receptive field centered on the corresponding image location. Brighter points indicate greater activity. Only every other orientation is shown. The input image is shown at the top left; the rest of the top row contains response maps for cells at 0° orientation.

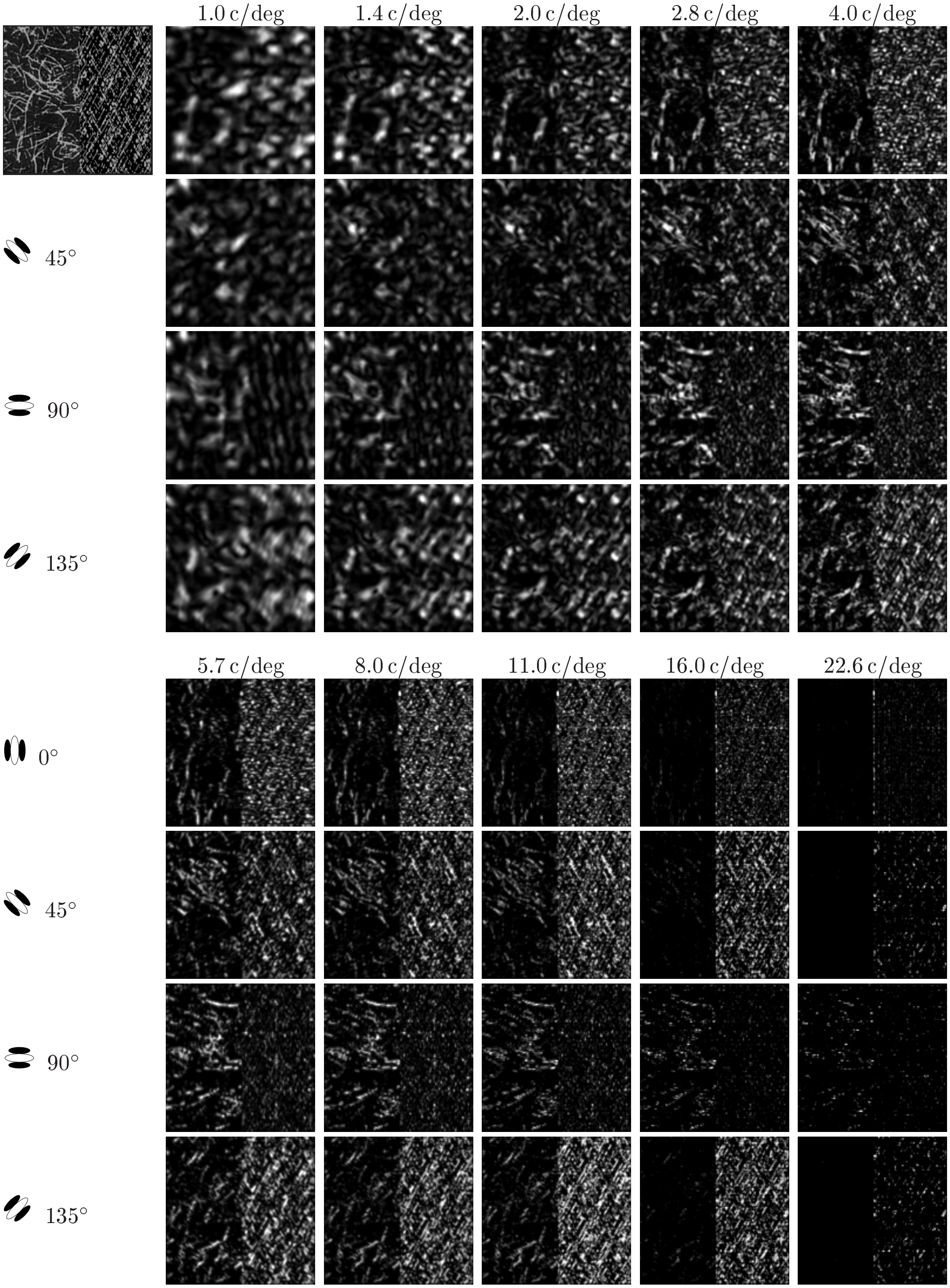


Figure 3.18: Maps of responses of simulated complex cells to the image shown in Figure 3.15(c) on page 100. The format of this figure is identical to that of the previous figure.

one of the texture regions in cells tuned to a subset of the possible frequency/orientation combinations. In this case, the right-half texture produced the clearest responses in cells tuned to oblique and near-oblique orientations, with only sparse activity in cells tuned to 90° at frequencies of 4.0–22.6 c/deg. Cells tuned to low frequencies (1.0–2.8 c/deg) tended to produce responses that do not discriminate at all well between the two texture patches.

Finally, Figure 3.19 on the following page shows the outputs in response to the last input image of Figure 3.15(d). The results in this case are much as would be expected given the previous three test cases. One surprising aspect is the relatively strong activity to the upper-left texture square shown by cells tuned to 2.0 c/deg at 90°. The texture in that region has dominant orientations at 45° and 135°, so the fact that cells tuned to 90° responded is at first unexpected. However, close inspection of the image reveals that the straw used in the weave of this texture has two slightly dissimilar shades of gray, and the resulting weave displays slight horizontal banding. This horizontal banding is sufficient to trigger activity in cells tuned to the horizontal orientation. In other respects, the results for this test case show good discriminability for the different texture patches.

The first prediction is thus confirmed by this part of the experiment. In all of the input images, there is at least one combination of spatial frequency and orientation that produces strong activity over the entire extent of one area of texture and simultaneously only moderate to weak activity in other areas of texture. This kind of segregation of responses is desirable for segmentation purposes, because it makes it simple to find regions of homogeneous texture.

3.4.3 Results for Prediction Two: Responses to Curved Textured Surfaces

The examples presented in the previous paragraphs all involved textures viewed head-on, in a fronto-parallel orientation. How will the model complex cells respond to images of textured surfaces that have curvature or slant? Intuitively, we expect to see texture gradients give rise to shifts in the sets of complex cells responding to the same surface. Cells tuned to high spatial frequencies should respond best to areas of the texture containing smaller texture elements (presumably corresponding to areas of the surfaces located farther away from the viewer), and cells tuned to low spatial frequencies should respond best to areas of the input that contain larger texture elements (presumably corresponding to those areas located closer to the viewer).

The four images used for this test are shown in Figure 3.20 on page 108. The responses for each of these inputs is shown in Figures 3.21 through 3.25. As is the case in the previous section, each point in the response maps represents the activity of a complex cell with its

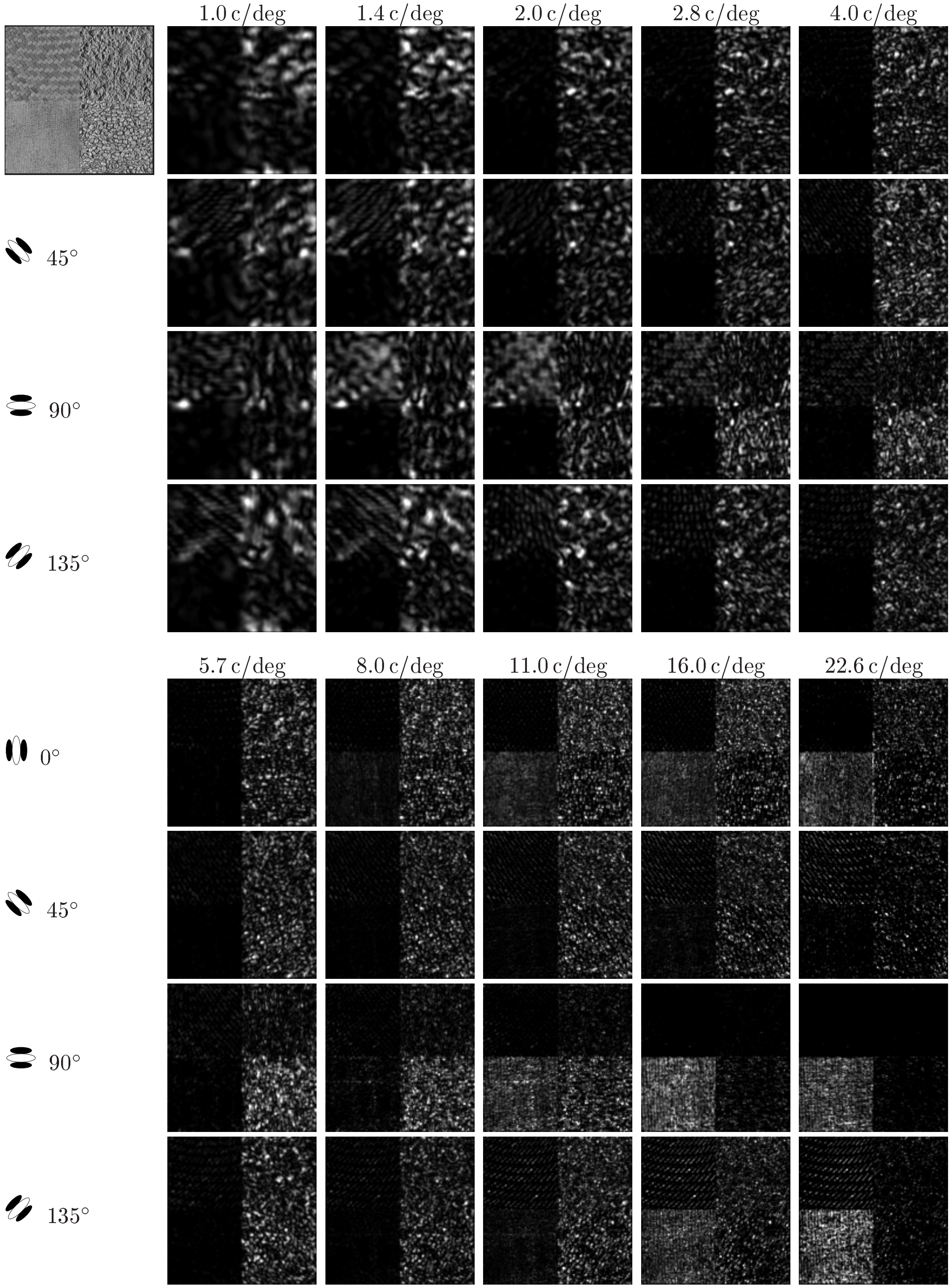


Figure 3.19: Maps of simulated complex cell responses to the image shown in Figure 3.15(d) on page 100. The format of this figure is identical to that of Figure 3.17.

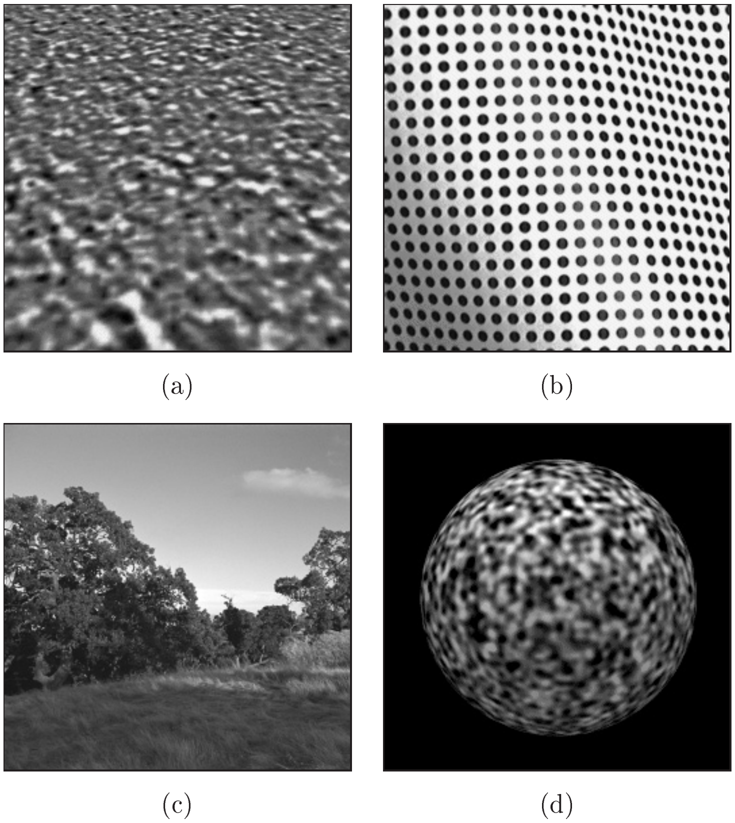


Figure 3.20: Images used to test the simulated complex cells' responses to slanted and curved textured surfaces. (a) An artificial, granite-like texture mapped in software onto a flat surface that is slanted away vertically. (b) A dot texture printed onto white, undulating paper. (c) A nature scene. (d) A random-noise texture mapped onto a sphere.

receptive field centered at that location in the input image. Degree of activity is coded as intensity in the maps. To conserve space, only every other orientation is shown; the results for the other orientations follow the same trends as for those presented here.

Figure 3.21 on the following page presents the results for the first input image, Figure 3.20(a). The simulated cell activities depicted in the response maps of Figure 3.21 clearly exhibit variation across spatial frequencies. Beginning with the column of responses for 1.0 c/deg, and looking at successively higher frequencies in the figure, we see that the bulk of the complex cell responses move upward within the maps as spatial frequency increases. In other words, the cells tuned to lower spatial frequencies (e.g., 1.0–1.4 c/deg) respond best near the bottom of the textured plane; cells tuned to middle frequencies respond best in the middle; and cells tuned to the highest frequencies (11–22.6 c/deg) respond best near the top of the water image. This is exactly as predicted for complex cell responses on an input containing a slanted, textured surface. Unlike the complex cell outputs for flat textures in the previous section, the outputs for slanted

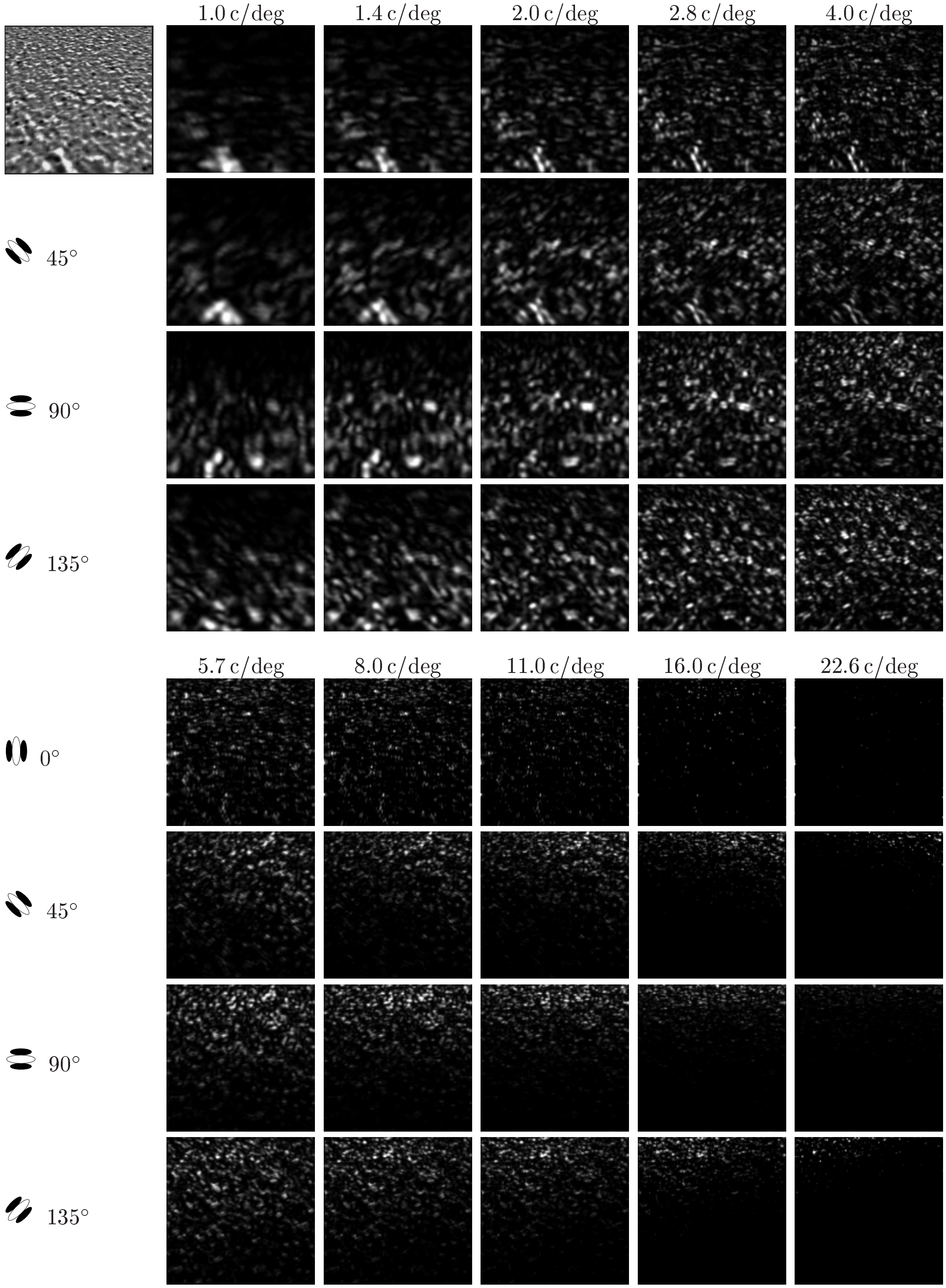


Figure 3.21: Maps of responses of simulated complex cells to the image shown in Figure 3.20 on the page before(a). Each response map has size 512 × 512 elements; each point in the maps represents the level of activity of a simulated cell with its receptive field centered on the corresponding image location. Brighter points indicate greater activity. Only every other orientation is shown. The input image is shown at the top left; the rest of the top row contains response maps for cells at 0° orientation.

textures are not homogeneous over a region containing a single surface.

The results for the second case are shown in Figure 3.22 on the next page, using the image of an undulating surface shown in Figure 3.20(b). The pattern of results for this case are similar to those of the first test case. Once again, the activity evoked in the simulated complex cells varies systematically over the surface, with cells tuned to higher frequencies responding more strongly at locations where the surface texture is smaller. Those are the locations where the surface slants away from the viewer. The pattern of responses is thus as expected.

One aspect of the response maps in Figure 3.22 is somewhat unexpected: some of the cell responses show dense covering of the input, at first glance encompassing the entire image. A good example of this occurs in the filters tuned to 2.0 c/deg at 90°. This may at first be surprising, because the surface contains variations in texture sizes and thus is expected to give rise to more selective activity. However, close examination shows that even this response map does not show complex cell activity over the entire surface—for instance, the lower left corner of the surface shows significantly reduced activity.

A number of the maps in Figure 3.22 display contrasting variations in response values along their edges. For example, the maps for the 90° orientation and frequencies 1.4 through 2.8 c/deg have a band around the edges that does not match the rest of the responses. This edge effect is a result of the reflection-based padding performed prior to computing filter responses. (The padding procedure is described in Section 3.3.3.) On occasion, an image can have texture elements positioned at its edges in such a way that when the pixels near the edges of the image are reflected, they introduce bands of spurious high or low spatial frequencies. This problem is actually rare for most input images. The image in Figure 3.22 is unusual in that it contains a regular texture whose elements are cut off irregularly at the image edges. Consider, for example, the dots of the texture at the top left border of the image. Many of the dots composing the texture in that region lie immediately next to the image border. When the image is reflection-padded prior to computing filter responses, the dots end up extended in length as illustrated in Figure 3.23 on page 112. The larger-appearing texture elements comprise an area of lower spatial frequencies, with the result that the complex cells at 1.4 c/deg and 90° show activity in a band in the top left region, whereas the cells at a higher frequency of 4.0 c/deg do not show activity in that region.

Edge effects are a common problem in image processing, and it is extremely difficult to avoid all possible spurious responses at the edges of images. I have found that reflection-based padding is the most effective for simulating receptive fields of the kind used in the present research, but the method is not perfect. Edge effects of the kind visible in

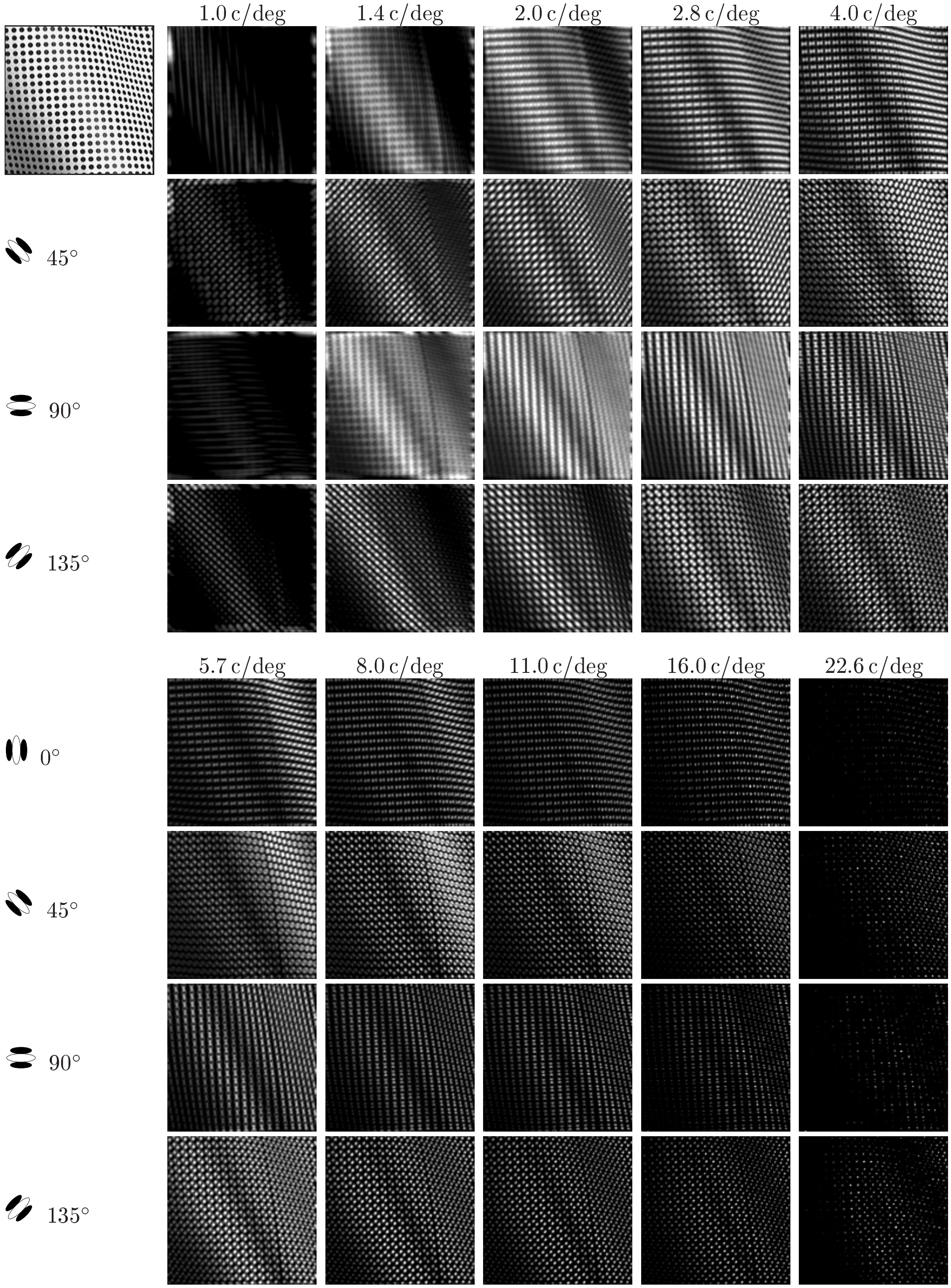


Figure 3.22: Maps of responses of simulated complex cells to the image shown in Figure 3.20 on page 108(b). The format of this figure is identical to that of the previous figure.

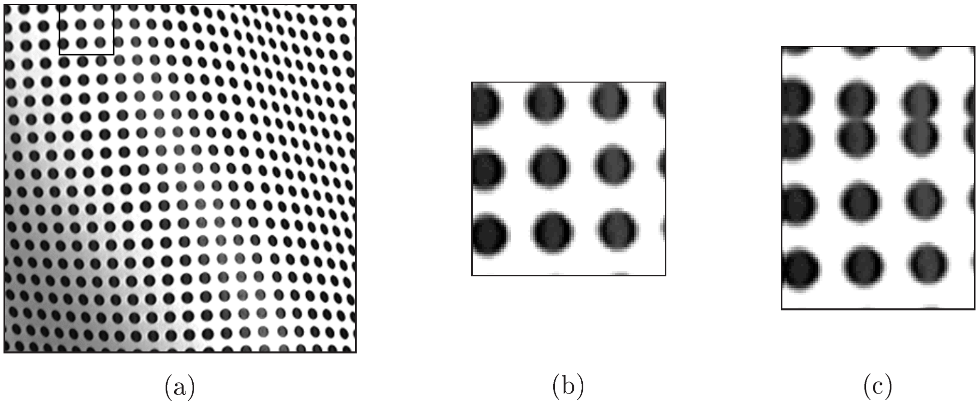


Figure 3.23: Closeup of the edge effect that is introduced by reflection-based padding for the image shown in Figures 3.20(b) and 3.22. (a) The original image, with a small region outlined near the upper left border. (b) Closeup of the outlined region in the image. (c) The result of reflection-based padding along the upper border of the image section. Note how the dots are turned into elongated elements. This becomes a conspicuous feature in an otherwise highly regular texture. As a result, the spatial-frequency content near the image border is slightly altered, leading to an edge effect in the filter response maps.

Figure 3.22 are most often encountered with regular, structured textures.

The third test case is shown in Figure 3.24 on the next page, using the image of a natural scene shown in Figure 3.20(c). Once again, this input produced activity in different subsets of simulated complex cells at different locations in the input, in a way that does not correlate directly with entire surfaces. However, the differences are small for this case. In fact, the entire set of responses is relatively sparse. Note, for example, the absence of activity over the ground plane in the image.

The reason for this sparseness of activity is that the textures in the input are extremely fine, making it difficult for the simulated complex cells to detect them. This illustrates a general difficulty with simulations of this kind: it is possible for an input image to have textures that are either too coarse or too fine to allow a good span of simulated filter responses. The primary reason for this is the limited number of samples of spatial frequencies and orientations available in the simulation. The result is that all of the texture energy in the image may end up concentrated at one end of the spectrum (in this case, at the high-frequency end). A real visual system has much finer sampling in frequency and orientation, and consequently is much less prone to this problem.

The responses to the remaining image, Figure 3.20(d), are shown in Figure 3.25. Once again, the complex cell response maps display the expected pattern of activities for the textured surface. The responses at lower spatial frequencies are primarily localized near the center of the textured surface. Responses at higher frequencies move progressively

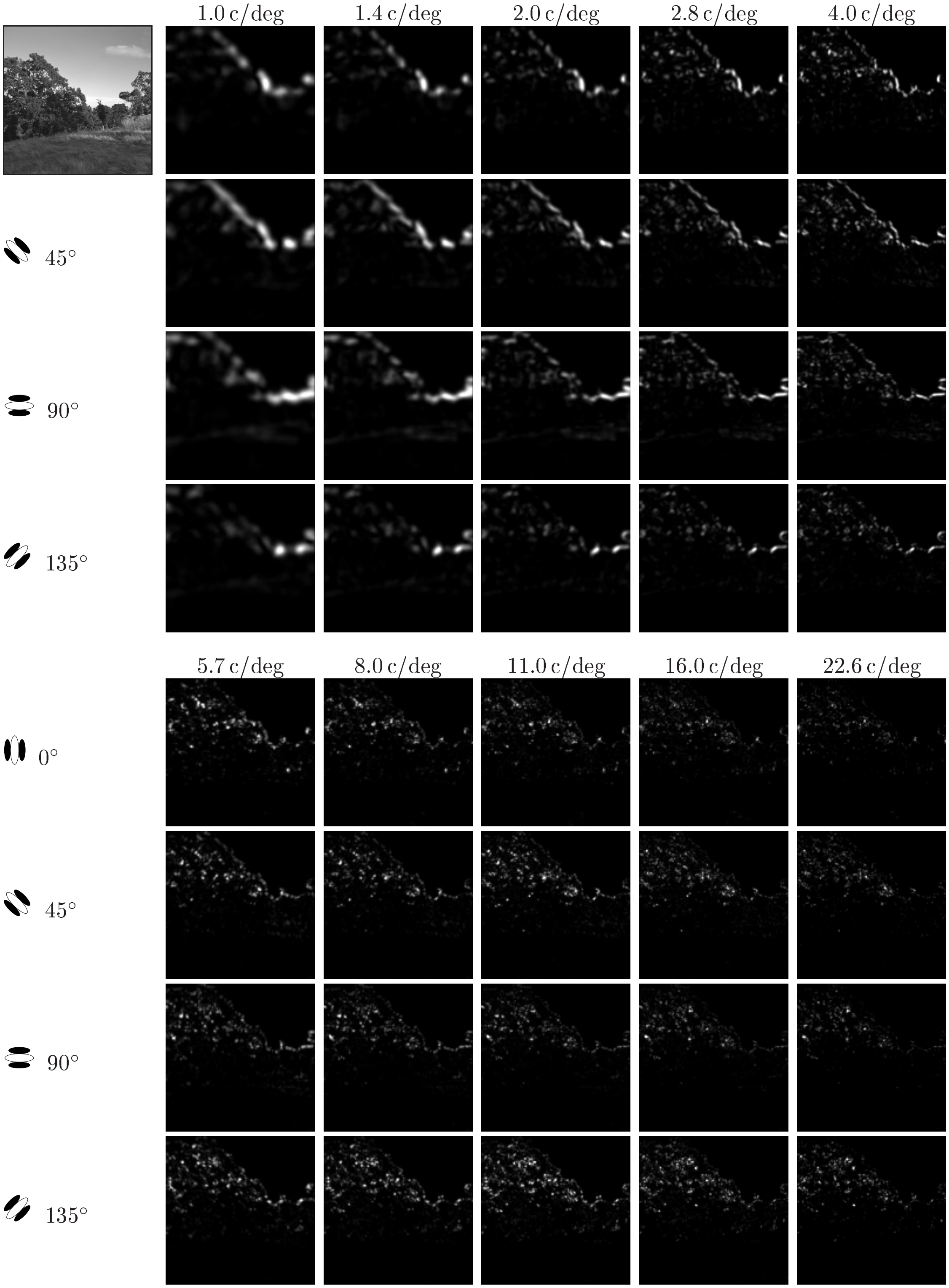


Figure 3.24: Maps of responses of simulated complex cells to the image shown in Figure 3.20 on page 108(c). The format of this figure is identical to that of Figure 3.21 on page 109.

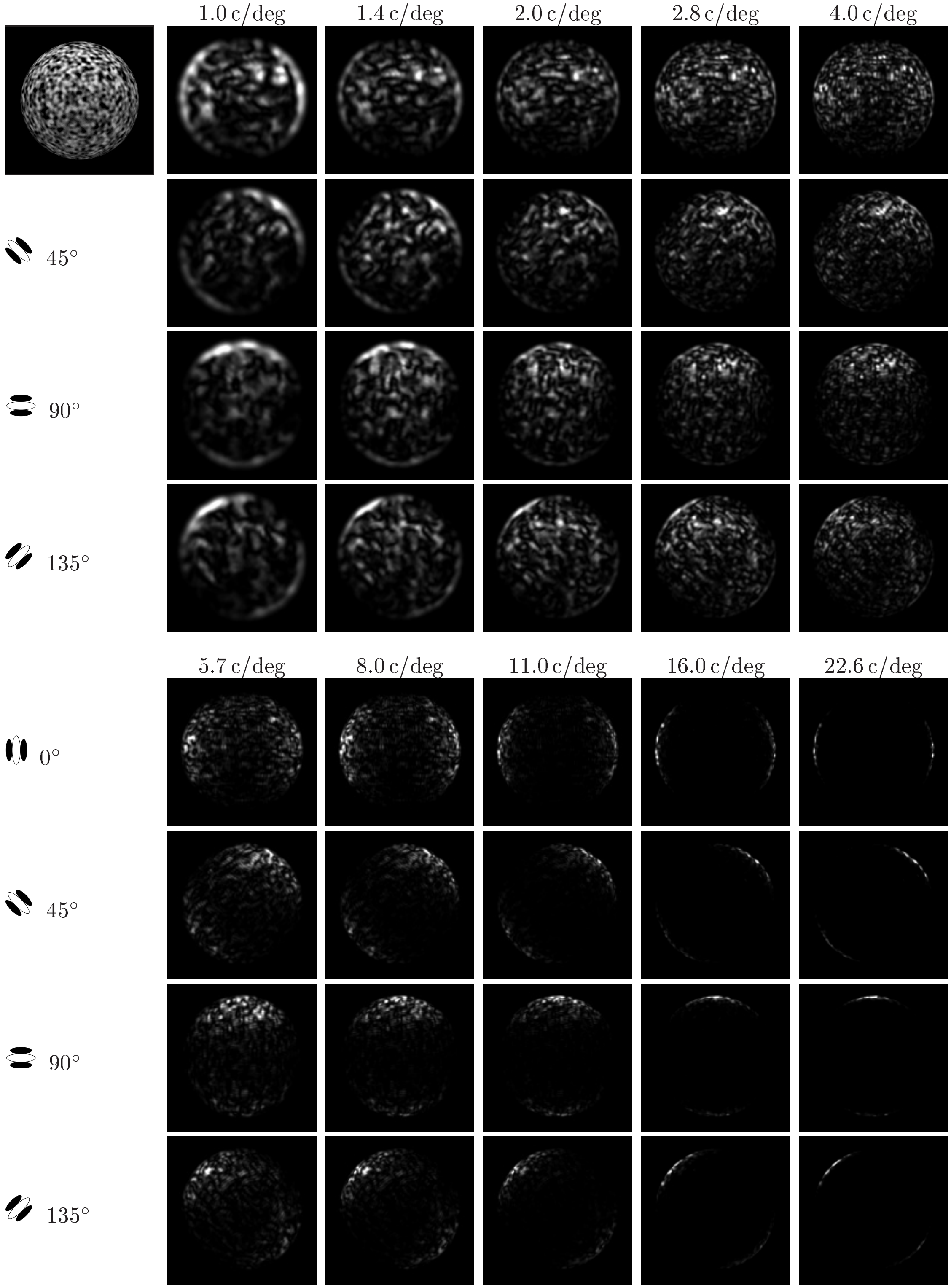


Figure 3.25: Maps of responses of simulated complex cells to the image shown in Figure 3.20 on page 108(d). The format of this figure is identical to that of Figure 3.21 on page 109.

towards the edges of the surface.

The second prediction, then, is supported on the basis of these results. Gradients in textures arising from changes in surface orientation lead to activation of different subsets of cells responding to the same region of texture.

3.4.4 Results for Prediction Three: Responses to Regions Devoid of Texture

The last prediction concerns image areas lacking texture. Intuitively, one expects that complex cells will not respond to blank areas, where there are no variations in light/dark—simple and complex cells are, after all, specifically responsive to *changes* in contrast in the localized region of their receptive fields.

Two of the input images used in testing the previous prediction, the image shown in Figure 3.20(c) and (d), already contained areas of texture and nontexture. To test the third prediction, it is sufficient to examine the responses to these two images. The responses are shown in Figures 3.22 and 3.25.

The plots in the figures show that the simulated complex cells generally did not respond in the areas lacking texture. In the case of the natural image whose responses are shown in Figure 3.22 on page 111, the untextured region is the sky in the upper half of the image; an examination of the response maps is enough to verify that none of the complex cells showed significant activity in that region. Similarly, none of the complex cells responses plotted in Figure 3.25 on the page before show activity in the blank area surrounding the textured sphere. Thus, the complex cells display the predicted behavior.

However, one aspect of the responses does not fit entirely the expected behavior. Note that the responses of cells tuned to the lowest spatial frequencies show a certain amount of spill-over into image areas that are actually devoid of texture. This is most evident in the responses shown in the first column (1.0 c/deg) of each figure, where there is activity relatively far away from the true borders between regions containing texture and regions devoid of texture.

At this point, the impact of this behavior on the rest of the model is unclear. Will it adversely affect later processing stages? I will return to this topic in later chapters. On balance, however, the responses of the complex cells support the third prediction: regions devoid of texture in an image do not lead to responses in the simulated cells.

3.4.5 Discussion of Experiment One

The first experiment, testing the simulation of model complex cells, has been successful. All three predictions have been largely supported:

1. Flat textures differing in average orientation, spatial frequency, or a combination of the two lead to responses in different subpopulations of the total set of complex cells. When the complex cell characteristics are well-matched to the texture characteristics, the cell responses over a texture region are relatively homogeneous.

2. Gradients in textures arising from changes in surface orientation lead to responses in different subsets of complex cells. These variations occur within single regions of texture.

3. Regions devoid of texture in an image do not lead to responses in the simulated complex cells.

3.5 Summary

This chapter has presented a computational model of the complex cells of the primary visual cortex. The simulation of the model produces a representation meant to resemble, in a highly simplified way, the outputs produced by complex cells in response to a visual input. The model does not need to be specifically tailored to respond to visual texture; complex cells have inherent properties that make them responsive to texture in an input. The behavior of the simulation in response to flat as well as slanted and curved textured surfaces has met expectations.

The model in this chapter represents a synthesis of two lines of research: the contrast-normalization model of simple and complex cells, and the Gaussian derivative model of simple cell receptive fields. These complementary models can be combined into an overall model of simple and complex cells that satisfies many constraints about real cortical neurons indicated by neurophysiological research. There are two novel aspects to the work presented in this chapter.

- It represents the first attempt to use Gaussian derivative receptive fields at multiple scales with the contrast-normalization model. Although at least one other model has used Gaussian derivatives in conjunction with contrast-normalized cells (Simoncelli & Heeger, 1997), at this time no published model has used multiple derivative orders. Doing so requires overcoming the challenge of achieving approximate tiling of the spatial-frequency domain in the context of Gaussian derivative receptive fields (cf. Section 3.3.2).

- The model developed here adheres closely to several qualities of actual cortical simple and complex cells. In particular, it includes correlations between preferred spatial frequency, spatial-frequency bandwidth, and orientation bandwidth that physiological research has shown exists in actual cortical cells (e.g. De Valois et al., 1982, 1985; Foster et al., 1985). This makes the work presented here unusual among models of visual processing for its attention to the constraint of biological reasonableness.

The next step in this research is to explore the use of the simulated complex cells as a basis for texture-based estimation of surface shape. That is the topic of the following chapter.

Chapter 4

A Model of Texture-Based Shape Estimation

The next step in this research is the development of a model of how complex cell outputs may be used both to segment visual regions in an input image and to estimate surface shapes within the segmented regions. Although these two processes are ultimately going to be combined, for purposes of exposition, it turns out to be important to examine first how surface shape may be extracted from visual texture.

In this chapter, I describe a model of texture-based perception of surface shape. It is based on prior work by Sakai and Finkel (1995). The model operates by characterizing the spatial-frequency spectrum of a surface in an image using a certain set of measures, and then tracking changes in this set in order to infer changes in the shape of the physical surface. In addition to reimplementing their approach using, as a starting point, the complex cells presented in Chapter 3, I have developed several enhancements in an effort to improve the model's performance. The enhancements apply to how averages are computed during intermediate steps, and how the final integration step combines local estimates into an overall estimate of surface shape.

The model presented in this chapter is admittedly not a general solution to the problem of texture-based shape estimation because it is predicated on a number of simplifying assumptions about the nature of the visual input. Nevertheless, within its limitations, the model demonstrates it is feasible to use a relatively simple approach that may ultimately serve as the basis for a more general solution. The implementation in this chapter also assumes that only a single surface is present in the input, but this limitation is removed in the final system described in Chapter 6.

This chapter is organized as follows. Section 4.1 summarizes orthographic projection and how textures are distorted under this type of image projection. In Section 4.2, I describe Sakai and Finkel's (1995) model of texture-based shape estimation, and in Section 4.3, I present a reimplementation of their model in the context of the present framework. Section 4.4 then covers Experiment Two, which tested the basic capabilities of the simulation. Finally, Section 4.5 summarizes the results of this chapter.

4.1 Image Projection and Texture Properties

Before moving on to the details of the model of shape estimation used in this research, it is important to discuss certain background topics in order to clarify some assumptions made in the model. The assumptions concern the nature of surface textures and how surface texture is distorted during the process of creating an image. The topics discussed in the following sections are:

1. What type of image projection is assumed in the model? What implications does this have for the model's operation and generality?
2. What are the formal relationships between a textured surface and the spatial-frequency content of its projection in an image?
3. What assumptions about the properties of surface texture are made in the model? What implications does this have for the model's generality?

4.1.1 Texture Changes Under Orthographic Image Projection

When a visual system constructs an image of a scene, the projection of light rays from objects in the scene to the imaging surface (a camera sensor, or the retina) will obey certain geometrical rules. Two different models of the geometry of image formation are in common use in vision research: perspective projection and orthographic projection. The former is a more realistic formulation of imaging for cameras and eyes; the latter is a simplifying approximation that can be used in certain situations.

The Perspective and Orthographic Image Projection Models

Perspective projection is the model closest to everyday viewing conditions and to the process of image formation by a camera (Horn, 1986; Jain et al., 1995; Nalwa, 1993). It is illustrated in Figure 4.1 on the following page. Perspective projection is based on the notion of an ideal pinhole camera: an infinitely small pinhole is placed at some distance in front of an imaging surface inside a box. Straight rays of light travel from points in the scene through the pinhole and onto the imaging surface. The pinhole is the center of projection, and a three-dimensional world coordinate system spanned by unit vectors $\mathbf{x}_w$, $\mathbf{y}_w$, and $\mathbf{z}_w$ is placed at this point. The image plane is assumed to be parallel to the $\mathbf{x}_w$ and $\mathbf{y}_w$ unit vectors, and located at a distance f along the optical axis, $\mathbf{z}_w$. To avoid the inversion that is introduced by the geometry shown in the upper half of Figure 4.1, it is customary to pretend that the image plane is in front of the center of projection at

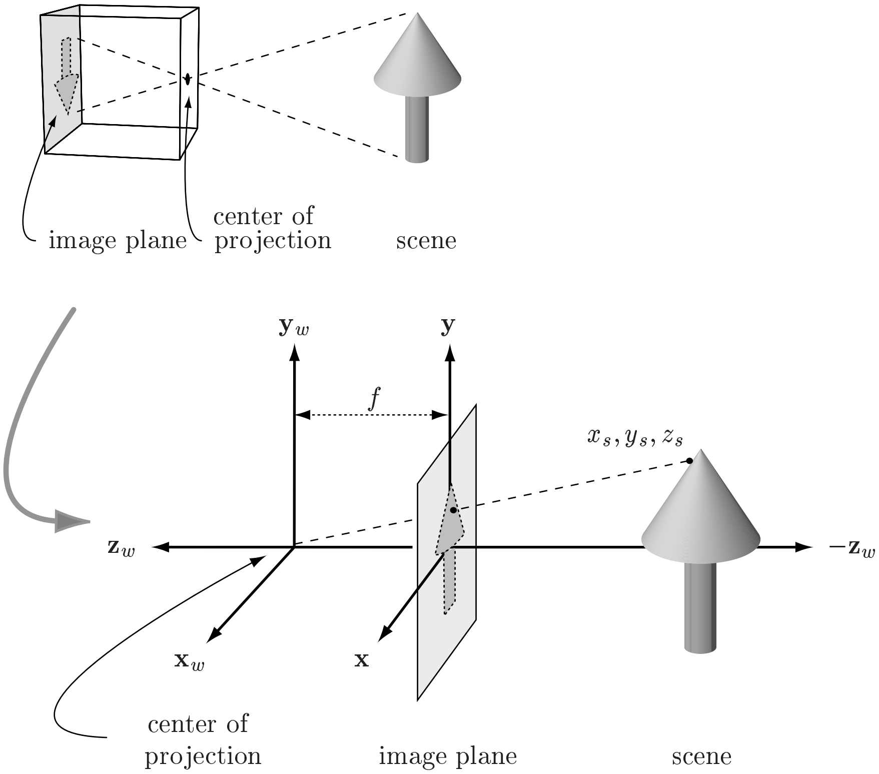


Figure 4.1: The basic model of perspective projection is based on an ideal pinhole camera, shown in the upper diagram. Rays of light from points on the scene pass through an infinitesimal pinhole (the center of projection) to land on the image plane. This imaging configuration creates an inverted image, so for convenience, the image plane is assumed to lie in front of the center of projection at a distance f, as shown in the lower diagram. The world coordinate system is spanned by unit vectors $\mathbf{x}_w$, $\mathbf{y}_w$, and $\mathbf{z}_w$ and is placed at the center of projection, and the image plane is given another coordinate system, $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$.

the same distance f, as shown in the lower half of Figure 4.1. In this configuration, the position of a point (x_s, y_s, z_s) on a surface in the scene is projected to a point (x, y) on the image plane according to the equations

$$x = x_s \frac{f}{z_s} \quad \text{and} \quad y = y_s \frac{f}{z_s}. \tag{4.1}$$

Thus, the fundamental effect caused by perspective projection is the scaling of dimensions in the scene in proportion to their distance z_s from the image plane.

Orthographic projection is a simplified projection model in which the image of a scene is projected onto a plane by parallel rays of light orthogonal to the plane. This is illustrated in Figure 4.2(a) on the next page. The equations for orthographic projection are simply

$$x = x_s \quad \text{and} \quad y = y_s, \tag{4.2}$$

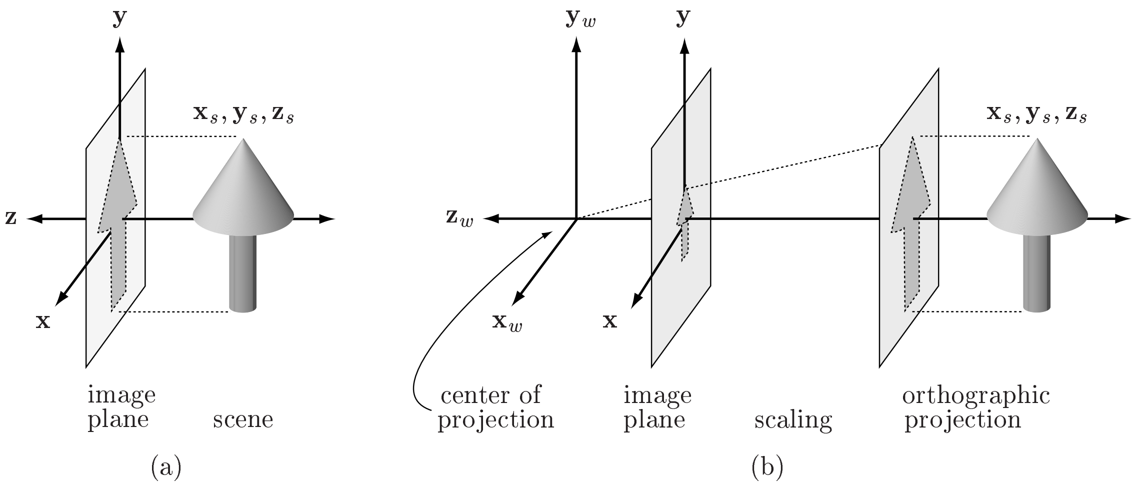


Figure 4.2: (a) The basic model of orthographic image projection. Each point in the scene is projected onto the image plane by straight, parallel rays of light that are orthogonal to the plane. (b) Orthographic projection can be combined with a constant scaling factor to give an approximation to perspective image projection.

where (x_s, y_s) are the coordinates of a point in the scene and (x, y) are the corresponding coordinates in the image plane.

Together with a scaling factor, orthographic projection can be used as an approximation to perspective projection. Two conditions must be satisfied: variations in distances within the scene must be small compared to the average distance between the objects and the image plane, and the objects in the scene must lie close to the optical axis (Nalwa, 1993). In that case, the factor f/z_s in the perspective equations becomes less sensitive to distances in the scene z_s, and can be approximated by a constant scaling factor m. This configuration is illustrated in Figure 4.2(b). The perspective projection equations then simplify to $x = mx_s$ and $y = my_s$, where $m = f/z_0$, and z_0 represents the average distance between objects in the scene and the camera. The scaling factor is often taken to be $m = 1$ for convenience; in that case, the equations reduce further to $x = x_s$ and $y = y_s$, which are the equations for plain orthographic projection.

Distortions of Texture Under Perspective and Orthographic Projections

In Section 2.3.1, I briefly described two geometric effects that can distort the appearance of texture a when the image of a three-dimensional physical surface is formed in perspective projection. One is the *perspective* effect; it arises from the change in distance from the physical surface to the viewer. Perspective causes a change in the local scale of the projected texture as compared to the actual physical texture. The second effect is *foreshortening*. It arises from a change in the orientation or curvature of the physical surface with respect to the line of sight of the viewer. Foreshortening introduces changes in the sizes, densities and shapes of texture elements in the projected image as compared to the physical texture (Cumming et al., 1993; Nalwa, 1993). The perspective and foreshortening effects together can produce a variety of distortions in the appearance of texture in an image. Much of the literature on texture-based shape estimation has explored how changes in texture qualities in an image, such as texture density, element size, and element orientation are related to the shape and orientation of a physical surface in three-dimensional space.

The perspective effect does not occur under orthographic projection; consequently, the information that distortions in texture can provide are different when orthographic projection is assumed than when perspective projection is assumed (Blake et al., 1993; Cutting & Millard, 1984; Stevens, 1983; Todd & Akerstrom, 1987). Three examples serve to illustrate this:

Scaling: In perspective projection, this refers to a change in size of texture elements in the image without deformation of their shapes. Assuming the surface texture has homogeneous element sizes, the apparent change in size is a direct result of the perspective effect: changes in distance across an extended surface cause elements farther away to have a smaller projection in the image (Equation 4.1 on page 120). Figure 4.3(a) on the following page provides an illustration of this: the circles, which project into ellipses in the image, become smaller in the distance. Under orthographic projection, the perspective effect does not exist and the scaling cue is not available, because distance from the viewer to points on the surface are assumed to be approximately constant. (Foreshortening does cause changes in the heights of texture elements, and thus their perceived sizes do change in orthographic projection, but this is technically distinguished from scaling.)

Density: Texture density is usually defined as the spatial distribution of texture element centers in the image (Blake et al., 1993). For a texture with homogeneous element density, apparent changes in the density in the image can result from either the perspective or foreshortening effects. Under perspective projection, the number of elements per solid angle of viewing increases with increasing distance and increasing curvature on a surface, which means that apparent texture density also increases. Under orthographic projection, changes in density reflect only changes in surface curvature (again, assuming the texture pattern is homogeneous everywhere on the surface). The changes also occur only in the direction of tilt: under orthographic

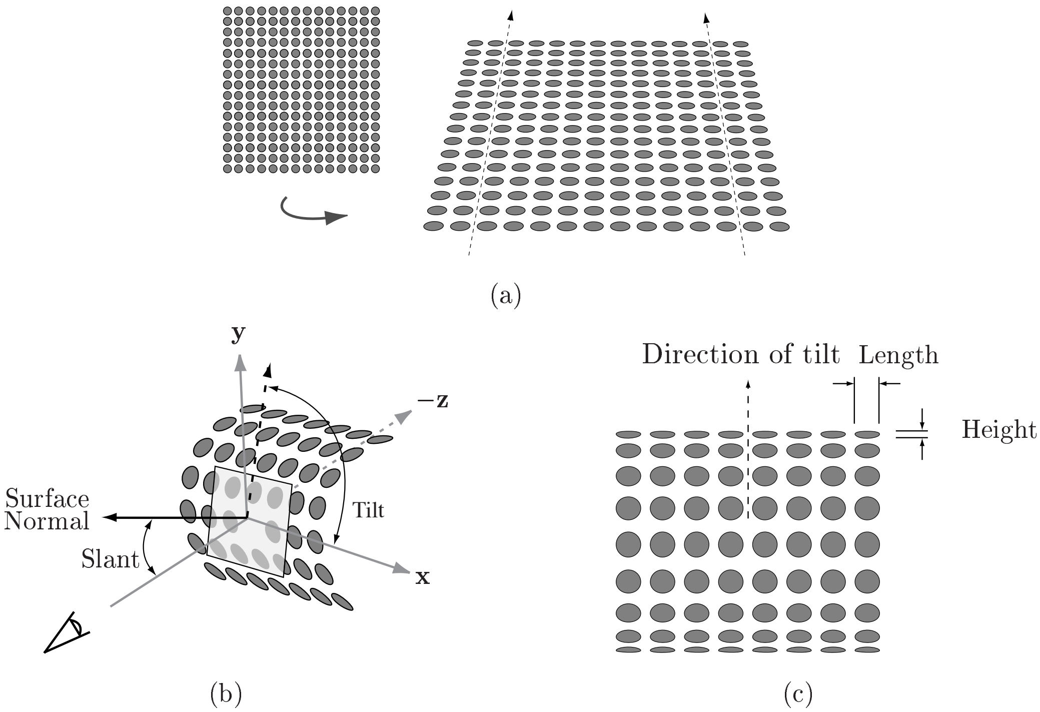


Figure 4.3: (a) Example of texture distortions under perspective projection. A flat plane covered with a regular arrangement of circles is viewed at an angle. Parallel lines on the surface are not parallel in the image projection; instead, they appear to converge towards a point in the distance. Slant is constant across the surface. Texture elements are primarily scaled in size, and the density of elements appears to increase with increasing distance to points on the surface. (b) Illustration of the meaning of slant and tilt. An imaginary set of **x**-**y**-**z** axes is placed at some point on the surface, with −**z** aligned with the line of sight. Slant is the angle made by the surface normal with respect to the line of sight; tilt is the angle, in the **x**-**y** plane, that the normal makes with the **x**-axis. (c) Example of texture distortions under orthographic projection. A cylindrical surface covered with a regular arrangement of circles is viewed head-on. Slant increases along the surface moving towards the top and bottom, and tilt is the direction in which slant increases most rapidly. Increasing slant causes texture elements to be increasingly foreshortened in the direction of tilt but not in the perpendicular direction—note how the lengths of the texture elements remain the same. Lines parallel to the direction of tilt remain parallel in the image projection.

projection, density remains unchanged in the direction perpendicular to the direction of tilt.

Compression. This refers to a change in the height-to-width aspect ratio of texture elements in the projected image. If the physical surface is covered with a homogeneous texture, the change makes texture elements appear flatter. It is caused by foreshortening. Under both perspective and orthographic projection, the degree of compression is relatively stable for flat surfaces; for curved surfaces, compression increases with increasing slant away from the viewer. Under orthographic projection, compression affects the heights of texture elements in the direction of tilt, but unlike in perspective projection, the lengths of the elements in the direction perpendicular to tilt are not affected. Figure 4.3(c) on the page before illustrates this. The effects of compression are stated more rigorously in the next section.

Research on human perception of shape from texture has shown that the dominant cue to surface curvature is compression (Cumming et al., 1993; Todd & Akerstrom, 1987), regardless of whether perspective cues are available. This suggests that employing texture compression as a basis for estimating surface shape is a reasonable approach, even in the restricted case of orthographic projection. The use of texture compression is the basis for the model by Sakai and Finkel (1995) described below.

Under orthographic projection, the number of usable distortion effects is reduced, the effects of the texture distortions are simplified, and the information that can be extracted is more limited compared to perspective projection. In particular, information about distance across a surface is unavailable—all the distortions are related to changes in surface shape. Despite this disadvantage, there are reasons to use orthographic projection as an approximation to perspective projection. First, there is the significant advantage that the mathematics are much simplified. Second, orthographic projection is an approximation to a number of viewing situations. For instance, collections of objects in scenes are often viewed at distances relatively large compared to the changes in depth across object surfaces, in which case the projection can be considered approximately orthographic. An orthographic approximation may also be usable piece-wise in a scene even if the overall scene has perspective effects (Nalwa, 1993). And finally, some computer vision researchers studying shape from texture have found that perspective effects are often too small to be useful (Super & Bovik, 1995). Thus, although at first it may seem to be an oversimplification, in practice orthographic projection is not unreasonable, at least as a first step toward understanding a problem.

A further limitation that arises in orthographic projection is the tilt ambiguity: tilt direction is determined only up to 180°, such that a given surface may be seen as concave or convex. The textured cylinder in Figure 4.3(c) on the preceding page illustrates this problem: it is impossible to tell whether the middle of the surface is closer to the viewer or farther away. The ambiguity cannot be resolved based on information in orthographic projection alone; perspective cues or additional information or assumptions are needed. It is interesting to note, however, that in natural viewing situations, it is often fair to assume that surfaces are convex (i.e., bowled away from the viewer). As a general rule, trees, rocks, hills, body parts, buildings, and many other objects have a convex shape. A visual mechanism that extracts surface information from texture could have a built-in default assumption about the convexity of surfaces, allowing that assumption to be overridden if other information contradicted it. I am not aware of any evidence either for or against such a built-in default in texture-based shape estimation; however, this kind of bias towards a particular interpretation has been shown to occur in other aspects of vision. An example is the bias towards assuming a single lighting source shining on a scene from above (Ramachandran, 1988; Kleffner & Ramachandran, 1992).

4.1.2 Relationships Between Surface Texture and Image Spatial-Frequency Content Under Orthographic Projection

In Section 2.3.2, I mentioned that an alternative to measuring changes in texture elements directly is to determine instead the changes in the spatial-frequency spectrum across the projection of the surface in the image. But how, exactly, is the spatial-frequency content in the image related to spatial distortions of surface texture? This section provides an answer to this question by reviewing formal expressions relating surface texture to the approximate spatial frequencies in the image under orthographic projection. The goal is to make explicit how the spatial-frequency spectrum of a texture pattern is transformed under orthographic projection, because this is the underlying basis for the operation of the model examined in Section 4.2. What follows is based on the locally planar approximation given by Super and Bovik (1995), which is sufficient for the present purposes; a more exact treatment would require the differential geometry framework used by Gårding (1992b).

Assume the imaging configuration illustrated in Figure 4.4 on the next page. This is identical to the configuration shown in Figure 4.2(b) on page 121 with a scaling factor of one. Let p be a point on a smooth, physical surface in the scene. The tangent plane to the surface at point p will have a surface normal that will form an angle σ with the $\mathbf{z}$ axis, and the normal's 2-D projection in the image plane will form an angle τ with the positive $\mathbf{x}$-axis. Angle σ is slant, and τ is the angle of tilt. Define a local coordinate system on the tangent plane at p with unit vectors $\mathbf{x}_s$, $\mathbf{y}_s$, and $\mathbf{z}_s$. Pick the direction of $\mathbf{z}_s$ to be aligned with the surface normal at the point p, pick the direction of $\mathbf{x}_s$ to be in the direction of tilt, and, pick the direction of $\mathbf{y}_s$ to be perpendicular to both $\mathbf{x}_s$ and $\mathbf{z}_s$. This creates an orthonormal coordinate system at p.

For orthographic projection with a scaling factor of one, the transformation $\mathbf{A}$ of a point (x_s, y_s) on the surface to the corresponding point (x, y) in the image plane is given simply by a change of basis. The change of basis involves two steps. First, the coordinate system $\mathbf{x}_s$-$\mathbf{y}_s$-$\mathbf{z}_s$ is rotated about $\mathbf{y}_s$ by the slant angle σ to align the surface normal with the $\mathbf{z}$-axis. Second, the whole frame is rotated about $\mathbf{z}_s$ by the tilt angle τ. Expressed in

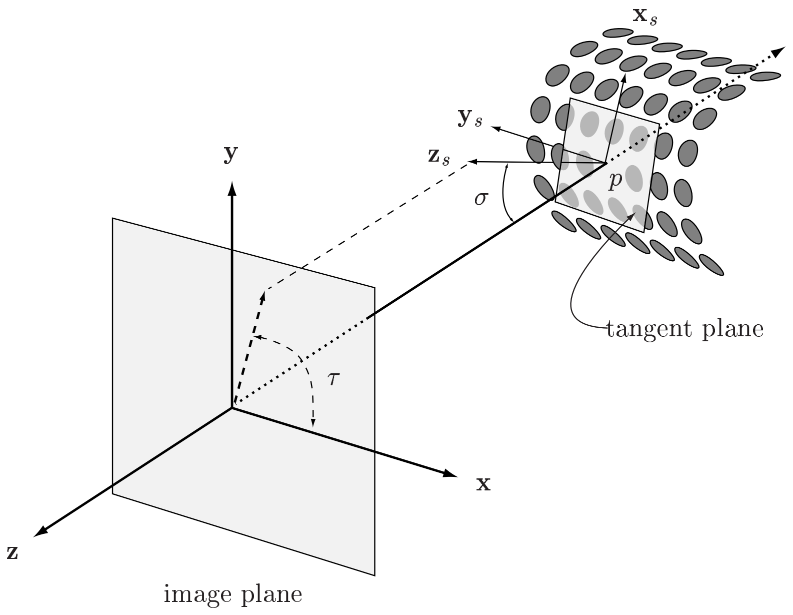


Figure 4.4: Illustration of the coordinate frame configuration used here for orthographic projection of a single textured surface. The image plane is centered at the world coordinate frame **x**, **y**, **z**, with the **z**-axis facing the viewer. A local coordinate frame $\mathbf{x}_s$, $\mathbf{y}_s$, $\mathbf{z}_s$ is placed at some point of interest p on the surface, with the $\mathbf{z}_s$ axis coincident with the normal to the surface at p. The $\mathbf{x}_s$-axis is aligned in the direction of the surface gradient, which in the image projection is the direction of tilt. The angle of slant σ and angle of tilt τ specify the orientation of the surface normal with respect to the world coordinate system.

matrix notation, this is

$$\begin{aligned}\begin{bmatrix} x \\ y \end{bmatrix} &= \mathbf{A} \begin{bmatrix} x_s \\ y_s \end{bmatrix} \\ &= \begin{bmatrix} \cos\sigma\cos\tau & -\sin\tau \\ \cos\sigma\sin\tau & \cos\tau \end{bmatrix} \begin{bmatrix} x_s \\ y_s \end{bmatrix}. \end{aligned} \tag{4.3}$$

Subscript s indicates the frame on the surface. Equation 4.3 shows that the effect of orthographic projection in the spatial domain is to multiply the $\mathbf{x}_s$ component of a pattern by a factor of $\cos\sigma$ (Super & Bovik, 1995). Since the $\mathbf{x}_s$ component points in the direction of tilt, this means that orthographic projection compresses spatial patterns by $\cos\sigma$ in the direction of tilt. The $\mathbf{y}_s$ component is unaffected.

Now, assume that the texture on the surface can be thought of as being painted on; that is, consisting only of variations in surface reflectance. This is not strictly valid for all textures, since many are actually due to variations in surface structure such as pits and raised regions; however, in many cases even these textures can be modeled approximately as variations in surface reflectance (Super & Bovik, 1995). We can then represent the texture pattern in a neighborhood of some point $\mathbf{p}_s = (x_s, y_s)$ on the tangent plane at p by a function $T_s(\mathbf{p}_s)$. The image intensity at the corresponding point $\mathbf{p} = (x, y)$ in the image plane can be approximated as a function $T(\mathbf{p})$ that is proportional to the reflectance at that point on the surface and is given by

$$T(\mathbf{p}) = \mathbf{k}{\cdot}T_s[\mathbf{p}_s(\mathbf{p})]. \tag{4.4}$$

Here, the value of $\mathbf{p}_s(\mathbf{p}) = \big(x_s(x, y), y_s(x, y)\big)$ is determined by using the inverse of transformation $\mathbf{A}$ from Equation 4.3 on the page before; $\mathbf{k}$ is a vector-valued constant of proportionality; and "·" indicates vector dot product. In reality, the image projection will also depend on the illumination falling on the surface and on the local surface orientation, and therefore the factor $\mathbf{k}$ will not be a constant but rather a function of $\mathbf{p}_s$. However, for present purposes it can be assumed that these effects vary slowly, and that the proportionality factor $\mathbf{k}$ can be assumed to be constant (Super & Bovik, 1995).

Consider now a simple texture pattern on the surface, composed of a single 2-D spatial-frequency component. In the spatial domain, this is a sine wave. How is the frequency of this sine wave changed when it is projected onto the image plane? Let the frequency of the sine wave be u_s in the $\mathbf{x}_s$ direction and v_s in the $\mathbf{y}_s$ direction, such that

$$\begin{aligned} T_s(\mathbf{p}_s) &= \cos[2\pi(u_s x_s + v_s y_s)] \\ &= \cos(2\pi \mathbf{u}_s \cdot \mathbf{p}_s), \end{aligned} \tag{4.5}$$

where $\mathbf{u}_s$ is simply a vector representation of u_s and v_s. The frequency $\mathbf{u}_s$ of this sinusoidal texture in the image is found from

$$T(\mathbf{p}) = \cos[2\pi \mathbf{u}_s \cdot \mathbf{p}_s(\mathbf{p})] \tag{4.6}$$

Although the frequency of the sinusoid in the surface coordinate frame will be constant, in the image the frequency may be a function of position because of the effects of projection. Therefore, we need to consider the instantaneous spatial frequency in the image plane. For a function of the form $\cos[2\pi F(\mathbf{p})]$, this frequency is $\nabla F(\mathbf{p})$. Thus, as given by Super and Bovik (1995),

$$\begin{aligned} \mathbf{u} &= \nabla[\mathbf{u}_s \cdot \mathbf{p}_s(\mathbf{p})] \\ &= \frac{1}{\cos\sigma} \begin{bmatrix} \cos\tau & -\cos\sigma\sin\tau \\ \sin\tau & \cos\sigma\cos\tau \end{bmatrix} \mathbf{u}_s \end{aligned} \tag{4.7}$$

This is the sought-after result. It shows that under orthographic projection, the frequency of the sinusoid is scaled in the tilt direction by $1/(\cos\sigma)$. Increasing the slant angle moves the spatial frequency to a higher value. The components in the perpendicular direction are not affected.

A more complicated texture will consist of a linear sum of spatial-frequency components. Each component will be a pattern described by Equation 4.4 on the preceding page. Each pattern will be transformed according to Equation 4.7, and the final result in the image will be the sum of the transformed components.

4.1.3 Assumptions About Texture Properties

The ability to recover surface shape and orientation from changes in the appearance of texture depends not only on the geometry of a surface and the image formation process, but also on the properties of the texture itself. Two basic questions can be asked about a texture. First, in what manner is the texture formed on the surface? Second, what statistical rules underlie the spatial pattern of the texture?

If a texture pattern is formed as a result of elements deposited on a surface, then if the elements are tall enough or wide enough, they may occlude each other when viewed from different orientations. This complicates the analysis of the effects of image projection on the texture. For this reason, most approaches to texture-based visual processing make the simplifying assumption mentioned in the previous section: the texture pattern is assumed to be painted on the surface, so that issues caused by reliefs, occlusions of elements, and so on can be ignored. This is the assumption I use in the present research as well.

The question of what statistical rules underlie the texture pattern is a fundamental and crucial one. It involves such issues as: are the texture elements distributed on the surface in a regular or a random manner? Do all elements have the same size and shape? Are they oriented randomly, or in a pattern? Properties such as these determine the methods that can be used to infer surface shape and orientation from texture. The dilemma is that one almost never has exact information about the statistical properties of a texture pattern. All models of shape estimation must make some assumptions about the properties.

Two broad categories of assumptions have been explored in research on texture-based visual processing (Rosenholtz & Malik, 1997). The assumptions and the approaches they imply can be summarized as follows:

1. Begin by assuming a certain statistical property of the surface texture, then measure that statistic in the neighborhood of a point in the image, and finally use deviations from the expected value to infer surface shape and orientation at that point.

2. Begin by assuming that a certain texture statistic is constant across the surface, then measure that statistic at different points in the image, and finally use the deviation between the values at different points to infer surface shape and orientation.

A common example of models that fall into the first category are those that assume texture *isotropy*, which is to say, that the orientations of texture elements on a surface are distributed evenly without a bias or dominant orientation (e.g., Blake & Marinos, 1990; Blake et al., 1993; Witkin, 1981). These models measure the orientation tendencies in the image, and use any differences from isotropy as a source of information about local surface orientation. They can therefore estimate surface orientation directly, at a given point, without having to compare different locations in the image. They can also build up an estimate of surface shape by integrating the results of measurements at different points in the image. Unfortunately, isotropy is violated by many natural textures: grass, hair, trees, bark on a tree, and many other common textures all have strong orientation biases, which makes this approach unsuitable for many kinds of ordinary surfaces.

The second category includes a wider range of models, such as those which use texture gradients (e.g., change in texture size or density), as well as Malik and Rosenholtz's (1994b, 1997) *affine texture distortion* model. All must make some sort of assumption about the *homogeneity* of the surface texture. The particular type of homogeneity assumed depends on the specific method. For instance, some approaches to estimating surface orientation work from measurements of the scaling gradient. Those methods must assume that the elements making up the surface texture all have nearly constant size. If the elements on the surface vary in size in some systematic manner, then the methods will fail to give an appropriate answer.

The model by Sakai and Finkel (1995) described in the next section is another model that falls into the second category listed above. It measures a certain average of the spatial-frequency spectrum at different points in the image, then uses the changes in those spatial-frequency measures to infer changes in texture compression resulting from image projection effects. Intuitively, the kind of homogeneity assumed here is that the texture is the same everywhere on a surface, in the sense that it is free of systematic trends from one location to another. There must not be systematic variations in texture element size, density, orientation, or other characteristics on the surface, because in the average spatial-frequency measure used in the next section, such variations will appear the same as changes in texture compression. This type of homogeneity is what Malik and Rosenholtz (1994a, 1994b, 1997) have formalized as the property of *texture stationarity.* In essence, it means that the texture has the same statistical properties at every surface location.

4.1.4 Summary: Projection Model and Assumptions About Texture Properties

To summarize, the model of texture-based shape estimation described in this chapter makes the following assumptions:

- Input images are formed by orthographic image projection. This is an approximation to a limited set of viewing conditions, namely when surfaces are being viewed at distances that are large compared to the variations in depth across the surface themselves. The use of orthographic projection unfortunately limits the kind of information that can be inferred from an image. In particular, only the changes in shapes across a curved surface will be recoverable; the method will be unable to determine changes in depth across *planar* surfaces, or the differences in relative depths between different surfaces.

- Texture homogeneity. The texture on a given physical surface must be homogeneous with identical statistics everywhere on the surface. This assumption is approximately valid for a number of types of natural and artificial textures, but is not satisfied by all textures. The closer to being homogeneous a texture is, the more correct will be the shape estimates produced by the model.

4.2 Using Average Peak Frequency for Estimating Surface Shape

In Section 2.3, I discussed a number of models of how a visual system could infer shape from texture. The different models use a variety of operating principles. In many cases it is unclear whether there is evidence to support them as models of texture-based shape estimation in the *human* visual system. The work by Sakai and Finkel (1994, 1995), Sakai, Yen, and Finkel (1996) is an exception; it is based on psychophysical experiments suggesting that human perception of shape from texture is correlated with a relatively simple measure of spatial-frequency changes in an image. This is an exciting finding because a method based on detecting a simple characterization of the spatial-frequency spectrum is computationally less demanding than working with the full, detailed spectrum. Sakai and Finkel have also developed a neural model based on these principles.

In this section, I describe Sakai and Finkel's theory and associated model in some detail, paying attention especially to the principles underlying the model's operation. In Section 4.3, I describe the modifications to the model for the current research.

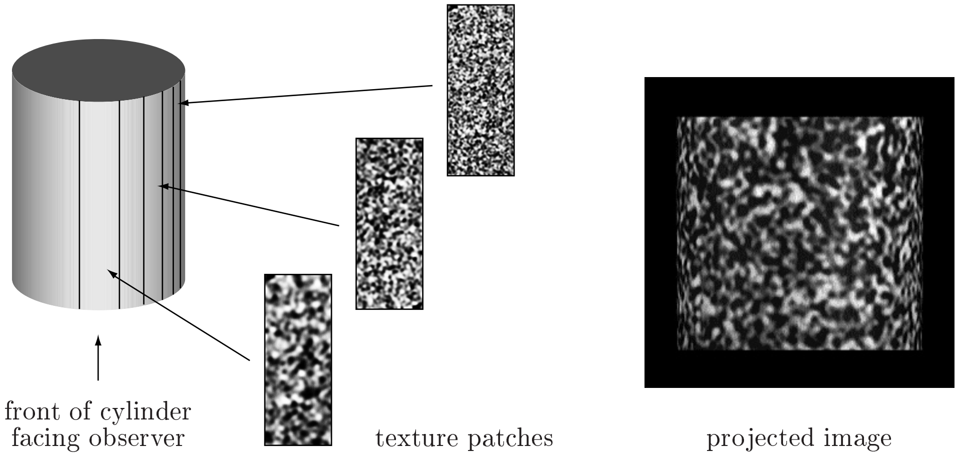


Figure 4.5: Basic principle used by Sakai and Finkel (1995) to construct stimuli with controlled spatial-frequency spectra. A shape such as a cylinder is divided into a large number of polygon facets, and a swatch of texture is mapped on each polygon face. The texture swatches are designed to have certain properties in the spatial-frequency spectrum. A cylinder with a normal three-dimensional appearance, such as the example shown at the right, can be generated as follows. First, the spectrum of the texture on the central midline facet facing the viewer is made to contain relatively low spatial frequencies, which is to say, a coarse texture. Then, the facets increasingly farther out toward the periphery of the cylinder are made to contain increasingly higher spatial frequencies in addition to the low frequencies present in the frontal facet. The result is a surface that appears to be smooth and covered uniformly in a random noise texture.

4.2.1 Experimental and Theoretical Basis of Sakai and Finkel's Model

Sakai and Finkel's (1995) psychophysical experiments consisted of showing to human observers images of surfaces whose spatial-frequency characteristics were manipulated artificially. An example of how they constructed their stimuli is shown in Figure 4.5. Their method consists of first generating swatches of textures from filtered random (white) noise and then mapping the swatches onto different facets of a polygon approximation to a surface such as a cylinder. The spatial-frequency content of each texture swatch can be controlled by filtering the spectrum in the Fourier domain using filters. By manipulating the spatial-frequency characteristics of the surfaces in the images and testing people's perceptions of the results, it is possible to examine which characteristics are important to people's perception of three dimensional shape from texture.

Summary of Main Experimental Results

Sakai and Finkel (1995) first demonstrated that the exact shape of the spatial-frequency spectrum is not the prime determinant of people's ability to perceive a given shape. For example, two cylinders covered in radically different texture patterns (and thus having different frequency spectra) can nevertheless be perceived as cylinders of the same diameter.

The authors examined which of many possible characterizations of the frequency spectra may be correlated with observer's perception of shape from texture, and found two such characterizations. For some types of stimuli, the strongest or *peak* spatial-frequency component at different spatial locations appears to be important, whereas for other types of stimuli—those that lack strong frequency peaks—it is the *mean* frequency that is better correlated with perceptual impression of shape.

The first case is illustrated in the upper half of Figure 4.6 on the following page. The figure depicts stylized spatial-frequency profiles measured at two different points on cylinders covered with different textures. The top half of the row presents horizontal slices through the spatial-frequency profiles in a neighborhood of the center of the cylinders, while the bottom half of the row presents frequency profiles in a neighborhood of the periphery of the cylinders. Suppose a cylinder is covered with a texture that has a spatial-frequency component whose magnitude is several times greater than any other frequency component. If, when going from the center of the cylinder to the edges, the peak frequency in the image changes in value as illustrated in (a) while the average frequency remains the same, then human observers report the impression of a three-dimensional shape. The image in the left column gives an example of a 3-D stimulus of this kind. On the other hand, if the strong peak frequency remains the same going from center to periphery while the average frequency changes as shown in (b), then observers perceive no three-dimensional shape—the surface appears flat.

The second case is illustrated in the lower half of Figure 4.6. Suppose a cylinder is covered instead with a texture that has only a weak spatial-frequency peak, such as a noise texture. If the (weak) spatial-frequency peak remains the same across the cylinder in the image but the average frequency changes as shown in (c), then observers report seeing a three-dimensional shape. Conversely, if the peak frequency changes from low to high frequency when going from the center to the periphery, but the mean frequency remains the same as in (d), then observers do not report seeing a three-dimensional shape.

These examples illustrate Sakai and Finkel's (1995) results suggesting that

- In the presence of a strong peak frequency, we perceive 3-D shape when the peak frequency changes and the mean frequency remains roughly the same across the image of the surface; and

- In the absence of a strong peak frequency, we see perceive 3-D shape if the mean frequency changes while the (weak) peak stays the same.

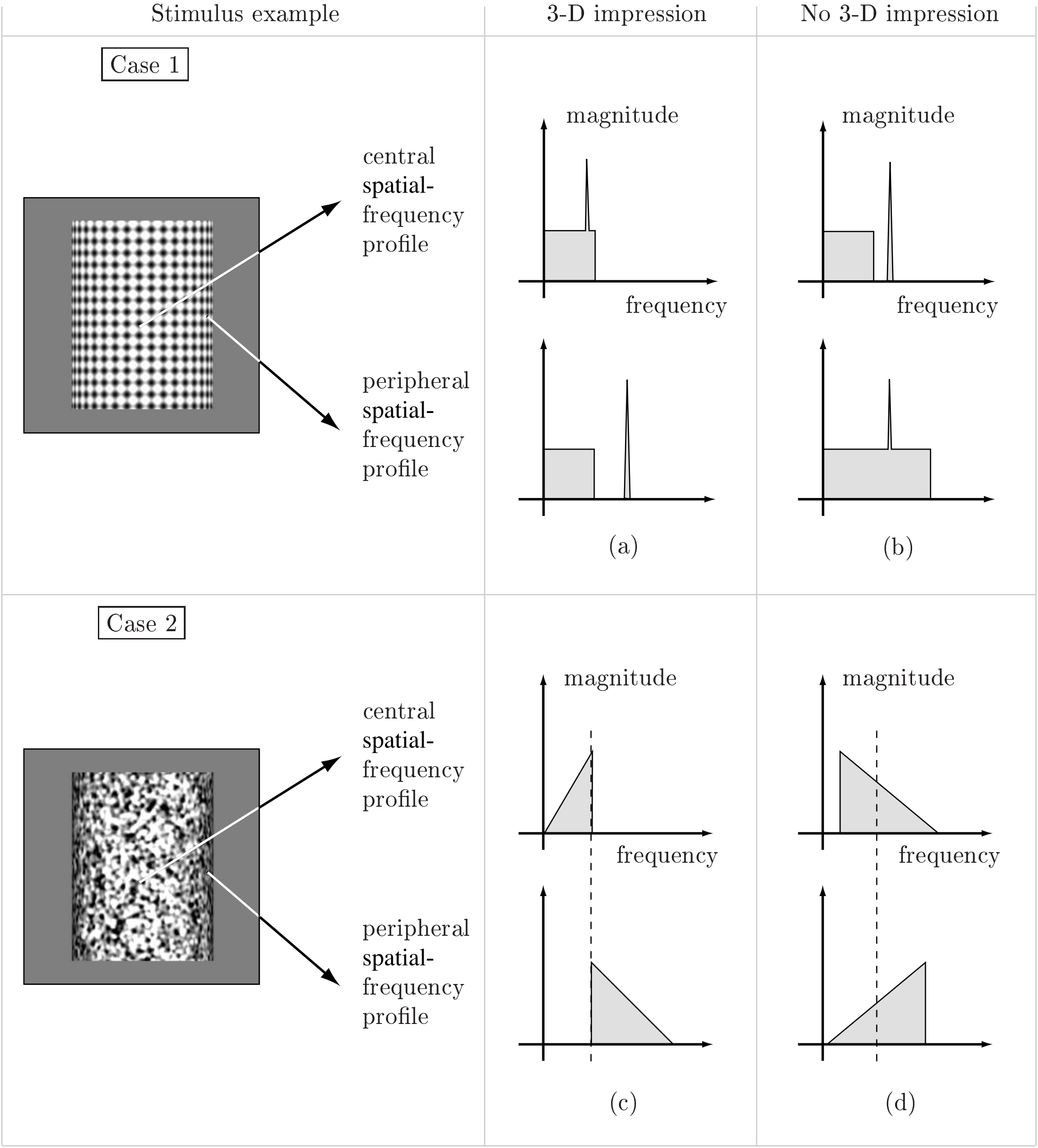


Figure 4.6: Summary of one of Sakai and Finkel's experimental results. Each graph in the second and third columns represents a stylized cross-section of the spatial-frequency spectrum measured at either the central or peripheral facets of textured cylinders. There are two cases shown: the first case in the upper half of this figure shows profiles corresponding to textures that have a dominant (peak) spatial frequency; the second case in the lower half shows profiles corresponding to textures that do not have a peak frequency. (a) If the dominant frequency at each location on the surface in the image changes when going from the central midline facet to the most peripheral facet of the cylinder, and the mean frequency remains the same, observers report seeing a three-dimensional cylinder, as in the example in the first column. (b) If the peak frequency does not change while the mean frequency changes, observers do not report an impression of three-dimensionality. (c) If the texture has only a weak peak frequency that stays the same from center to periphery while the mean frequency changes, observers again report seeing a 3-D cylinder, as in the example in the first column. (d) If the weak peak changes while the mean frequency stays the same, observers do not report an impression of three-dimensionality. Adapted from Sakai and Finkel (1995).

Summary of Theoretical Model

Sakai and Finkel (1995) propose a simple measure of the spatial-frequency spectrum that allows detecting either the peak or the mean as appropriate: the local spatial average of the peak spatial frequency, dubbed the *average peak frequency* (APF). It can be computed by determining the dominant frequency at each spatial location in an input and then averaging the result over a spatial neighborhood centered at each location. For the two cases of Figure 4.6 on the page before, the average peak frequency behaves as follows. When the spatial-frequency spectrum contains a relatively strong peak f and there is little variability about the peak, as illustrated in case one of Figure 4.6, then the average of the peak frequency at different locations closely approximates f itself. On the other hand, when the spectrum is not dominated by a single peak frequency and there is significant variability in the strongest frequency component between different locations, as in case two of Figure 4.6, the APF more closely follows the mean value of frequency.

How is APF related to surface orientation? On a single, continuous, homogeneously-textured surface under orthographic projection, the change in the spatial-frequency spectrum in an image from one point to another is directly related to texture compression. To determine the amount of texture compression, Sakai and Finkel (1995) first assume that the largest texture in the input (and therefore the lowest APF) lies in the region of zero slant; that is, the coarsest texture is assumed to be on a part of the surface that is closest to and facing the viewer. Let $\bar{F}(x, y\,; \theta)$ represent the average peak frequency at some point (x, y), and $\bar{F}_{\text{min}}(\theta) = \min\limits_{x,y \in I}[\bar{F}(x, y\,; \theta)]$ represent the lowest APF value in orientation θ across the whole input image I. (Treating the orientations separately allows anisotropic textures to be handled in a stage described below.)

If the largest texture is closest to the viewer, then under orthographic projection, the texture in the image is equal in size to the texture on the surface. This means that, based on the derivation in Section 4.1.2 on page 125, the spatial-frequency spectrum of the surface texture at locations away from the frontoparallel region is modified by a factor of $1/(\cos\sigma)$ when it is projected onto the image plane:

$$\bar{F}(x, y\,; \theta) = \frac{\bar{F}_{\text{min}}(\theta)}{\cos\sigma}, \tag{4.8}$$

where σ is the angle of slant. From this, one immediately gets the following expression for the local slant angle at (x, y) in the direction θ:

$$\sigma(x, y\,; \theta) = \cos^{-1}\left[\frac{\bar{F}_{\text{min}}(\theta)}{\bar{F}(x, y\,; \theta)}\right]. \tag{4.9}$$

In order to be able to integrate together the local values of slant into an estimate of the overall shape, it is convenient to use the local gradient of the surface instead of σ directly. Perceptually, the gradient of the surface corresponds to the slope of the surface, which is given by the tangent of the slant angle. Let $\sigma(x, y\,; \theta)$ be the estimated local surface slant σ in a given orientation at location (x, y); then the tangent of the local slant is

$$\begin{aligned}\tan[\sigma(x,y\,;\theta)] &= \tan\left\{\cos^{-1}\left[\frac{\bar{F}_{\min}(\theta)}{\bar{F}(x,y\,;\theta)}\right]\right\} \\ &= \sqrt{\frac{\bar{F}(x,y\,;\theta)^2-\bar{F}_{\min}(\theta)^2}{\bar{F}_{\min}(\theta)^2}}. \qquad (4.10)\end{aligned}$$

For simplicity, Sakai and Finkel (1995) approximate this quantity by the linear relationship

$$\tan[\sigma(x,y\,;\theta)] = \frac{\bar{F}(x,y\,;\theta)-\bar{F}_{\min}(\theta)}{\bar{F}_{\min}(\theta)}. \qquad (4.11)$$

The numerator in this expression corresponds to the change in texture size, and the division by $\bar{F}_{\min}(\theta)$ represents the ratio of how much the texture at a location is compressed with respect to the largest texture on the surface. The computation is performed in each orientation θ separately. The linear approximation satisfies the condition that the slant $\sigma(x, y\,; \theta)$ is zero at those locations where $\bar{F}(x, y\,; \theta) = \bar{F}_{\min}(\theta)$.

The expression above provides an estimate of local slant in a particular orientation. However, more processing is needed to estimate the direction of tilt and combine the local slant and tilt estimates into an interpretation of the overall surface. Under orthographic projection, the direction along which the texture is maximally compressed corresponds to the direction of the surface gradient, which is the direction of tilt (see Section 4.1.2). Sakai and Finkel (1995) use a type of lateral inhibition to identify the direction of maximal texture compression. The lateral inhibition process takes, at each location (x, y), a regional average of the normalized APF in one orientation θ as an excitatory input, and the average of the normalized APF in the perpendicular orientation $\theta + 90^\circ$ as an inhibitory input. The process passes unchanged the APF value in the orientation with the largest normalized frequency, and at the same time inhibits the values in other orientations to a degree depending on their relative frequencies. If a texture is compressed equally in multiple directions, as might happen if the texture is scaled due to perspective effects, the lateral inhibition suppresses all the APF values equally and thereby prevents the perception of surface curvature at that point.

Note that it may be more appropriate to refer to this inhibitory process as a type of "cross-orientation" inhibition rather than "lateral" inhibition. However, *cross-orientation* *inhibition* is a term commonly used for an experimental phenomenon involving the responses of cortical cells to superimposed, orthogonally-oriented gratings (Bonds, 1989; Nestares & Heeger, 1997). It is unclear whether the process behind this experimental phenomenon is the same as Sakai and Finkel's (1995) "lateral inhibition"; therefore, I use Sakai and Finkel's term, despite that it may not be as intuitively clear in this role as the term "cross-orientation inhibition" might be.

The final step in the model, combining the local slant and tilt estimates into an interpretation of the overall surface, is accomplished by integrating the local slants to produce a depth map. This is a common approach in computer vision (where it is sometimes known as *integrating a needle diagram*; c.f. Horn, 1986); it simply involves adding up, step-wise, the local changes in surface gradients along an extended path across a region. Each change in gradient between two locations becomes a change in relative depth across the surface. The integration process in Sakai and Finkel's (1995) model adds up the largest average peak frequency at each location from among all the orientations, discarding the other values. The integration is performed along a path from a given location towards the point closest to the viewer (that is, the region of lowest APF). The direction of the path is found by searching locally for the direction of steepest change in the local slant. This process is discussed in more detail in the next section.

4.2.2 Summary of Sakai and Finkel's Computational Model

Sakai and Finkel (1994, 1995) have proposed a biologically-motivated neural model that embodies the empirical and theoretical results summarized in the previous section. Their software simulation of the model can estimate surface shape from patterns of texture using the average peak frequency measure. Their model is divided into four main stages:

1. Computation of the responses of simulated complex cells at each point in an input image as a means of estimating the spatial-frequency content at that point. (Since complex cells respond to a limited range of frequencies and orientations, each cell's response can be taken as an estimate of whether the stimulus falling within the cell's receptive field contains those frequencies and orientations.)

2. Computation of the average peak spatial frequency at each point in the input.

3. Normalization of the average peak frequency followed by lateral inhibition between orthogonal orientations, to provide a local estimate of slant and tilt.

4. Integration to combine the local estimates of slant and tilt at each location into a depth map representing the overall surface shape.

Figure 4.7 on the next page provides a diagram of the model's overall architecture. The first stage performs spatial filtering in a manner similar to complex cells in cortical area V1. In Sakai and Finkel's implementation, input images are first convolved with spatial filters having receptive field profiles resembling those of cortical simple cells. The input images used in Sakai and Finkel's (1995) simulations are 180 × 180 pixels in size. The spatial filters cover four orientations in 45° intervals and nine preferred spatial frequencies with wavelengths ranging from thirty-two to two pixels in multiples of $\sqrt{2}$ (which corresponds to a range of about 1–20 c/deg). Each filter's output is passed through a rectification stage, then the outputs of on- and off-center filters are combined to achieve contrast invariance. The maximum filter response over a small local neighborhood is then computed, and this value is taken to approximate the output $O_1(x, y\,; f, \theta)$ of a complex cell tuned to spatial frequency f and orientation θ.

The second stage in the model estimates the unnormalized APF (average peak spatial frequency) at each point in the input. It involves first finding, at each location, the strongest-responding complex cell from among those responding to all different spatial frequencies. This is essentially a winner-take-all computation that reduces the nine complex cell responses to one value per orientation. At the same time, the location is tagged with the spatial frequency to which the strongest-responding cell is tuned. These spatial-frequency values are then averaged separately per orientation in a spatial neighborhood centered at each location. Together these computations are expressed as

$$O_2(x, y\,; \theta) = \underset{x', y' \in I_a(x,y)}{\text{average}} [\text{pfreq}(x, y\,; \theta)]. \tag{4.12}$$

The operation *pfreq* finds the spatial frequency of the strongest-responding complex cell at a given location (x, y) and orientation θ:

$$\text{pfreq}(x, y\,; \theta) = \{\, f : O_1(x, y\,; f, \theta) = \max_{f} O_1(x, y\,; f, \theta) \,\}. \tag{4.13}$$

The neighborhood $I_a(x, y)$ in the formula for $O_2(x, y\,; \theta)$ consists of all locations within a radius r_a of (x, y). The *average* operation is Gaussian-weighted averaging, using a standard deviation equal to r_a. In their simulation, Sakai and Finkel (1995) used a radius $r_a = 25$ cells. It is important to note that the results from this second stage are frequency values, and not, for example, strengths of neural responses.

At this step, Sakai and Finkel (1995) also subsample the results by only computing $O_2(x, y\,; \theta)$ at every other pixel of the input. This reduces the size of the output to a 90 × 90 array of values, and thereby reduces computational requirements on later stages.

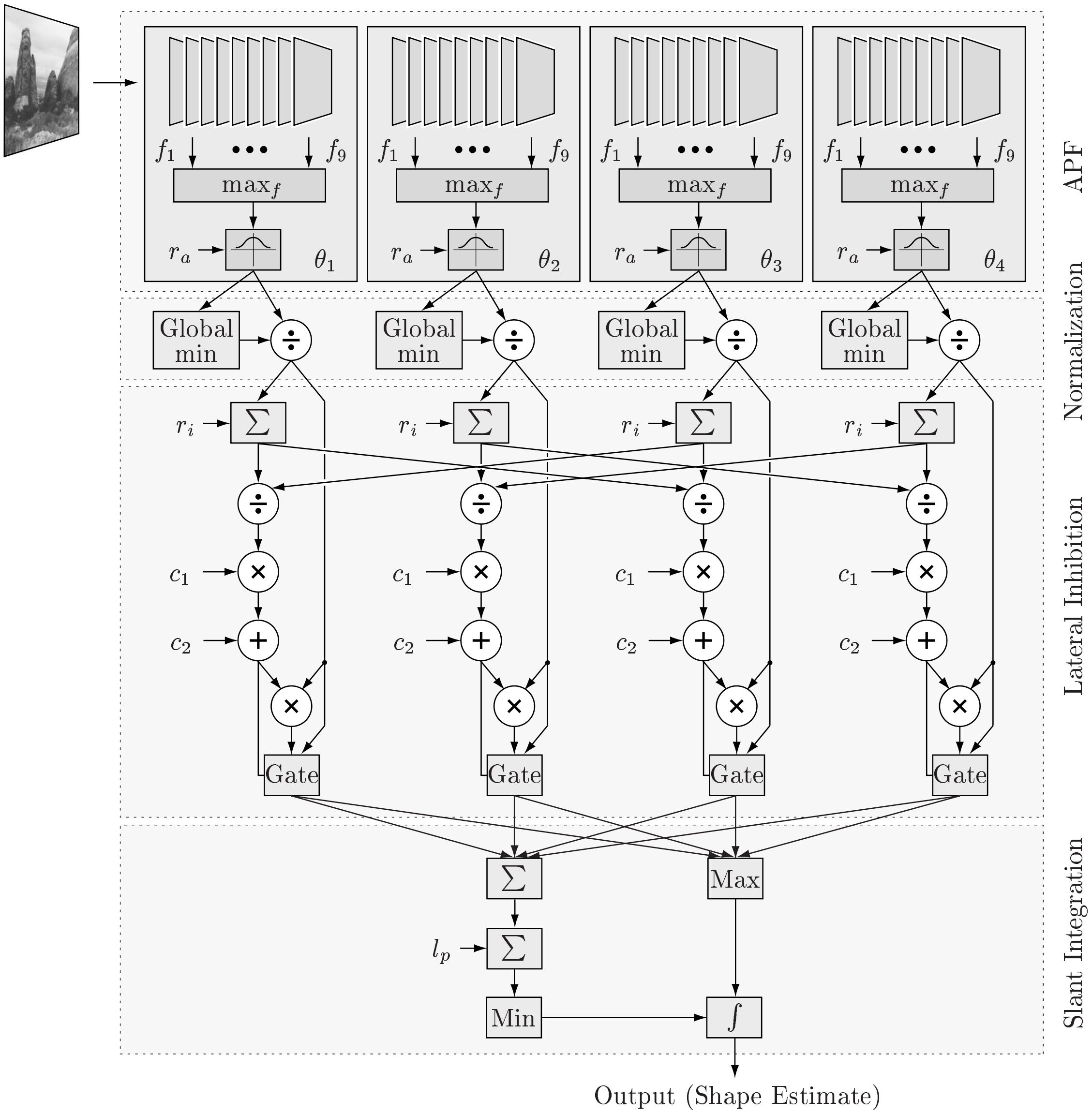


Figure 4.7: The general architecture of Sakai and Finkel's (1994, 1995) model for deriving shape from visual texture information. The first stage uses spatial filters that simulate complex cells. There are filters for each of nine spatial frequencies (f_1–f_9) and four orientations (θ_1–θ_4). The model finds the average peak spatial frequency (APF) at each local in the visual input by finding the strongest-responding complex cell at each point across all nine frequencies, and then averaging the frequency values in a circular neighborhood. The APF values are then normalized in each orientation separately; this produces an estimate of local texture compression along each orientation. A lateral inhibition stage between orientations then gives an estimate of the direction in which texture compression is greatest. The combination of these two values (amount of compression and direction of greatest compression) gives surface slant and tilt at each point in the image. These values are integrated to produce an estimate of the three-dimensional surface shape.

The third stage begins by computing the normalization of the average peak frequency,

$$\widetilde{O}_2(x, y\,; \theta) = \frac{O_2(x, y\,; \theta)}{\displaystyle\min_{x', y' \in I(x,y)} O_2(x', y'\,; \theta)} - 1, \tag{4.14}$$

where the denominator is the minimum frequency found by searching the entire input $I(x, y)$. This is simply a restatement of Equation 4.11 on page 135. The expression above normalizes the average peak frequency at a given point and given orientation, thereby providing an estimate of the amount of texture compression at that location. This is taken to be an estimate of the relative surface slant in that particular orientation at location (x, y).

To determine the direction in which compression is maximal, and therefore the direction of tilt, Sakai and Finkel (1994, 1995) use a lateral inhibition process. The goal of this process is to promote, at each location, the orientation with the highest normalized frequency and to suppress the activity of neurons responding to orthogonal orientations. The result is still a map for every orientation, but now at each location the orientation containing the largest normalized frequency (which is to say, the greatest degree of texture compression) corresponds to the estimated direction of tilt at that point. The lateral inhibition process is performed according to the formula

$$O_3 = \begin{cases} 0 & B(x, y\,; \theta) < 0, \\ \widetilde{O}_2(x, y\,; \theta)\, B(x, y\,; \theta) & 0 \leq B(x, y\,; \theta) \leq 1, \\ \widetilde{O}_2(x, y\,; \theta) & 1 < B(x, y\,; \theta). \end{cases} \tag{4.15}$$

The term $B(x, y\,; \theta)$ is a scaling coefficient given by

$$B(x, y\,; \theta) = c_1 \left[1 - \frac{\displaystyle\sum_{x', y' \in I_i(x,y)} \widetilde{O}_2(x', y'\,; \hat{\theta})}{\displaystyle\sum_{x', y' \in I_i(x,y)} \widetilde{O}_2(x', y'\,; \theta)} \right] + c_2. \tag{4.16}$$

Here, $\hat{\theta}$ is the orientation opposite to θ; i.e., $\hat{\theta} = \theta + 90^\circ$. $I_i(x, y)$ is a neighborhood of radius r_i surrounding location (x, y), and c_1 and c_2 are constants that affect the degree of lateral inhibition. In Sakai and Finkel's (1995) implementation, $r_i = 5$ cells, and c_1 and c_2 have the values 5 and -0.5, respectively.

The result of the computations above is a distributed representation of slant and tilt. There are as many $O_3(x, y\,; \theta)$ maps as there are orientations, and each such map has values that have been rescaled by lateral inhibition. This distributed representation is coalesced into a representation of relative depth across the surface by the processes of the fourth stage of the model. In the fourth stage, the local slant at each location is represented by the largest normalized frequency across all orientations. This is performed by a winner-take-all mechanism. The resulting values are then integrated step-wise, from one point to the next, leading to an estimate of relative depth across the surface.

The fourth stage consists of two main parts. One part involves finding the direction of integration for every location in the image. This is chosen to be the direction heading towards the region of lowest normalized APF out of the eight possible directions away from a given point (x, y). (The eight possibilities come from the fact that each point on a rectangular sampling grid has eight neighbors.) The direction is represented in terms of the neighboring location that lies on a path leading to the minimum sum of APF values, according to

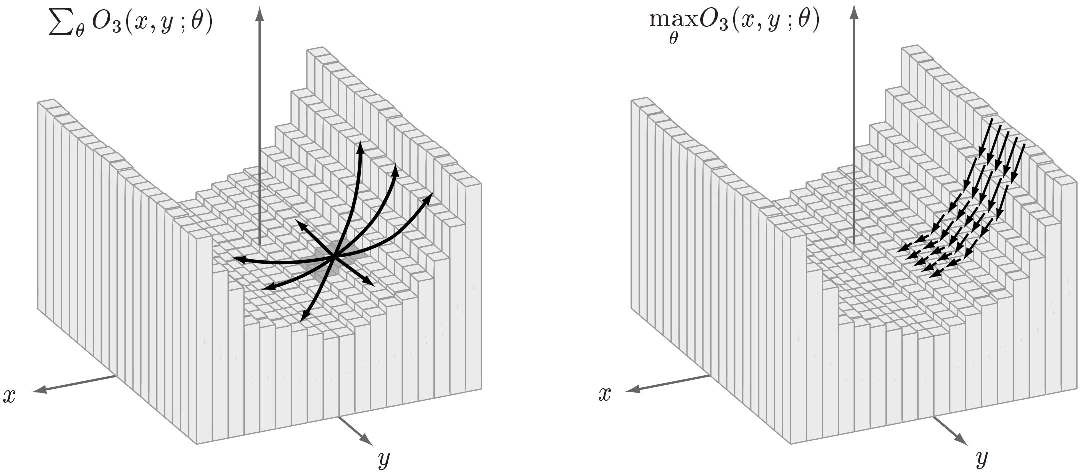
$$D(x,y) = \left\{ (r,s) : \sum_{r',s' \in I_p(r,s)} \sum_{\theta} O_3(x+r', y+s' ; \theta) \right. \\ \left. = \min_{r,s \in I_n(x,y)} \left[\sum_{r',s' \in I_p(r,s)} \sum_{\theta} O_3(x+r', y+s' ; \theta) \right] \right\}. \quad (4.17)$$

In this formula, $I_n(x, y)$ is the set of eight immediate neighbors of (x, y), and $I_p(r, s)$ is the set of l_p consecutive units in the direction of (r, s) from (x, y). In Sakai and Finkel's (1995) implementation, $l_p = 20$.

The equation above simply finds, for each location (x, y), the neighboring cell (r, s) lying on the path of l_p cells away from (x, y) that constitutes the lowest sum. This is illustrated conceptually in Figure 4.8(a) on the next page. At each location, the model examines l_p consecutive values of $\sum_\theta O_3(x, y ; \theta)$ in each of eight possible directions. To take an example, suppose that at some location (x_p, y_q), the sum of values along the northeasterly diagonal direction have the lowest sum of values. Then $D(x_p, y_q)$ would indicate that the proper direction is the one represented by cell (x_{p+1}, y_{q-1}). This process of choosing directions can be performed at every location in parallel.

The second part of the fourth stage is the integration of local slant. First, the maximum values across all orientations in $O_3(x, y ; \theta)$ are found by computing $\max_\theta[O_3(x, y ; \theta)]$. Second, the locations having the *lowest* resulting values are assigned a relative depth of zero. Finally, the relative depths of all other locations are computed according to the formula

$$O_4(x,y) = O_4(r,s) + \max_{\theta} O_3(x, y ; \theta). \quad (4.18)$$


$\sum_\theta O_3(x,y\,;\theta)$
x
y
$\max\limits_\theta O_3(x,y\,;\theta)$
x
y

(a) (b)

Figure 4.8: Illustration of the process of choosing directions during slant integration. The input for this example is the image of a small, regularly-textured cylinder. (a) The set of values of $\sum_\theta O_3(x,y\,;\theta)$ can be plotted as vertical bars at each location of the input, with the height of each bar representing magnitude. This produces a landscape of values. For a textured cylinder, the landscape might have the appearance shown here. The process of choosing directions for integration involves, at each location, taking the sum of values of $\sum_\theta O_3(x,y\,;\theta)$ along paths of length l_p. These sums are taken in each of eight directions, the directions set by the immediate neighbors of a given (x,y) location. (The neighbors for this case are shown shaded in dark gray.) The direction chosen is the one that gives the lowest sum across the l_p units in the path. The result is represented in terms of the neighbor lying along the path of the chosen direction. (b) The set of values of $\max_\theta[O_3(x,y\,;\theta)]$ can also be plotted as vertical bars at each location of the input, yielding another landscape. The slant integration process involves adding the value at each location with the value of the neighbor chosen during the search for the directions of integration. Ideally, the direction at each location will point towards the area of lowest frequencies, as illustrated here for a subset of the locations.

The point (r,s) is the neighbor of (x,y) chosen according to the slant integration map $D(x,y)$. This integration formula must be iterated at every location in the input. At each iteration, every location's depth value is updated based on one of its neighbors. Those locations that have the lowest normalized APF values are considered to be at zero depth and are not updated; they serve as seed regions. In the first iteration, only the immediate neighbors of these regions will be assigned the correct relative depths. At the next iteration, the neighbors twice removed from the low frequency regions will be assigned the correct depth, and so on for other iterations. Ideally, the neighbor updates will proceed from the region of lowest APF outward towards the region of highest normalized APF, in a manner illustrated in Figure 4.8(b). Sakai and Finkel (1995) report that for the 90×90 arrays used in their experiments, 30 to 70 iterations are typically required for the process to settle to steady-state values.

The computation in Equation 4.17 only needs to be performed once for a given input image because it relies only on $O_3(x, y\,; \theta)$ and not on $O_4(x, y)$. Since the map of values $D(x, y)$ represents the direction along which the piece-wise slant integration of Equation 4.18 is performed, I will call it the *integration direction map.* The computation of the $\max_\theta[O_3(x, y\,; \theta)]$ term in Equation 4.18 also needs to be performed only once for a given input image, again because it relies only the outputs of the previous stage and not the current values.

It is worth noting that the value of the integration direction map at each location is the opposite of the local direction of tilt. Thus, if the results of Equation 4.17 are plotted as arrows at each location, and then the direction of each arrow is reversed, the result is a representation of the tilt direction at each point in the input. I will call this the *tilt direction map.*

The use of integration in the last stage is not strictly part of the theory of estimating surface shape from average peak spatial frequencies. Sakai and Finkel (1995) note that the nature of surface representation in the brain is unknown, and the depth-based shape reconstruction is simply a convenient way of extracting and summarizing surface shape information out of a distributed slant/tilt representation. The brain probably does not perform explicit reconstruction of surfaces of the form described here, but some process of connecting the local estimates of slant and tilt must take place.

4.2.3 Discussion

In summary, the model of texture-based shape estimation described in this section is based on the observation that human observers' perception of surface shape is correlated with a relatively simple description of texture. That description is the local spatial average of the peak spatial frequency. By tracking the changes in average peak frequency across a surface, a visual processing system can estimate its overall form.

The neural model developed by Sakai and Finkel (1994, 1995) uses a four-stage processing architecture. Beginning with the responses of complex-cell-like filters, the model computes the average peak frequency at each point in the input, then normalizes the frequency values and uses lateral inhibition to produce local estimates of slant and tilt, and finally uses an integration stage to tie together the local estimates into an estimate of overall surface shape.

Relationships Between the Model and Processes in the Brain

The individual computational elements used in Sakai and Finkel's (1994, 1995) model are all simple, consisting of such things as winner-take-all operations, regional averaging, and propagation of activity among neighboring computational units. It is fair to expect that all could be implemented in a neural network. A number of the computations in the model could also be performed in parallel, including much of the integration stage. For example, the integration direction map and the local slant (in the form of $\max_\theta[O_3(x, y\,; \theta)]$) could be computed in parallel at each location (and indeed this is how Sakai and Finkel implemented their simulation).

The substrate for the neighboring-neuron interactions that are required for these computations are known to exist in the brain. Long-range connections between neurons of similar and orthogonal orientation preferences have been found in the visual cortex (Gilbert & Wiesel, 1989, 1990; Gilbert, 1993), as have connections between neurons tuned to different spatial frequencies (Bonds, 1989; De Valois & De Valois, 1990). Psychophysical experiments have also demonstrated frequency- and orientation-dependent interactions between stimuli in a visual field (Polat & Sagi, 1993, 1994a, 1994b; Sagi & Hochstein, 1985; Sagi, 1990, 1995).

Perhaps the most awkward computational element in the theory is the use of an iterative approach in Equation 4.18 on page 140. Data on object recognition and other types of visual processing in the brain show that these processes are extremely rapid—measurements of brain responses show stimulus-evoked recognition activity at about 100–150 msec after stimulus presentation (Victor & Conte, 1991). It takes approximately 50 msec for signals evoked by a visual image to reach area V1 in the cortex (measured at layer 4B of V1; Nowak, Munk, Girard, & Bullier, 1995), and a neural spike takes a minimum of about of 1–5 msec to travel from one neuron to the next (Nicholls, Martin, & Wallace, 1992; Shepherd, 1994). This implies that from the moment neural signals reach V1 to the moment of recognition, the brain has time to perform a maximum of 100 iterations of some process, and more realistically, probably has time for no more than 20 iterations. This implies that an iterative propagation algorithm for the integration step is unrealistic.

However, I hypothesize that the visual system may not need to build up a surface representation through integration. The reason is as follows. The information about local slant and tilt is already available in a distributed form prior to the integration step in the model above: the tilt direction map and the map of $\max_\theta[O_3(x, y\,; \theta)]$ values constitute a distributed representation of local tilt and slant. Such a distributed representation may be enough for a system to *recognize surface configurations* without having to build an explicit surface representation. After all, the nature of brain representations *is* distributed activity—the very fact that some neurons are active, and other are not, is its own representation (Churchland & Sejnowski, 1992). It therefore seems unlikely that the visual system would expend time and effort on reconstructing a surface shape, when it could instead work by recognizing the collective pattern of activity induced by the shape in some network of neurons. A superb example of the power of this approach is the recent work by Rao and Ballard (1995). They used the set of outputs of simple-cell-like spatial filters as the direct input into a sparse distributed memory system (c.f. Kanerva, 1988), without constructing an intermediate representation. In effect, the memory in their model is iconic. The resulting system can recognize entire objects by the signature of activity they produce in the memory network.

Thus, although an explicit representation such as that provided by the integration of local slant is useful for us to observe the simulation results, the brain does necessarily have to use this approach. Nevertheless, for the purposes of this research, I will use the same integration process used by Sakai and Finkel (1995) in order to represent the outputs of the system.

Strengths and Weaknesses of the Model

The texture-based shape estimation model described above has several strengths. One is that it offers a hypothesis about how the human visual system may exploit patterns of texture to judge surface shape. Although many other models of shape estimation have been inspired by biological vision, the majority are computer vision efforts. Few could be considered biologically reasonable or have the kind of experimental foundations that Sakai and Finkel's (1994, 1995) model has.

Another strength of the model is its relative simplicity. This makes the model easier to understand and analyze, and simpler to integrate with other texture-based processing components. Of course, when it comes to brain mechanisms, simplicity of a model does not necessarily impart greater plausibility. But simpler operations do lend themselves more easily to implementation in neural mechanisms of the sort known today.

For the present research purposes of developing a shape estimation model that can work with images of natural scenes, the most glaring weakness of the model is its simplifying assumption of orthographic image projection. As discussed in Section 4.1, under orthographic projection texture distortions are assumed to be related only to surface curvature and shape. The orthographic projection assumption is reasonable in a number of viewing situations, but natural scenes often contain perspective cues to depth. The presence of perspective effects will misled the model. Although I have not made use of it in the present research, Sakai and Finkel (1994, 1997) have developed a modified version of their shape estimation model that can exploit cues from perspective projection. A future enhancement to the work described in this dissertation will be to use Sakai and Finkel's extensions and add the ability to handle perspectively projected views of surfaces.

A second important limitation of the model is its assumption that the region of lowest spatial frequencies in the image correspond to a surface patch that is oriented parallel to and facing the viewer. This assumption is necessary because there is no way to estimate absolute surface orientation within the framework. By assuming that there is some part of the surface oriented frontoparallel, the model can determine absolute slant without knowledge about the surface texture, an idea used in other models as well (e.g., Super & Bovik, 1995). This assumption again limits the kinds of inputs that can be handled. Overcoming this limitation would require using perspective cues or additional information. In a complete visual system, other processes can be expected to supply the missing information.

4.3 Reimplementation of Sakai and Finkel's Approach in the Present Framework

In this section, I describe a version of Sakai and Finkel's (1995) average peak frequency model that is based on the simulated complex cells described in the previous chapter. The version described in this chapter, like Sakai and Finkel's model, is designed to handle input images containing only one surface. In Chapter 6, I extend the model to handle inputs containing multiple regions of texture.

Much of this implementation follows closely that of Sakai and Finkel (1995). Therefore, in the following paragraphs, I mainly contrast my implementation with theirs in order to highlight the differences. Except for the differences in the complex cell simulation, the changes described in this section represent the result of following a strategy of systematic exploitation of failure: I began with a close reimplementation of Sakai and Finkel's model, and gradually added enhancements intended to overcome limitations and problems.

4.3.1 Stage One: Computing Complex Cell Responses

The first stage of this model uses the formulation of complex cells described in Chapter 3. This stage computes the responses of the set of complex cells $C(x, y\,; f, \theta)$ to an input image; the resulting maps of responses constitute the set of cell responses $O_1(x, y\,; f, \theta)$ used by the average peak frequency model. The primary differences between the present implementation of this first stage and the implementation described by Sakai and Finkel are the following:

- The spatial filters used by Sakai and Finkel (1995) have fixed spatial-frequency and orientation bandwidths for all frequencies. As explained in Chapter 3, the present simulation of complex cells uses spatial-frequency and orientation bandwidths that vary with the peak frequency of the cells. This is in closer agreement with biological data (e.g., De Valois et al., 1982; De Valois & De Valois, 1990; Foster et al., 1985; Kulikowski & Bishop, 1981). Using a detailed model, such as the one here, is more difficult; on the other hand, it reduces the risk of making a simplifying assumption that affects, in some unforeseen way, the process of estimating shape from the complex cell responses.

- The complex cells simulated in this research include a slightly larger sample of peak spatial frequencies (ten versus nine) and twice as many orientations (eight versus four). The difference in the number of spatial-frequency bands is insignificant as far as the shape estimation theory is concerned; the particular choice of frequency bands used here was motivated by the desire to achieve an adequate range and covering of spatial frequencies, based on the tuning properties of the cells and the Nyquist sampling limits on the input images. The greater number of orientations was driven by the need to cover the spatial-frequency spectrum adequately in spite of the narrow orientation bandwidths of the cells tuned to the highest spatial frequencies. Using a greater number of orientations does have a functional benefit: it provides finer resolution in orientation and leads to somewhat smoother surface shape estimates. The disadvantage of using more frequency bands and more orientations is that simulating them requires a greater amount of computation.

4.3.2 Stage Two: Computing Average Peak Spatial Frequency

The second stage of the model computes the unnormalized APF at each location using Equations 4.12 and 4.13 on page 137:

$$O_2(x, y\,; \theta) = \operatorname*{average}_{x', y' \in I_a(x,y)} [\mathrm{pfreq}(x, y\,; \theta)].$$

$$\mathrm{pfreq}(x, y\,; \theta) = \{\, f : O_1(x, y\,; f, \theta) = \max_{\theta} O_1(x, y\,; f, \theta)\,\}.$$

The result of computing $O_2(x, y\,; \theta)$ is a map of spatial frequency values. For the *average* operation, Sakai and Finkel (1995) used Gaussian-weighted averaging. However, there is an unresolved question of how zeros in the cell responses $O_1(x, y\,; f, \theta)$ should be handled.

Complex cells will generally not respond over areas that have no textural or other contrast variations. This means that near borders between textured and untextured regions, and over smooth areas of a surface, the averaging operation will encompass locations where some cell responses are zero. These zeros can drag down the average, resulting in an artificially low value for $O_2(x, y\,; \theta)$ in these regions. This can be illustrated by the difference between the averages of, say, a set of values such as

$$\begin{array}{rrr} 10 & 7 & 7 \\ 9 & 8 & 6 \\ 7 & 9 & 5 \end{array} \qquad \text{and} \qquad \begin{array}{rrr} 10 & 7 & 0 \\ 9 & 8 & 6 \\ 7 & 9 & 0 \end{array}$$

The average of the second set will be lower than the average of the first. In the context of cell responses, zeros in $O_1(x, y\,; f, \theta)$ have a special meaning: they represent complex cells that did not reach a threshold of response. The problem is that their values are not useful during the computation of a local average, and unless they are treated specially, they adversely affect the APF computation.

Sakai reports that in their implementation, they did not perform averaging at those locations in the $O_1(x, y\,; f, \theta)$ response maps having zero values (Sakai, personal communication). However, in practice I have found it critical not only to skip the averaging computation at locations having zero response, but also to discount the contributions of zeros in the average calculation itself. That is, instead of computing the average by multiplying all the responses in the neighborhood of some location (x, y) by a weighting function, and then adding the results, the calculation should use only those responses in the neighborhood that are nonzero. A formula for this kind of averaging is

$$\underset{x',y' \in I_c(x,y)}{\text{average}} \; F(x, y) = \frac{\sum\limits_{x',y' \in I_c(x,y)} [G_\sigma(x', y')\, F(x', y')]}{\sum\limits_{\substack{x',y' \in I_c(x,y) \\ F(x',y') \neq 0}} G_\sigma(x', y')} \tag{4.19}$$

where $F(x, y)$ is the function being averaged, $I_c(x, y)$ is a circular neighborhood of radius r_c centered on (x, y), and $G_\sigma(x, y)$ is a rotationally-symmetric, unit-volume Gaussian function with standard deviation σ. This is the averaging procedure I used in implementing Equation 4.12 on page 137 for computing the unnormalized average peak frequencies, as well as for other instances of averaging described below. This procedure tends to produce less distortion of APF values at border regions than an averaging procedure that does not discount zero responses.

The circular window $I_c(x, y)$ is in this case equivalent to $I_a(x, y)$ in Equation 4.12. Choosing the size of the window involves a tradeoff between the smoothness of the eventual surface estimates, and the amount of blurring of boundaries between regions. A larger radius causes more blurring, but on the other hand, the cell responses must be smoothed with a fairly large window in order to allow the later stages of the model to function well. I chose a radius of $r_c = r_a = 40$ pixels for all the simulation runs presented in this chapter; this gives a diameter of roughly 1.25° of visual angle in my simulation, under the assumption that 64 pixels equal one degree (see Section 3.3). I set the standard deviation of the Gaussian function $G_\sigma(x, y)$ to $r_c/3$. I settled on these values after considerable experimentation with different settings in the full model.

It is worth noting that Sakai and Finkel (1995) used a proportionally larger radius of averaging compared to the width of their input images. In their simulations, they used a radius of 25 pixels, which is approximately one-seventh of the 180-pixel width of their inputs. In contrast, the value of 40 pixels here is approximately one-thirteenth of the 512-pixel wide inputs—proportionally a much smaller amount. A smaller radius produces less blurring of borders between texture regions, which is an important consideration in the present research because the model will be combined with a segmentation component in later chapters. The downside to using a smaller averaging radius r_a is that estimates of surface shapes tend to be less smooth than when using a larger radius.

An additional change in my implementation over that of Sakai and Finkel (1995) is in the degree of subsampling of the $O_2(x, y\,; \theta)$ values. Sakai and Finkel subsampled the results by two in each direction, reducing the width of the square output map by one-half, to 90×90, but they started with much smaller response maps. I chose to sample every fourth value, producing output maps of size 128×128. This was done purely for practical reasons; the theoretical results of this research results apply equally well to using full-sized 512×512 maps. The computational costs of the subsequent stages (segmentation and estimation of surface orientation) are high enough that one can gain large speed-ups in execution time by shrinking the sizes of the APF maps at this stage.

A natural concern in subsampling in this manner is that the results will containing aliasing artifacts. The smoothing that results from the Gaussian-weighted averaging procedure, assuming a fairly large value for r_a, is enough to prevent this problem.

4.3.3 Stage Three: Estimating Local Slant and Tilt

The third stage in the model computes the normalized average peak frequencies $\widetilde{O}_2(x, y\,; \theta)$, and uses lateral inhibition to produce a distributed estimate of surface slant and tilt at each location in the form of $O_3(x, y\,; \theta)$. (See Equation 4.15 on page 139.) For this stage, I followed closely the implementation described by Sakai and Finkel (1995). The only differences are in the choices of averaging neighborhood radius r_i and the factors c_1 and

c_2. I chose a radius of $r_i = 22$ pixels; this value produced smoother estimates of the tilt direction overall across the different images used in testing. For c_1 and c_2, I used values of 3.0 and -0.25, respectively.

4.3.4 Stage Four: Integrating Surface Slant and Tilt to Estimate the Shape of a Surface

The last stage of the model involves computing an estimate of the shape of the textured surface that produced the landscape of normalized averaged peak frequencies. This is done by computing an integration direction map according to Equation 4.17 on page 140, and then, using Equation 4.18, integrating local depth inferred from the changes in average peak frequency values.

My implementation of this stage of the model includes a number of modifications to the implementation described by Sakai and Finkel (1995). Most of these modifications represent an effort to improve the shape estimation results; some (in particular the method for choosing among equally good integration directions) were developed in anticipation of extending this system to handle inputs with multiple regions.

Averaging the APF Values

The first modification is the addition of averaging into the computations of both the integration direction map and slant integration. Doing so improves the smoothness of the final surface estimates.

It is convenient to begin by defining two quantities as a shorthand:

$$\bar{O}_{3s}(x,y) = \underset{x',y' \in I_s(x,y)}{\text{average}} \sum_{\theta} O_3(x',y';\theta) \tag{4.20}$$

$$\bar{O}_{3m}(x,y) = \underset{x',y' \in I_m(x,y)}{\text{average}} [\max_{\theta} O_3(x',y';\theta)] \tag{4.21}$$

These simply define the spatial averages of the sum of $O_3(x,y;\theta)$ values and of the maxima of $O_3(x,y;\theta)$ values across orientations. The window functions $I_s(x,y)$ and $I_m(x,y)$ are the neighborhoods over which the averaging is performed in the two cases; both have radius r_s. For the simulations presented in this chapter, $r_s = 22$. The averages are computed using the Gaussian-weighted, zero-discounting averaging process described by Equation 4.19 on page 147.

For additional notational convenience, it is useful to give a name to the inner summation in the formula for computing the integration direction map (Equation 4.17):

$$\mathrm{ssum}(x, y, r, s) = \sum_{r', s' \in I_p(r,s)} \bar{O}_{3s}(x + r', y + s'). \tag{4.22}$$

$I_p(r, s)$ here is the same as in Equation 4.17; it is the set of l_p consecutive units in the direction of (r, s) from (x, y).

The modification to the step computing the integration direction map changes Equation 4.17 to the following:

$$D(x, y) = \{\, (r, s) \, : \, \mathrm{ssum}(x, y, r, s) = \min_{r,s \in I_n(x,y)} [\mathrm{ssum}(x, y, r, s)] \,\}. \tag{4.23}$$

$I_n(x, y)$ is the set of eight immediate neighbors of (x, y). In this chapter, $l_p = 30$ cells. The modification to the slant integration step changes Equation 4.18 on page 140 to

$$O_4(x, y) = O_4(r, s) + \bar{O}_{3m}(x, y). \tag{4.24}$$

Here, (r, s) are the coordinates of the immediate neighbor found using Equation 4.23.

Filling In of Zeros

Sometimes a surface may have a smooth patch that produces very little response in the set of complex cells. Also, the lateral inhibition stage can sometimes produce small patches of zero responses. Either of these situations can result in zeros in $\bar{O}_{3m}(x, y)$ and $\bar{O}_{3s}(x, y)$ within the region of a surface, and these values are awkward to handle in the integration step. To help deal with these zero values within surface regions, I propose that a filling-in process can average over the zeros. Neural filling-in processes are well-known in vision (Gilbert, 1993, 1994; Gilbert & Wiesel, 1992), so they are a plausible addition to this model.

I implemented the filling-in process separately for $\bar{O}_{3m}(x, y)$ and $\bar{O}_{3s}(x, y)$. The procedure is simple:

- At every (x, y) location having a zero value in $\bar{O}_{3m}(x, y)$ within the region corresponding to a surface, perform Gaussian-weighted averaging (Equation 4.19 on page 147) using an averaging radius of r_s.

- At every (x, y) location having a zero value in $\bar{O}_{3s}(x, y)$ within the region corresponding to a surface, perform Gaussian-weighted averaging using an averaging radius of r_s.

In a neural network architecture, these steps could be implemented in parallel by circuits located at each (x, y) point and sensitive to zeros produced in the course of computing $\bar{O}_{3s}(x, y)$ and $\bar{O}_{3m}(x, y)$.

Thresholding Applied to Average Peak Frequencies

The third modification concerns the slant integration step. One of the primary assumptions in Sakai and Finkel's (1995) method is that the region having the lowest normalized APF corresponds to that part of a surface that is closest to the viewer. This region also acts as an anchor for the integration step: integration is not performed for locations within the region of lowest APF, and those locations are assigned a depth of zero. Unfortunately, most images end up with only one or a few points having a value equal to the lowest normalized average peak frequency across the whole textured surface. Given so few, possibly disjoint, points of low frequency, the slant integration process often produces poor results.

The quality of the slant integration results can be improved substantially by expanding the number of points that are considered to be part of the "region of lowest normalized average peak frequencies". This allows a larger region to act as anchor for the integration process, making it more likely that the process will find smooth paths that run from the anchor region to the outer edges of the surface.

Increasing the size of the region of lowest normalized APF be achieved by growing the region starting from the points that have the lowest frequencies in $\bar{O}_{3s}(x, y)$. A simple way of growing the region is to introduce a thresholding step into the equation for slant integration, modifying Equation 4.24 on the page before to be

$$O_4(x, y) = O_4(r, s) + T_s[\bar{O}_{3s}(x, y)] \tag{4.25}$$

where the thresholding function $T_s(x)$ is given by

$$T_s(x) = \begin{cases} x & \text{if } x > c_l \min\limits_{x',y' \in I(x,y)} [\bar{O}_{3s}(x', y')], \\ 0 & \text{otherwise.} \end{cases} \tag{4.26}$$

The parameter c_l is a small factor and $I(x, y)$ is the entire input. In practice, I have found that values of c_l between 1.2 and 2.5 work well. In the simulations described in this chapter, $c_l = 2.25$. The result of this thresholding operation is an increase in the number of locations considered to be in the region of lowest APF.

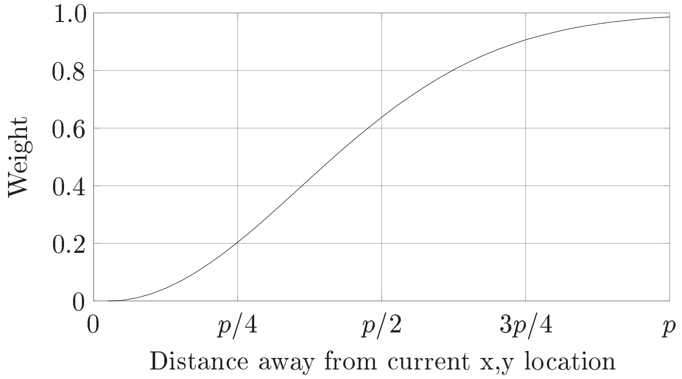


Figure 4.9: Profile of the inverted Gaussian used for the weighting factors $W(r, s)$ in Equation 4.27. This gives low weight to units near 0 (the current location) and increasingly more weight to units farther away, up to l_p units away.

Inverted Gaussian Weighting on the Slant Sum

The fourth modification involves the computation of the integration direction map. I have found it useful to add a nonlinear weighting factor on the inner summation term in Equation 4.22 on page 150. This changes Equations 4.22 and 4.23 to be the following:

$$\text{wssum}(x, y, r, s) = \sum_{r', s' \in I_p(r,s)} \left[W(r', s')\, \bar{O}_{3s}(x + r', y + s') \right] \tag{4.27}$$

$$D(x, y) = \{\, (r, s) \;:\; \text{wssum}(x, y, r, s) = \min_{r,s \in I_n(x,y)} [\text{wssum}(x, y, r, s)]\,\}. \tag{4.28}$$

The weighting terms $W(r, s)$ are found by point-sampling an inverted, two-dimensional circular Gaussian of standard deviation $3/l_p$. The cross-section of the shape of the Gaussian is shown in Figure 4.9. It gives low weight to points near (x, y) and increasingly greater weight to points farther away.

The purpose of this weighting is the following. The set of $\bar{O}_{3s}(x, y)$ values can be imagined as a landscape, similar to the landscape formed by the bar graphs in Figure 4.8. The lowest point in this landscape corresponds to the area interpreted as being closest to the viewer. The goal of the integration process is to add up values while moving away from this low region. The goal of the step computing the integration direction map (embodied by Equations 4.17 and 4.23) is to find the direction in which the integration process should go at each location. But the landscape of values will not generally be as smooth as shown in the example of Figure 4.8; instead, the landscape will be quite

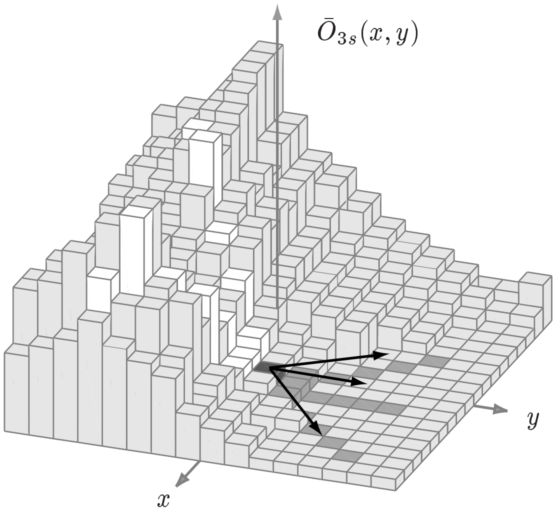


Figure 4.10: A landscape of frequency values and the problem of ambiguous directions. Each vertical bar represents the value of $\bar{O}_{3s}(x, y)$ at a particular (x, y) location. These values are used in Equation 4.28 to choose the direction having the lowest sum of values along different paths. But it sometimes happens that several directions from a given (x, y) location have identical sums, as shown above with the dark gray-colored paths. Which direction should then be chosen? The resolution mechanism implemented here takes each of the equally-valid directions, looks at the sum in the opposite directions (along the paths indicated by the white bars), then selects from the equally-valid directions the one whose sum in the opposite direction is greatest. The effect is to pick directions away from steep areas in the frequency sum landscape. In this example, the direction that would probably be chosen is the middle of the three shown above.

bumpy, with basins of lower average peak frequency values than surrounding hillsides. Such basins of local minima tend to attract the slant integration directions. By giving greater weight to values of $\bar{O}_{3s}(x, y)$ farther away along the path of l_p cells, the integration directions that are chosen for points near a basin are less likely to head straight into the basin. This helps the process look past such dips in the landscape.

Choosing From Among Equally Good Directions

In the computation of the integration direction map, it sometimes happens that among the eight possible directions away from a location (x, y), there is more than one immediate neighbor with the same sum in Equation 4.28. This happens most often in the vicinity of the region of lowest peak frequencies. The model must then choose from among several equally valid directions.

Using a policy of, say, always picking the first direction from a list of choices tends to bias the direction map. Instead, it turns out to be more effective to use the following simple strategy. First, for a series of n consecutive neighbors all lying along directions that produce the same minimum sum in Equation 4.28, examine the sums in the diametrically *opposite* directions. Then, pick the direction whose sum in the opposite direction is greatest.

The idea is to resolve ambiguous directions by picking the direction that is mostly likely to be heading *away* from regions of higher average peak frequencies. This is illustrated in Figure 4.10 on the preceding page. The reasoning behind this is that the purpose of the sums computed by Equation 4.28 is to find the direction heading *towards* the regions of lowest APF; thus, picking the direction heading *away* from the highest APF is an alternative that follows the same principle, but works when the sums in Equation 4.28 do not directly determine the best direction.

Interpolating Diagonal Values

The final modification is another enhancement to the slant integration step that helps to overcome the rectangular bias inherent in using a rectangular array of values. The problem is that, on a rectangular grid, the centers of units lying on diagonals are farther away from each other than those lying parallel to the horizontal or vertical axes. If this distance effect is not accounted for, the integration direction map (Equation 4.28 on page 152) can show biases in the horizontal and vertical directions. This results in curved surfaces having a squarish appearance.

To overcome this, I added interpolation to the computation of the integration direction map. The purpose of the interpolation is to compute an approximation to the values along diagonal paths. The modification affects Equation 4.27 on page 152. The formula remains unchanged for computing sums along the horizontal and vertical directions, but changes for computing the sums along the diagonals:

$$\mathrm{wssum}(x,y,r,s) = \begin{cases} \sum\limits_{r',s' \in I_p(r,s)} \left[W(r',s')\, \bar{O}_{3s}(x+r', y+s') \right] & \text{if } x = r \text{ or } y = s, \\ \sum\limits_{r',s' \in I_p(r,s)} \left[W(r',s')\ \mathrm{diagval}(x,y,r',s') \right] & \text{otherwise.} \end{cases} \tag{4.29}$$

The operator *diagval* is ordinary bilinear interpolation (Pratt, 1991), adapted to compute diagonal values specifically in the map of $\bar{O}_{3s}(x,y)$ values. A formula for this interpolation is given in Appendix B.4.

This use of interpolation is only necessary to compensate for biases introduced by the rectangular nature of the image representation used in this simulation. A different representation—for example, a hexagonal grid—would not need this. The visual system of the brain does not appear to use a rectangular grid, and so presumably it does not need this kind of elaborate machinery.

4.3.5 Summary

Figure 4.11 on the next page illustrates the general architecture developed in this chapter. Compared to Sakai and Finkel's (1995) original model as depicted in Figure 4.7, the modified architecture uses a greater number of complex cells tuned to more spatial-frequency and orientation bands. The new model also includes averaging, filling-in, and thresholding mechanisms in the slant integration stage. Certain additional modifications are not shown in Figure 4.11, in particular the use of a zero-discounting averaging procedure (Equation 4.19), and additional mechanisms in the final integration process.

Table 4.1 on page 157 summarizes the different parameters in the model. Each of these parameters needs to be given a value for simulation purposes. The values and how I set them are discussed in the context of Experiment Two below.

4.4 Experiment Two: Texture-Based Surface Shape Estimation for the Case of Single Surfaces

This section presents the results of experimental testing of the model described in the previous section. The goal of this experiment was to verify that the model works as intended, as well as to explore qualitatively its general performance.

The prediction underlying this experiment is simple: given images of single, textured surfaces, the simulation should produce representations of the estimated shapes of the surfaces. The estimates should be reasonable approximations to the true shapes, in the sense that there should be clear similarities between the estimates and the true surface shapes. This test is qualitative—the goal is for the simulation to achieve an approximate analysis that agrees with human perception of the surfaces, not to obtain a quantitatively exact representation.

4.4.1 Methods

I implemented the simulation of the model as an addition to the complex cell simulation detailed in the previous chapter. As before, the implementation consisted of a mixture of code written in the language C (Kernighan & Ritchie, 1988) and the Matlab environment (MathWorks, 1998a).

All inputs were 256-level gray-scale images 512 × 512 pixels in size. The images used in this experiment came from a variety of sources, and are shown for reference in Figure 4.12 on page 158. Each image depicts a single textured surface. The textured surface in image A consists of a regular pattern of ellipses wrapped around a cylinder; it was created using

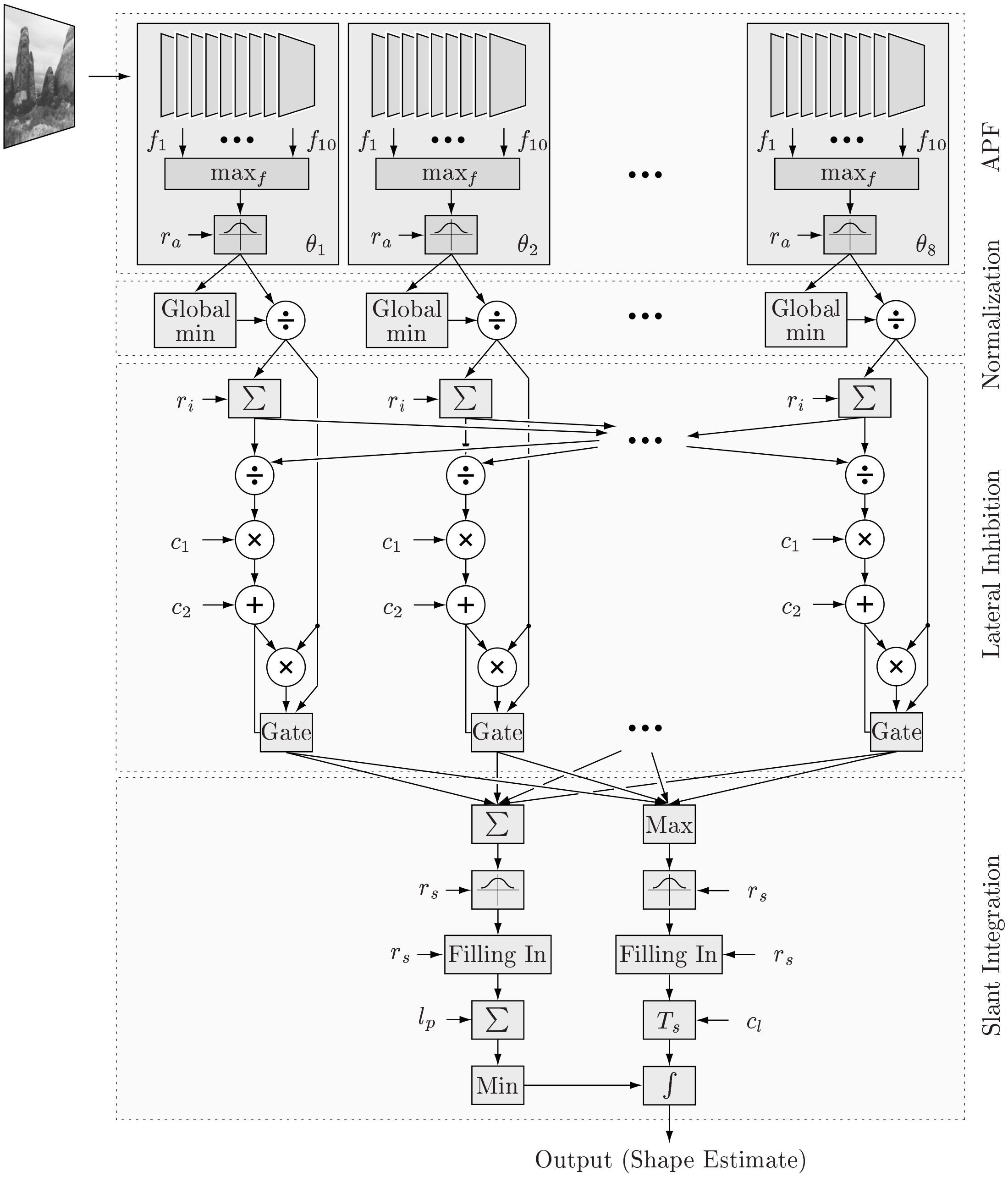


Figure 4.11: The general architecture of the modified version of Sakai and Finkel's(1994, 1995) model. In the implementation used in this research, there is a greater number of frequency and orientation bands, and additional mechanisms in the slant integration stage.

Table 4.1: Parameters in the shape estimation model

Parameter	Symbol	Description	Equation
APF averaging neighborhood	r_a	Radius of the neighborhood of cells over which frequencies are averaged in the computation of APF maps $O_2(x, y\,; \theta)$	4.12
Averaging neighborhood for lateral inhibition	r_i	Radius of the neighborhood of cells over which normalized APF values $\widetilde{O}_2(x, y\,; \theta)$ are averaged in the computation of the lateral inhibition coefficient $B(x, y\,; \theta)$	4.16
Averaging neighborhood for slant integration	r_s	Radius of the neighborhood of cells over which values in $\bar{O}_{3s}(x, y)$ and $\bar{O}_{3m}(x, y)$ are averaged prior to slant integration	4.20
Scaling factor for lateral inhibition	c_1	Dimensionless constant used for scaling the ratio of $O_3(x, y\,; \theta)$ responses in computing the lateral inhibition coefficient $B(x, y\,; \theta)$	4.16
Offset constant for lateral inhibition	c_2	Dimensionless constant used for additively adjusting the lateral inhibition scaling coefficient $B(x, y\,; \theta)$	4.16
Scaling factor for zero slant region threshold	c_l	Dimensionless factor used to scale the lowest APF value in $O_3(x, y\,; \theta)$ in order to calculate a cut-off threshold	4.25
Path length for slant direction estimation	l_p	Length (in number of cells) of the path used in computing direction of maximal surface gradient, for choosing direction of integration at each location	4.22
Number of depth iterations	n_s	Number of iterations for which the slant integration computation is performed at each location	4.25

a commercial software package for three-dimensional solid modeling. Images B, C, D, and I were created by first printing a texture pattern onto a sheet of white writing paper, then wrapping the paper around a small cylinder (for B and I) or affixing it to a flat surface in a way that created undulations (for C and D), and finally photographing the results. I obtained the photographs using a digital camera with a 75 mm telephoto lens at a distance of one meter. The use of a telephoto lens approximates the orthographic image projection assumed by the shape estimation component described in this chapter. The texture in B is composed from regularly-spaced black lines; the textures in images C and

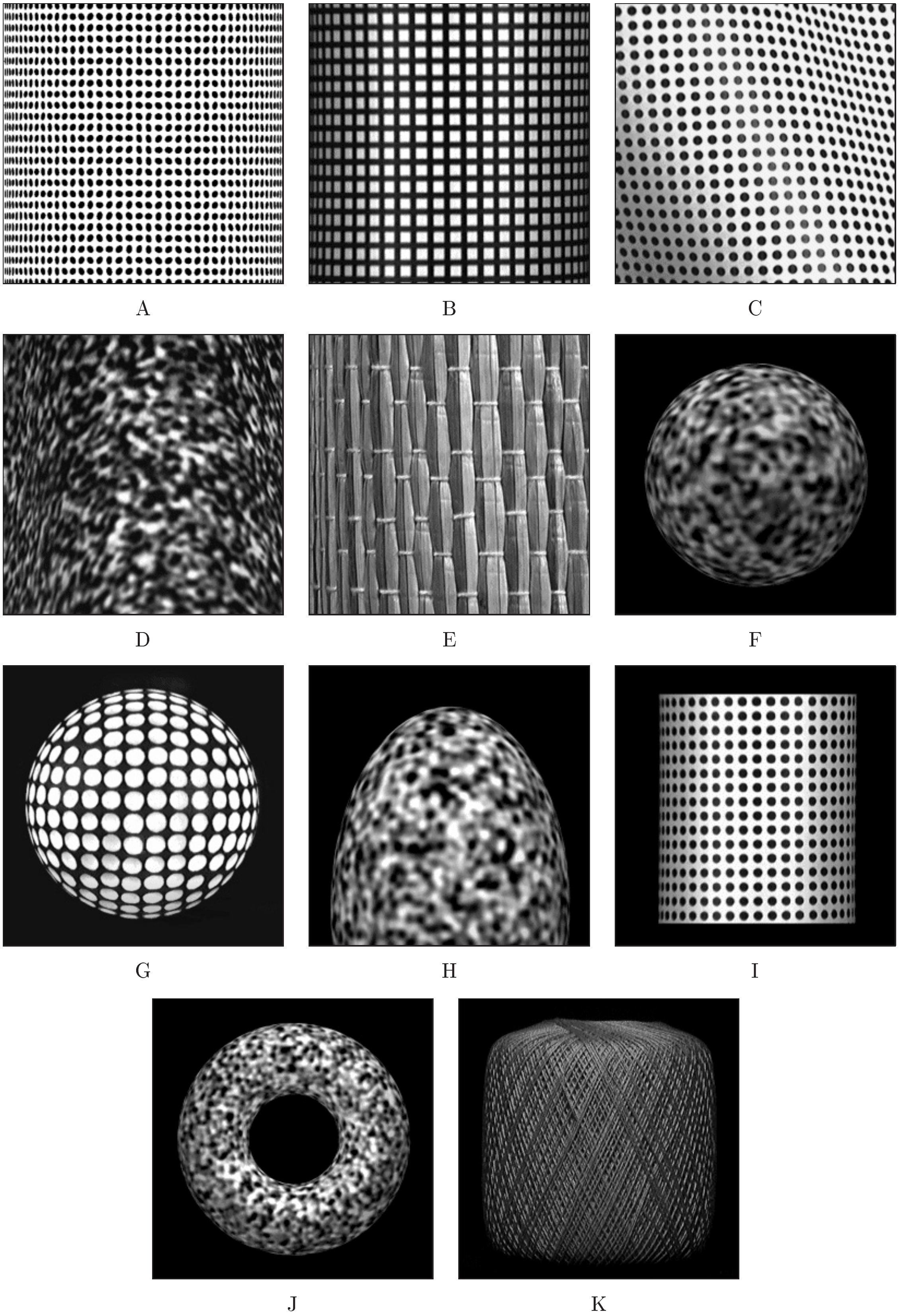


Figure 4.12: Images used as inputs for Experiment Two. More information about the images and how they were created is provided in the Methods section.

I are simply regularly-spaced black dots; and the texture in D was created by generating an array of random numbers, filtering the array with a circularly-symmetric Gaussian filter, and finally performing histogram equalization on the result. The image in E is a photograph of a portion of a woven straw mat wrapped around a small cylinder, using the same lens arrangement as for the previous cases. Similarly, G is a photograph of a painted golf ball, and image K, of a roll of string. Finally, F, H, and J are software-generated images created using a ray-tracing rendering package (POV-Ray™ by POV-Team™, 1997). The textures on the surfaces in these images are filtered random noise mapped onto the surfaces in software.

I cropped the first two images shown in Figure 4.12 on the preceding page right up to the sides of the cylinders in order to avoid issues of segmentation and blank areas. For images containing surfaces that did not span the full image area, such as the textured sphere, I segmented the images manually and provided the simulation with a representation of the region of interest. This segmentation method consisted of finding the nonzero values in $\max_\theta[O_2(x, y\,; \theta)]$, the output of the second stage of the model (see Section 4.3.2).

Running the simulation involved a two-step process. For each image, I first generated unnormalized APF maps from the complex cell responses by using Equation 4.12 on page 137, producing a total of eight unnormalized APF maps for each input image (one map per complex cell orientation). Because computing the complex cell responses is a time-consuming step, I stored these data in hard disk files in preparation for the next step in the simulation. In the second step, I ran the simulation described in Section 4.3 to compute the shapes of the surfaces in the input images based on the unnormalized APF maps.

The simulation produces two types of results: a two-dimensional representation of the integration direction map, and a three-dimensional representation of the estimated surface shape (see the discussion of the fourth stage in Section 4.3.4). Although the input images have size 512×512 pixels, the outputs of the implementation are expressed on a grid of size 128×128. The size reduction comes about from the subsampling operation in Stage Two of the model (Section 4.3.2).

Setting Parameter Values

Choosing reasonable values for the parameters listed in Table 4.1 on page 157 is without question the most arduous part of making the simulation work. Sakai and Finkel's (1995) prior work offers some guidance for setting some of the parameters, but differences in the details of the present implementation and the presence of new parameters required all the values to be reevaluated.

Table 4.2: Default parameter values used for simulations reported in this section

Parameter	Value	Parameter	Value
c_1	3.0	r_a	40
c_2	-0.25	r_i	22
c_l	2.25	r_s	22
l_p	30	n_s	140

Choosing reasonable values for the parameters is a difficult and ill-constrained problem. The parameters are not independent: nearly all of them interact and have an impact on the quality of the final shape estimate. In order to find values for them, I varied each parameter over a wide range and examined the outputs of the simulation on all of the test images shown above, as well as additional images (not shown). I searched by trial-and-error for a set of values that led to acceptable shape estimates for all of the input images. Table 4.2 shows the values that generally produced the best results. Except where noted, all of the simulation results used the parameter values shown in the table.

4.4.2 Results

In all cases, I examined the outputs of the simulation visually and evaluated how reasonable the outputs were in terms of their resemblance to the shapes in the input images. The results for each input image are presented in the following paragraphs. I use two types of display formats to present the simulation outputs in the figures that follow:

- *Arrow diagram of the tilt direction maps.* This shows the estimated direction of tilt at each location of the input image. As mentioned above, the tilt direction is the opposite of the direction of integration taken by the model when computing the results of Equation 4.25 on page 151. The orientation granularity of the tilt direction map is 45°; that is, there are eight possible directions at each location, arising from the way that the directions are computed in Stage Four of the model. (The fact that there are also eight complex cell orientations is a coincidence.)

  The full-sized tilt direction maps have 128 × 128 points; however, to make them legible in the figures, the maps have been subsampled to size 32 × 32 prior to plotting. This means that each arrow in each direction map shown in the following figures represents a 4×4 area of the actual tilt direction map used in computations.

- *Surface plot of slant integration results.* This format provides a three-dimensional representation of the surface computed by the fourth stage of the model described in Section 4.3.4. It is produced by plotting a small patch for the depth value at each location computed by the integration process. In these plots, depth is relative across the estimated surface, and the viewpoint is above the surface looking down at it.

Although the shapes depicted as the outputs of the simulation are explicit three-dimensional reconstructions, this is not intended to imply that I assume the brain's representation of surface shape is likewise explicit and three-dimensional. The use of a three-dimensional depiction here is merely a convenient device for expressing the results of the shape estimation process. The actual representation used by the brain is unknown, and is probably implicit in the pattern of activity of certain networks of neurons (Sakai & Finkel, 1995).

Test Image A

Figure 4.13 on the following page presents the outputs of the simulation for the first test case, an image of a vertical cylinder facing the viewer. As mentioned above, the image of the cylinder has been purposefully cropped tightly in order to avoid issues of segmentation. Figure 4.13 shows the input image in the upper left, the computed tilt direction map in the upper right, and in the bottom row, two views of the three-dimensional representation of the final, approximate surface representation.

Looking at the test image, one expects that the location of the region of lowest APF in the input (and thus the area closest to the viewer) should be the middle portion of the surface, extending vertically across the whole image. The surface should fall away from the viewer on the left and right sides. This is indeed what the simulation results exhibit in both the tilt direction map and the 3-D surface representation.

The dotted region in the tilt direction map of Figure 4.13 indicates the region of lowest APF across the surface, computed from the thresholding step described in Section 4.3.4. It shows that in this region, the depth and the slant are both taken to be zero, and the tilt direction is estimated to be towards the viewer. The slant integration process (Section 4.3.4) assigns a relative depth of zero to these locations. On either side of this middle band, the tilt directions point away horizontally, again consistent with the true shape of the surface.

The central portion of the surface representation in the bottom row of Figure 4.13 has a somewhat flattened appearance. This portion of the output corresponds to the region of lowest normalized APF (i.e., the global $\bar{F}_{\min}$ region discussed in Section 4.2.1 and calculated in the simulation using Equation 4.26). It corresponds to the region of zero

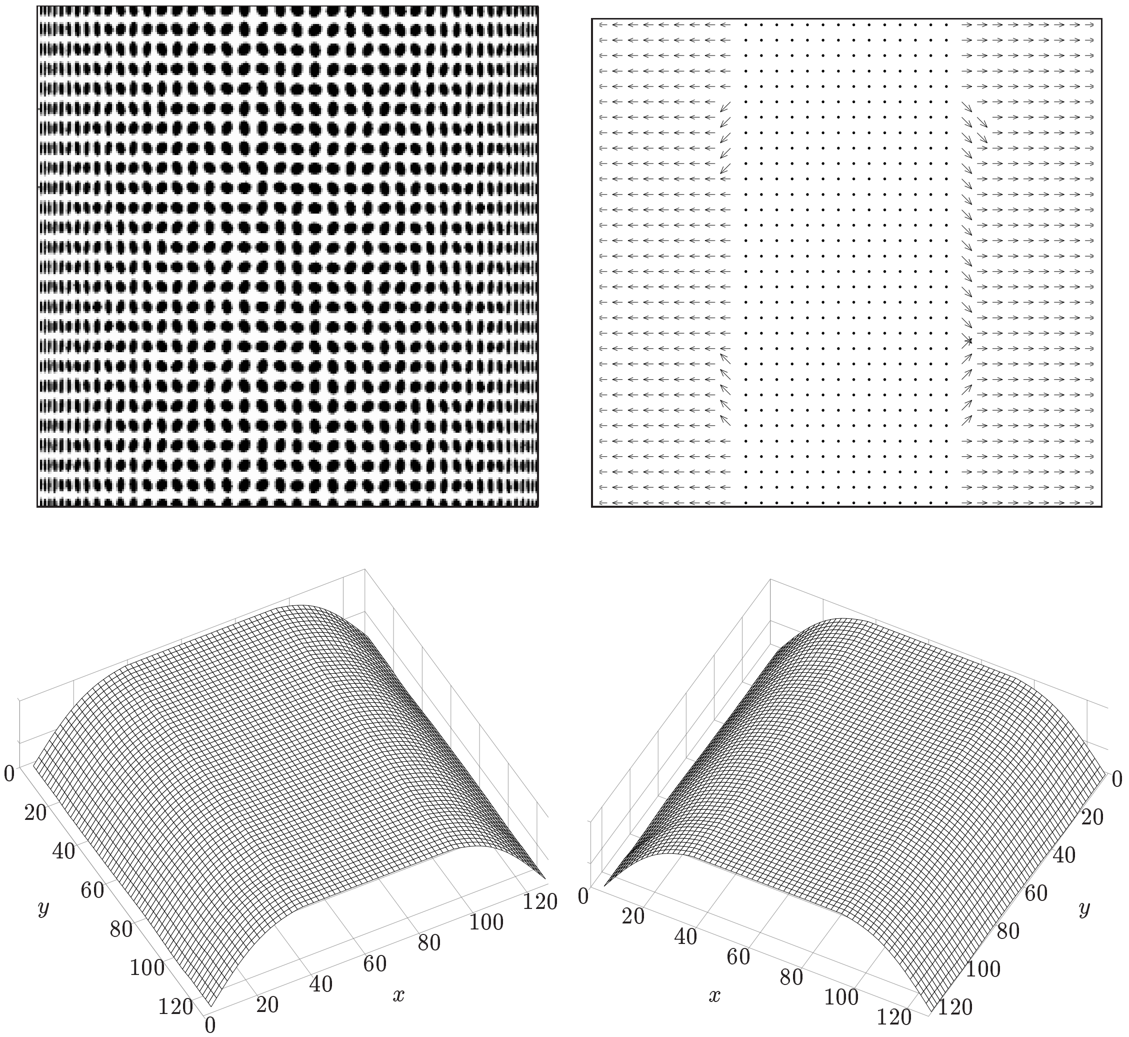


Figure 4.13: Shape estimation results for test image A. The input image is reproduced in the upper left. It has size 512 × 512 pixels. The tilt direction map is shown in the upper right; its actual size in the simulation output is 128 × 128 locations, but it has been reduced to size 32 × 32 in order to make it legible in this figure. The orientation granularity of the direction map is 45°. The surface rendition shown in the bottom row is a plot of the shape estimated by the model for the given input image. The surface is shown plotted from two vantage points; in each plot, the observer's viewpoint is an imaginary location above the surface and looking down at it.

slant. The flatness is a consequence of assigning a depth of zero to locations throughout that region during the integration stage of the model (Section 4.3.4). Although the surface rendition for this case appears to be too flat, it is worth noting that it is difficult to get an impression of curvature in the middle third of the input image. The strongest impression of curvature occurs towards the sides of the cylinder. So in this sense, the shape estimated by the model is at least partly consistent with human subjective interpretation.

It is actually possible to reduce the flat appearance on a case-by-case basis for different test images by adjusting the parameters shown in Table 4.1 on page 157. Unfortunately, the flatness is affected by nearly all of the parameters together, so it is difficult to give a recipe for adjusting them to produce an optimal result. The parameters that make the greatest difference are: c_1, c_2, c_l, l_p, and the averaging radii r_a, r_i, and r_s. The parameter values used in the experiments described here were chosen to yield adequate results on (nearly) all inputs, the goal being to make the model be as unsupervised as possible and avoid making adjustments for individual cases.

Test Image B

The outputs of the simulation for the second test input are shown in Figure 4.14 on the following page. The image is similar to the previous test case—a vertically-oriented cylinder covered with a highly regular texture—but this time, it is an actual photograph and includes shading effects.

The model's estimate of the surface shape in Figure 4.14 is once again in good qualitative agreement with the apparent shape of the true surface in the image. The tilt direction map shows that the region of zero slant was correctly localized by the model to be in the middle center of the cylinder. The surface reconstruction shown in the bottom of the figure is excellent.

The tilt direction map in Figure 4.14 displays more errors than in the previous test case. Many of the locations next to the middle band have slightly incorrect tilt directions: instead of pointing away horizontally, they point diagonally away from the upper middle of the cylinder. This indicates that the upper middle of the cylinder was interpreted to have the lowest average peak spatial frequencies, and that the surrounding regions of the cylinder had slightly higher APF values. Strictly speaking, this is incorrect, because the texture is homogeneous and the upper and lower middle regions should be judged to be identical to the center. However, the fact that the texture elements are cut off at the top and bottom affects the spatial-frequency content in those regions, and this edge effect may be misleading the process of estimating tilt directions. Still, the errors are relatively small, and had little effect on the surface rendition in the bottom of the figure.

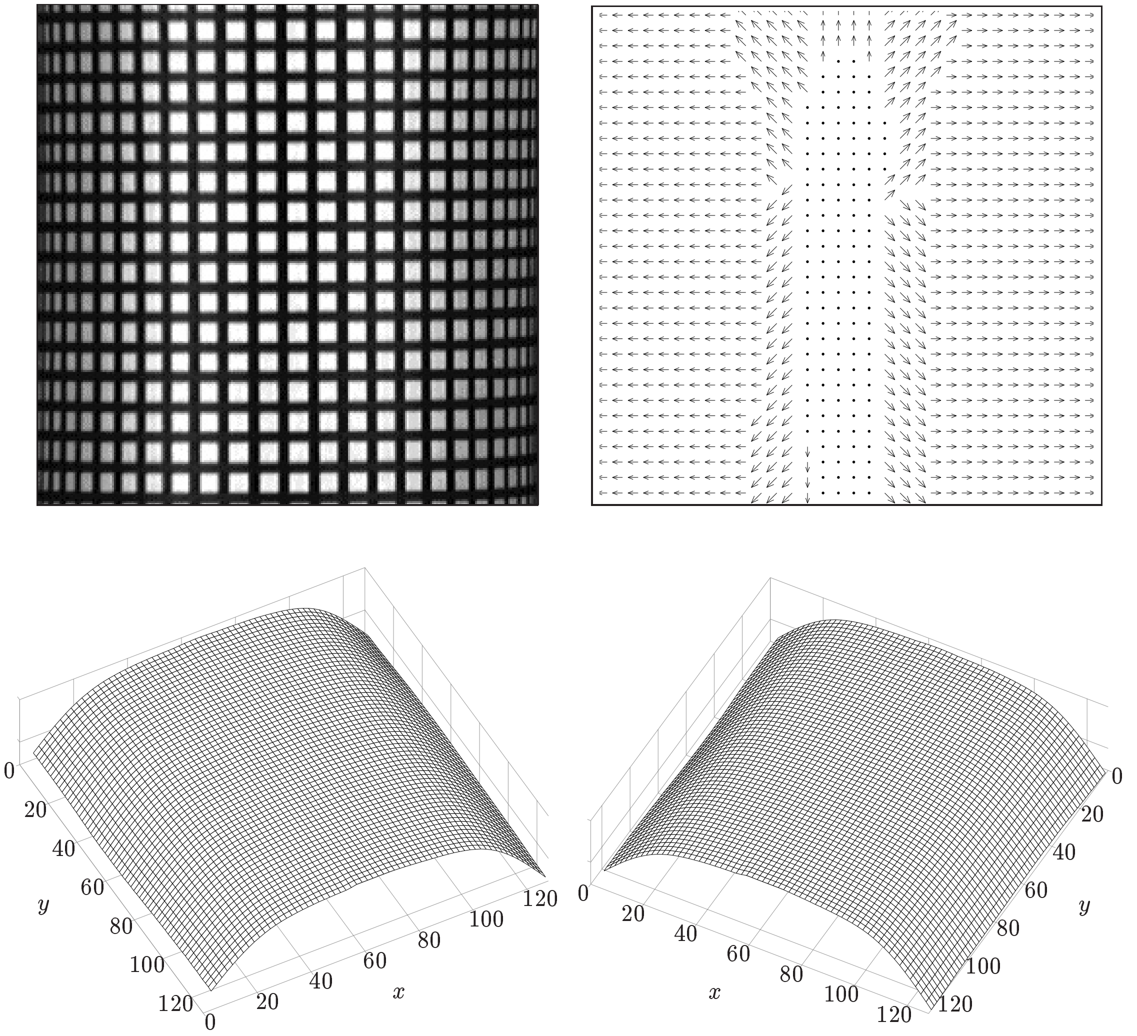


Figure 4.14: Shape estimation results for test image B. The format of this figure is the same as that of the previous figure.

This test case also demonstrates that the system can work with anisotropic textures. The texture pattern has strong vertical and horizontal orientation components, but these anisotropies did not affect the shape estimation process.

Test Image C

Figure 4.15 on the next page shows the results for the third input, a photograph of a curved surface viewed from a slight angle. The actual surface is slightly closer to the viewer at the bottom; it has a hump running diagonally from the bottom towards the upper left, and falls away from the viewer on either side of the hump. The raised hump is somewhat flatter near the top than the bottom of the image.

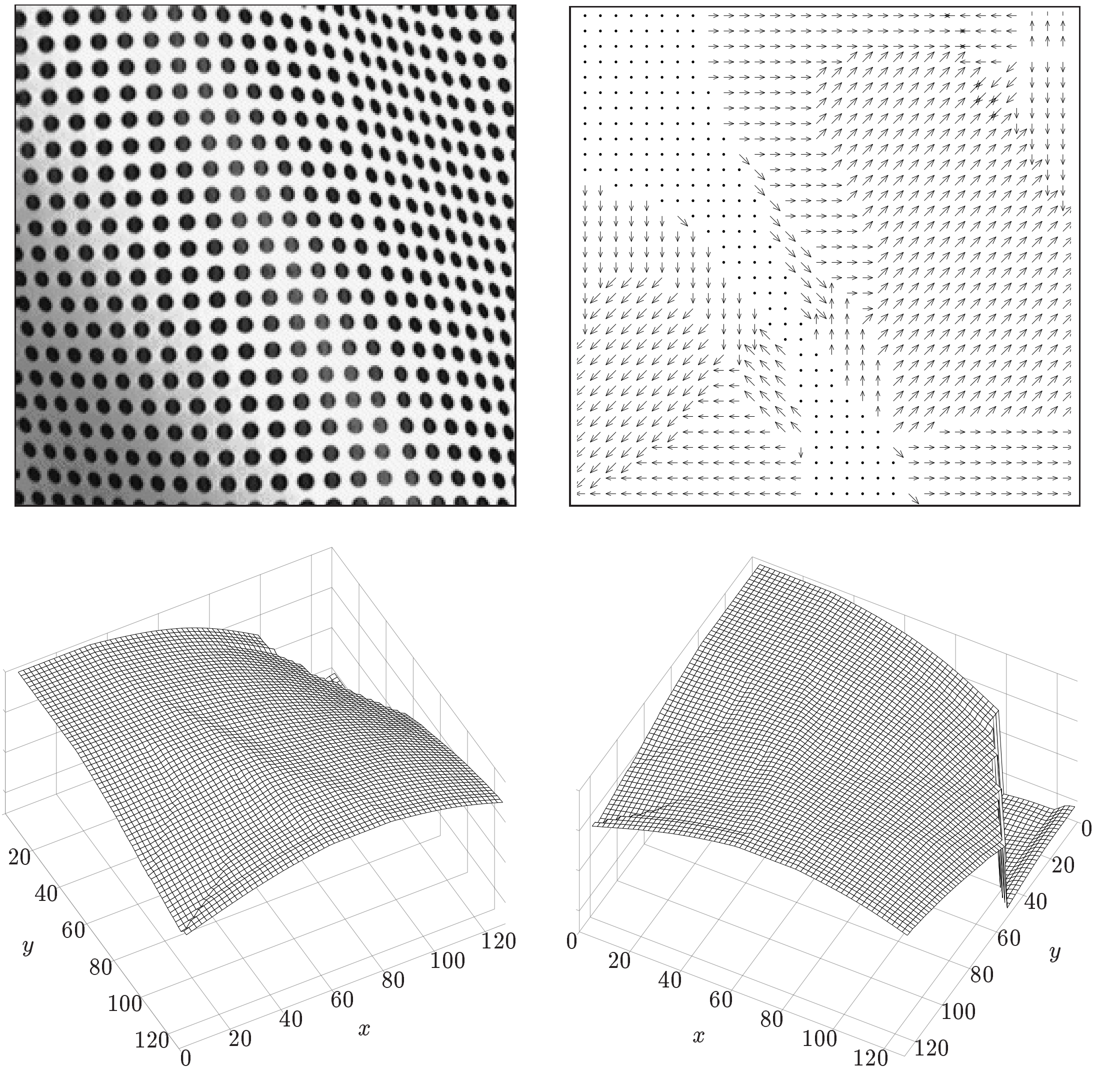


Figure 4.15: Shape estimation results for test image C.

Except for the region in the upper right-hand corner of the image, both the tilt direction map and the three-dimensional reconstruction shown in Figure 4.15 agree qualitatively with the perceived shape of the surface in the input. The region of lowest APF in the tilt direction map is roughly where it is expected. The estimates of the tilt directions are reasonably good, although there are a number of locations where the estimated tilt direction runs contrary to expectations.

The upper right-hand corner of the image demonstrates a new feature: the surface in this corner produced a break in the shape estimate and a reversal of the surface slope. This occurred because the surface in the corner is flattened out, and thus constitutes a reversal of the surface gradient compared to the region in the middle of the image.

The flattening out induced the model to interpret this area as having lower average peak frequencies, and therefore as being relatively closer to the observer than the region further in towards the middle. Since the hump and the corner were both interpreted as having lower APF than the area between them, the tilt directions were interpreted as heading towards each other, leading to a break in the shape rendition shown in the bottom of the figure.

This case shows that changes in slope can be problematic for the shape estimation routine: they can lead to sharp reversals in the tilt directions an rifts in the surface estimates. One way to handle this kind of input would be for the segmentation component to treat the corner region as a separate surface. A change in the surface gradient from a positive slope to a negative slope could be treated by the segmentation process as a sign of a surface change. Although this is not entirely correct—humans perceive the surface in this image as unitary, not as two surfaces—resolving this ambiguity may in fact require recognizing the object in the input image. Because the visual processes taking place at the level of this model do not concern themselves with object identities, a segmentation that breaks the surface into multiple parts may be the best that can be expected.

Test Image D

Figure 4.16 on the following page presents the results for the fourth test input, the image of a curved piece of paper covered with a random noise texture. The shape is that of a hump that falls away sharply on either side.

In order to produce a qualitatively correct result, this test case required a modification to the parameters of the simulation: the scaling factor c_l for thresholding the region of zero slant had to be greatly increased, to $c_l = 5.5$. Without this increase, the region of zero slant was limited to a small area at the bottom, and the tilt directions pointed towards the middle from both the top and bottom halves of the image. This produced the appearance of two offset cylinders buckled together in the middle.

With the increased value of parameter c_l, the simulation results in Figure 4.16 show reasonable qualitative agreement with the overall shape of the surface in the test image. The central region of the physical surface is off-center and towards the right in the image, and indeed the surface estimated by the model is also off-center in this way. The left side of the surface is somewhat longer than the right side, and the surface estimate matches this characteristic as well.

The sides of the reconstructed shape appear to be somewhat too long compared to the sides of the surface in the image. Although it is difficult to judge the true proportions of the sides in the image, they nevertheless do not appear to be quite as long as they end up

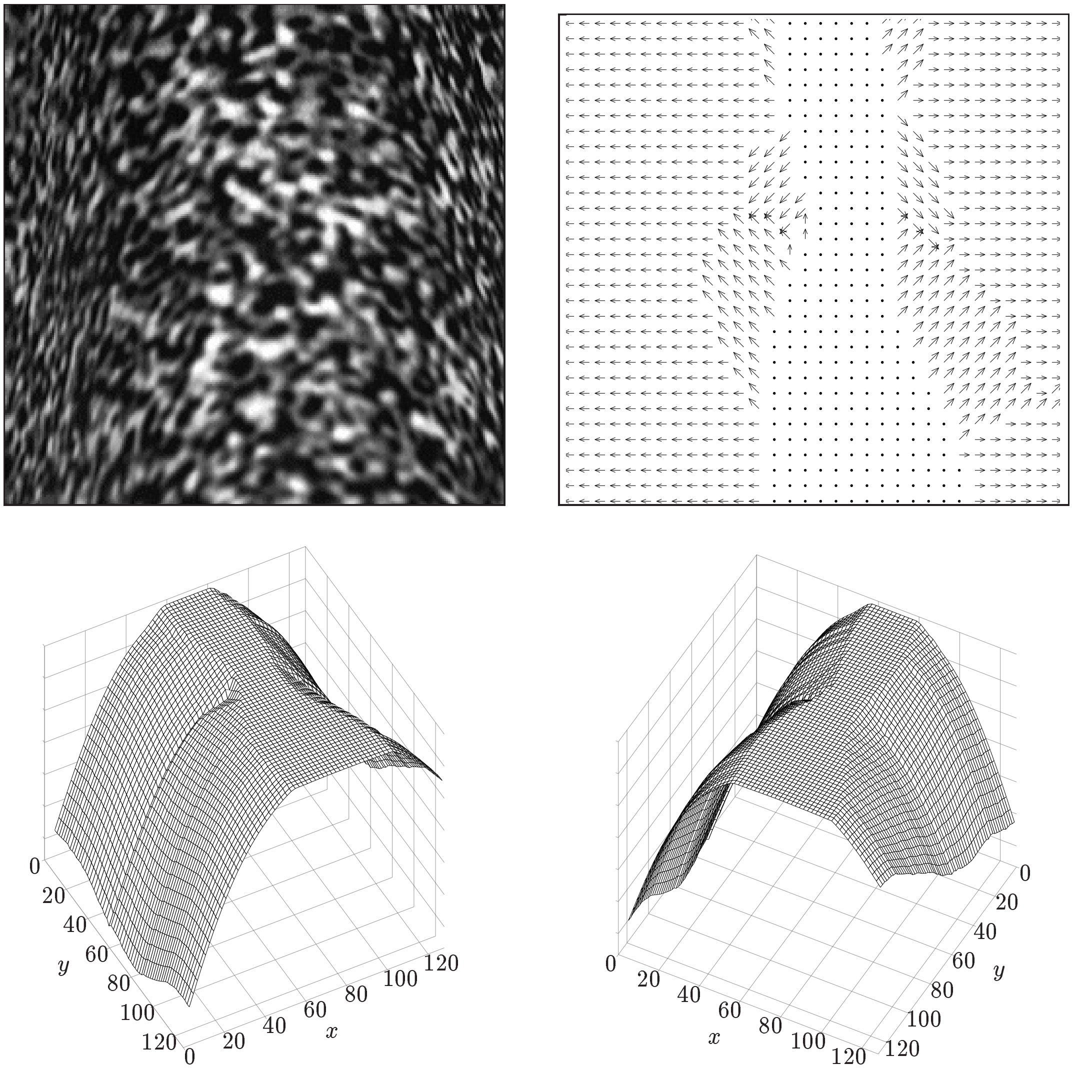


Figure 4.16: Shape estimation results for test image D using a modified value of parameter c_l.

in the model's output. It is not clear what led to this result. One contributing factor may be the noisiness of the texture itself. Unlike the previous test cases, which used regular textures, the texture on this surface is highly irregular. This can be expected to be more difficult for the model to handle than a regular texture pattern. This may also account for the slightly pinched appearance of the middle of the hump.

The need to adjust parameter c_l for this test case probably reflects a need for further research on how to best set the parameters in the model. For example, it may be appropriate to find the region of lowest APF using a different method, one that does not rely on a fixed factor such as c_l. I leave this kind of investigation for future work.

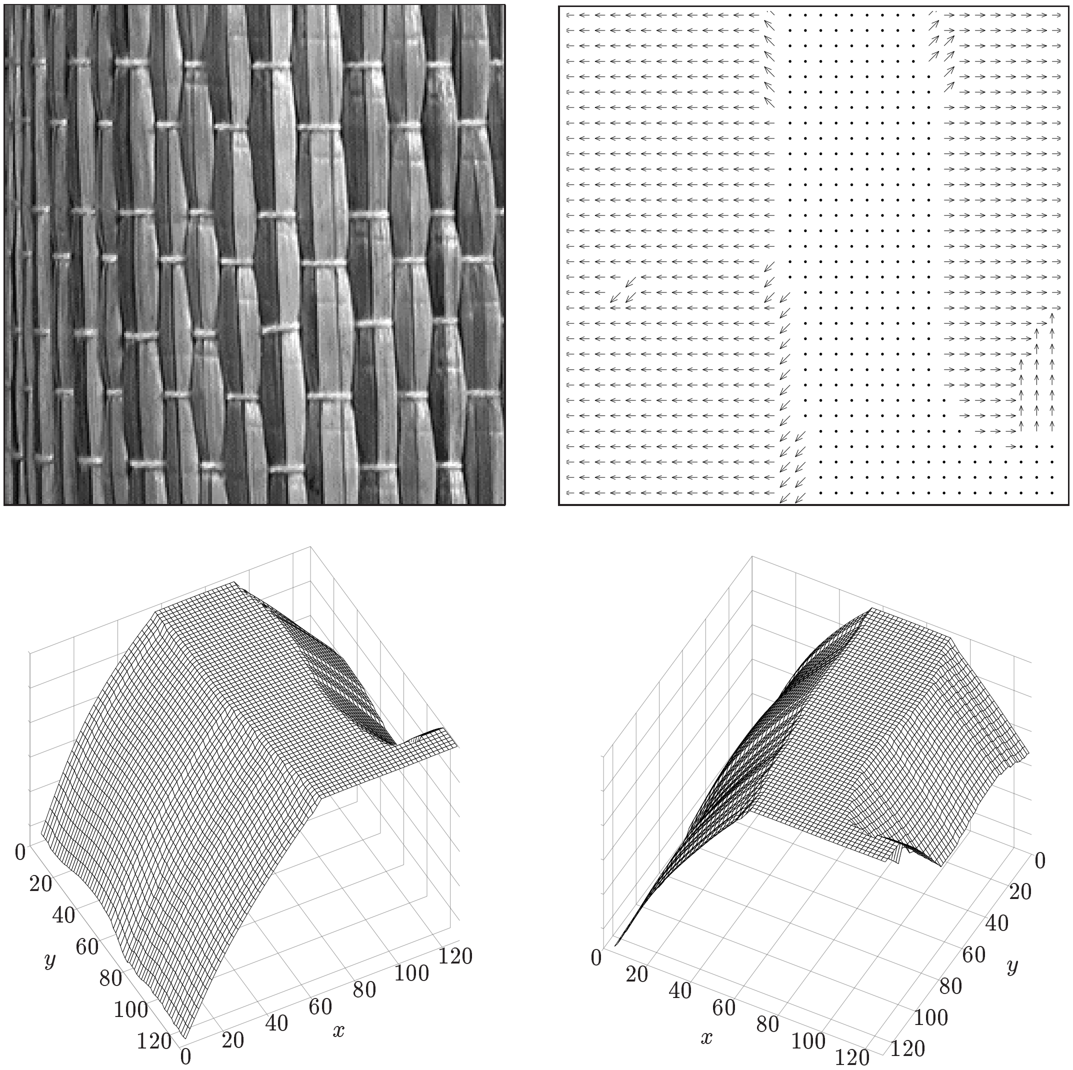


Figure 4.17: Shape estimation results for test image E.

Test Image E

Figure 4.17 shows the results for the fifth input. The surface pictured is a portion of a woven straw mat wrapped around a cylinder. The right edge of the cylinder is not visible in the photograph, and the region closest to the viewer is located off-center towards the right.

Figure 4.17 shows that the results for this case agree qualitatively with the shape of the surface in the image. The reconstructed shape is a bit too angular, but the tilt directions generally agree with expectations. There is a small step in the lower right portion of the shape rendition due to the fact that the area of lowest normalized APF encompasses the lower right-hand corner.

Once again, there is a significant flat area in the surface reconstruction. As noted above, the appearance of a flat region is endemic to the method used in the model for locating which area of the surface is facing the viewer. Again, it is possible to reduce the size of the flat area and produce a smoother surface estimate by adjusting the parameters in the model, but the goal of these experiments has been to use the same set of parameters as much as possible for all test cases. Unlike the situation with test image D, in which the output was not qualitatively correct unless one of the parameters was adjusted, the output here is still roughly in agreement with expectations. As an approximate analysis of the shape in the input, the representation is acceptable even with the prominent flat region.

It is interesting that in the area to the right of the middle, the surface has been interpreted to fall away from the viewer. This is technically correct, because the physical surface is a portion of a cylinder, but subjectively, the surface in the image does not appear to fall away noticeably. The texture on the surface in the image is not entirely homogeneous; the reconstruction errors may therefore indicate that the computations were led astray by the inhomogeneities in the surface texture. It is also possible that the spatial-frequency content in this input image reveals more about the shape of the surface than humans perceive in this case. A complicating factor for human perception is the flat, square border surrounding the image on the printed page. The presence of the surrounding flat area contradicts the shape in the image; this may lead our visual system to miss some of the subtleties of the shape.

Test Image F

Figure 4.18 on the next page presents the results for the sixth test case. The input is an image of a sphere covered with a random noise texture. This time, the image is not cropped to the edges of the surface; instead, the object is set on a black background.

The results shown in Figure 4.18 on the following page are excellent. The tilt direction map is nearly perfect within the limitations imposed by the 45° granularity of tilt directions—nearly all of the tilt directions point away from the center of the sphere. The shape rendition shown in the bottom row of the figure is also quite good, with a definite hemispherical appearance and a nearly symmetrical form.

There is a small protrusion in the front side of the surface reconstruction, such that the bottom edge of the surface is not entirely symmetrical. This is due to a difference in relative heights across the surface; the protrusion indicates the surface is interpreted to be farther from the viewer at that point than in other regions. Apparently the average peak spatial frequencies near that part of the surface border were slightly higher than

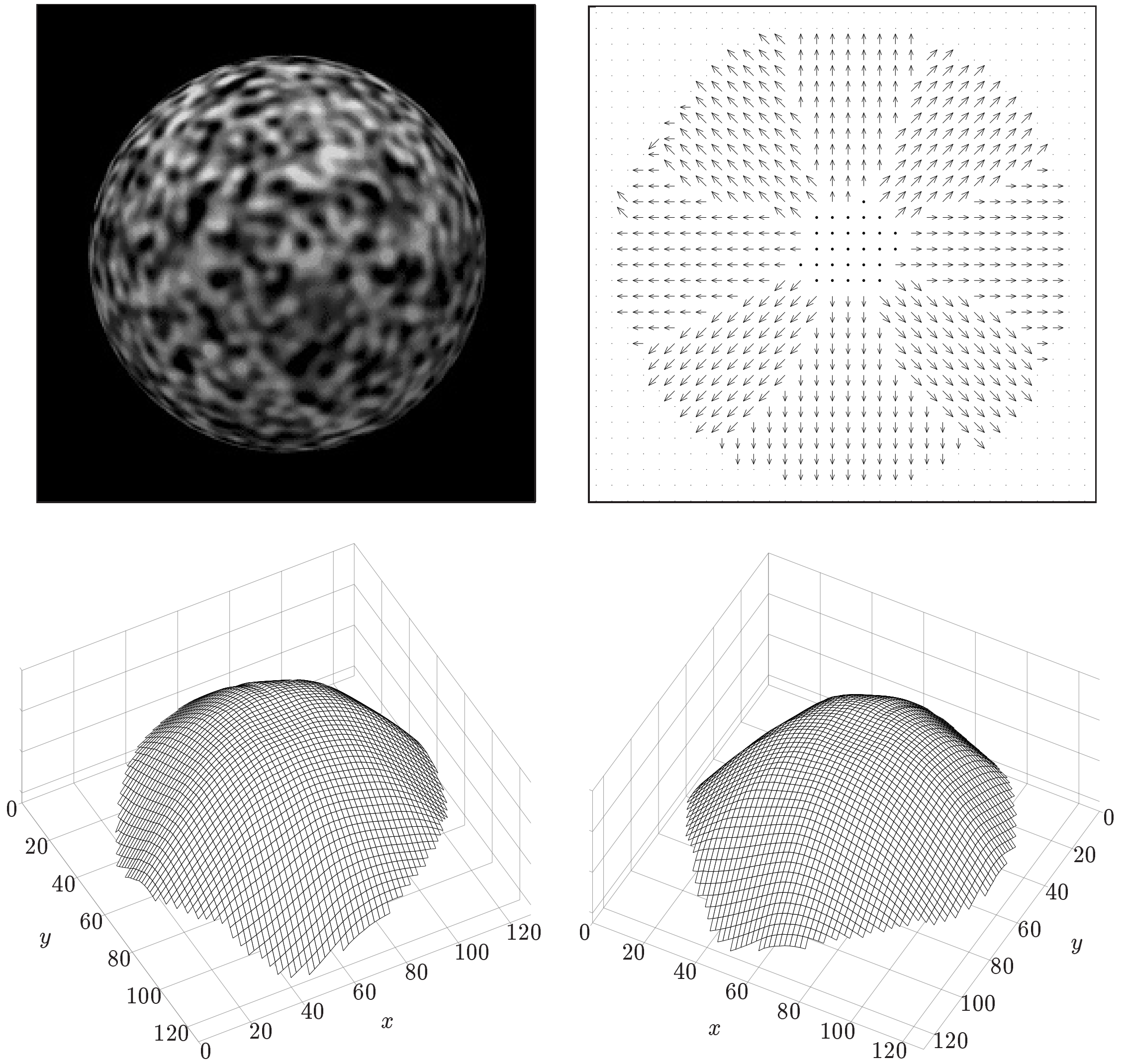


Figure 4.18: Shape estimation results for test image F.

at other points along the border, possibly due to inhomogeneities in the surface texture. Noise textures are not homogeneous, so these kinds of effects are to be expected. Overall, however, the results for this test case are outstanding.

Test Image G

Figure 4.19 on the next page presents the results for the seventh case. The surface is again spherical, as in image F, but this time it is a photograph of a painted golf ball.

Once again, the results in Figure 4.19 on the following page show good qualitative agreement with the perceived shape of the surface in the input image. The tilt direction map is excellent, with the tilt pointing in the expected directions at nearly every location.

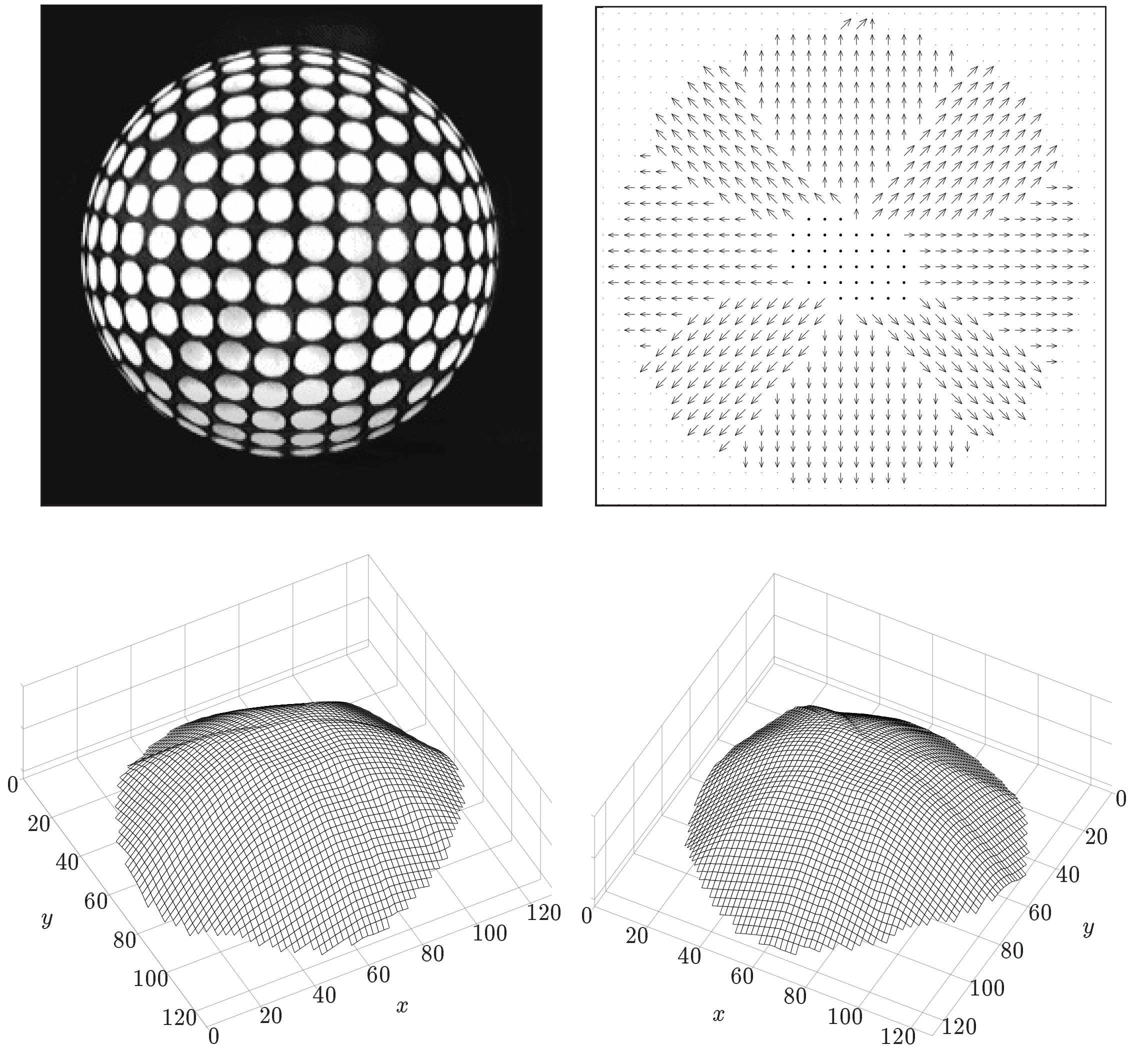


Figure 4.19: Shape estimation results for test image G.

The area of zero slant is oblong rather than circular as it should be, but it is located exactly in the center of the surface. Apart from the asymmetrical shape of the center, the main error in this result is that the estimated shape is too flat.

Test Image H

Figure 4.20 on the next page shows the results for the eighth input case, a portion of an ellipsoidal surface covered in a random noise texture. The top of the surface is visible in the image but the rest is cut off at the bottom.

The output of the simulation in Figure 4.20 agrees well with the surface in the image. The tilt directions at most locations in the output are oriented properly, facing away from

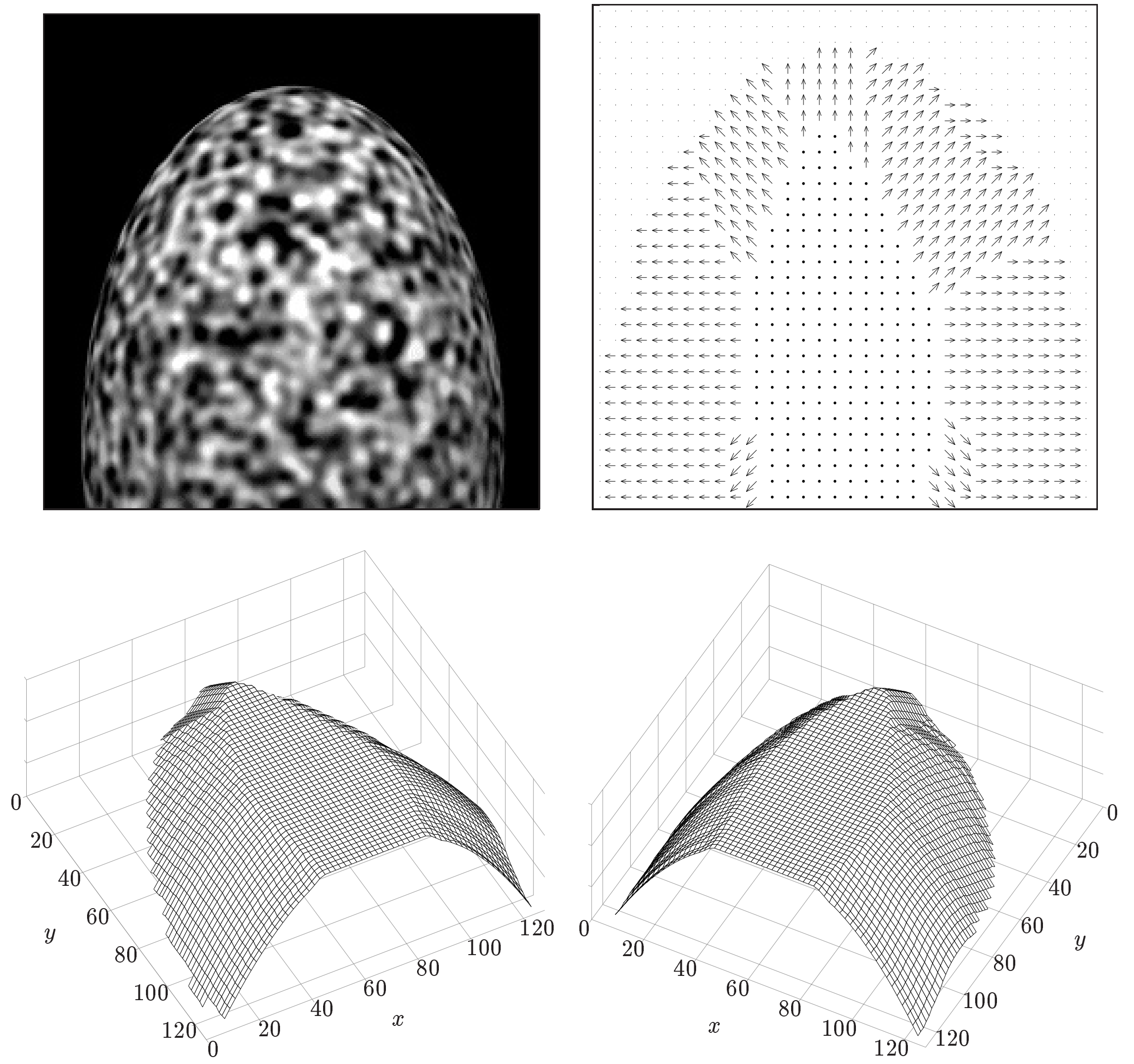


Figure 4.20: Shape estimation results for test image H.

the center of the textured surface. There is only a small amount of asymmetry in the tilt direction map and in the shape rendition. The main errors for this test case are that the upper edge of the surface in the output is too pointed, and once again there is a prominent flat area in the surface reconstruction.

Test Image I

Figure 4.21 on the following page shows the results for the ninth input, a photograph of a cylinder covered with a regular dot texture.

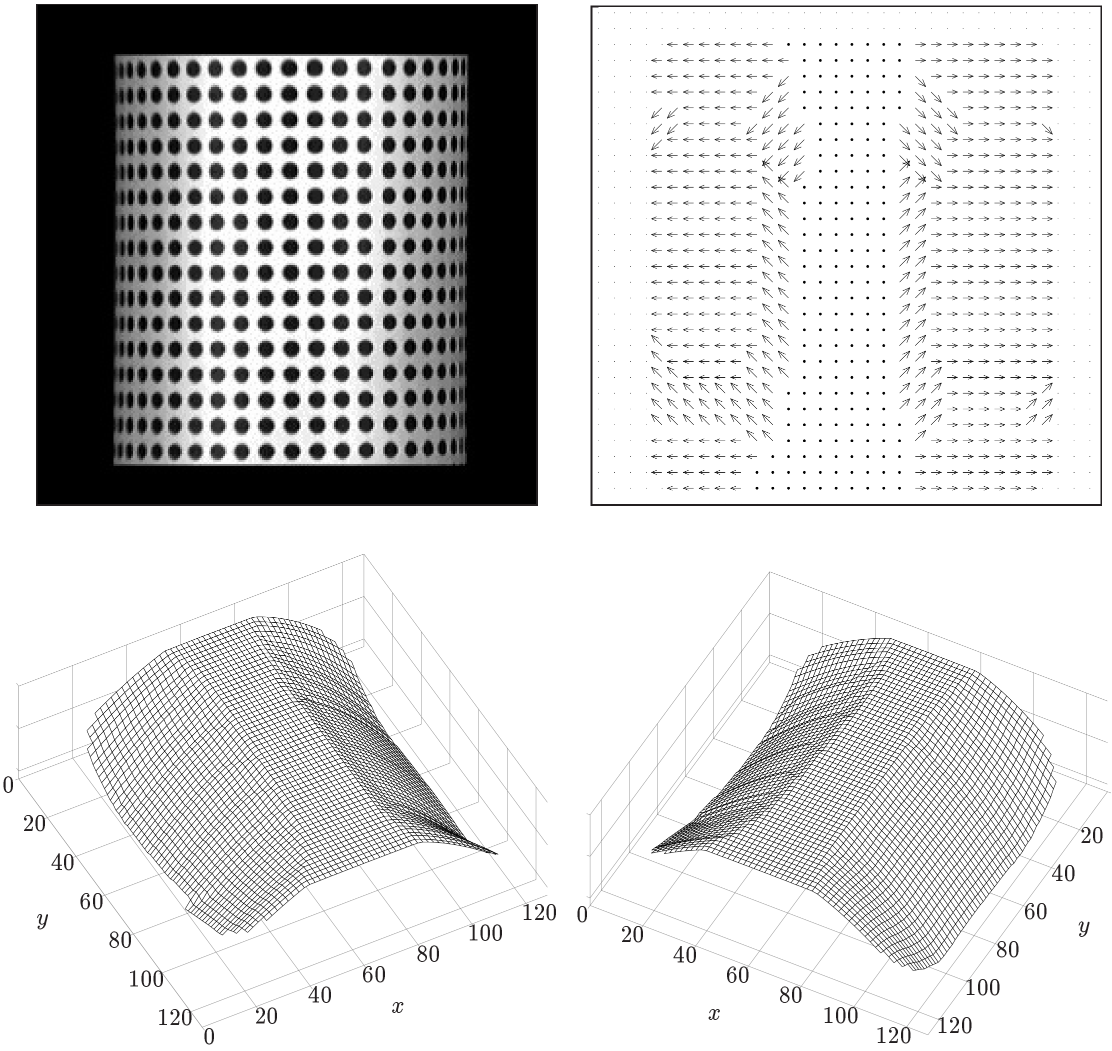


Figure 4.21: Shape estimation results for test image I.

The outputs shown in Figure 4.21 are reasonably good. The surface has a roughly cylindrical appearance, although the sides of the reconstructed surface are slightly too angular. The quality of the tilt direction map is fair; there is some amount of confusion near the center region, but across most of the surface, the tilt arrows point in the expected directions. On balance, the results for this test input are again successful.

There is an unusual feature of the results for this test case: close comparison between the input image and the outputs reveals that the size of the estimated surface is actually *larger* than the size of the surface in the image. Note, for example, how much closer to the edges of the output the tilt direction maps extends in Figure 4.21 compared to the surface in the input. In fact, close inspection reveals that this was also true of the outputs

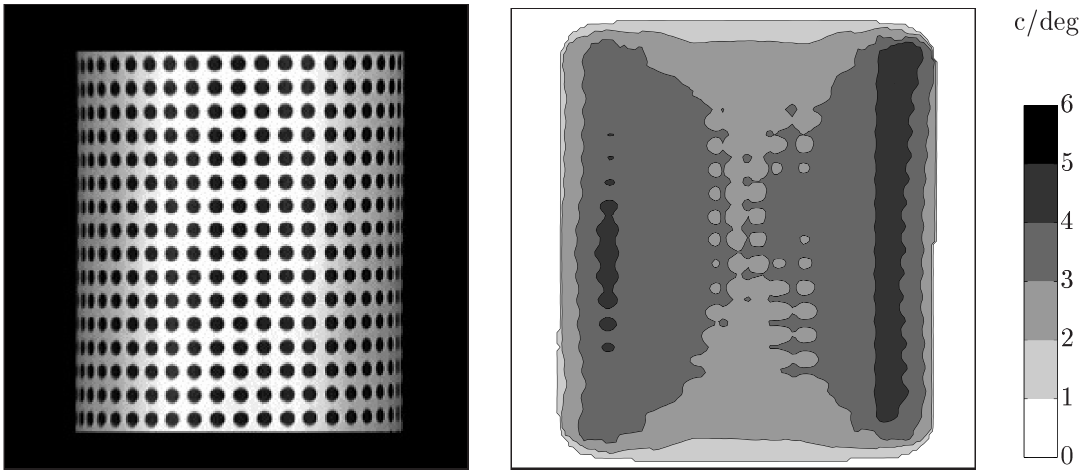


Figure 4.22: (Left) Input image I, used in the results shown in Figure 4.21 on the preceding page. (Right) Contour plot of the the maximum unnormalized APF values across all eight orientations in the image. The values shown here were computed using the formula $\max_\theta[O_2(x, y\,; \theta)]$, where $O_2(x, y\,; \theta)$ represents the results from Stage Two of the model as described in Section 4.3.2. The colorbar to the far right indicates the shades of gray corresponding to different average peak frequencies indicated by the contours of the plot. The contour plot shows that the extent of the responses exceeds the extent of the surface in the input image.

in the last three test cases: each of the results in Figures 4.18, 4.19 and 4.20 exhibit the same enlargement compared to the input images, and it also occurs in the test cases that follow. What is going on?

The source of the problem turns out not to be the slant estimation process, but rather its inputs, the complex cell responses. Figure 4.22 provides a clue to the source of the problem. This shows a contour plot of the quantity

$$\max_\theta[O_2(x, y\,; \theta)] \tag{4.30}$$

where $O_2(x, y\,; \theta)$ represents the results from Stage Two (Section 4.3.2) of the model for test case I. This is a single map representing the maximum value, across all eight orientations, of the unnormalized APF at each location in the input. It provides an indication of the locations where the complex cells respond across the input image, along with information about which cells produce the strongest responses. The figure shows that, as expected, the left and right vertical sides of the cylinder (areas of high texture compression) produce high APF values, indicating that cells tuned to high spatial frequencies responded in those regions. These show up in the contour plot as the darkest-colored vertical bands. But the figure also shows that further outward from the bands of high frequency responses, there are bands of lower APF values—responses from cells tuned to lower spatial frequencies. In other words, the transitions from areas of high texture compression to areas devoid of texture at the left and right cylinder sides are not abrupt as expected from the input image. This is also true for the upper and lower surface borders, which likewise produced responses from cells tuned to low spatial frequencies. Thus, the outputs of the complex cells result in a halo of low spatial frequencies surrounding the surface region in the image.

This halo is apparently narrow enough that it does not impair the shape estimation model's ability to infer the surface shape, but it does create the impression that the surface is larger than it actually is. In retrospect, this is to be expected given that the complex cells tuned to the lowest spatial frequencies have large receptive fields. It is likely that the low-frequency cells can be triggered relatively far away from the true border between a region containing texture and a region devoid of texture.

I have found that the use of a zero-discounting averaging procedure (Equation 4.19 on page 147) is critical to making this halo as small as possible. Alternative averaging methods, especially ordinary convolution, tend to cause the APF values near borders to go even lower, increasing the size of the low spatial-frequency halo seen in Figure 4.22 on the page before. The problem of eliminating this halo is an area ripe for further research.

Test Image J

Figure 4.23 on the following page shows the results for the tenth case, the image of a torus-shaped surface covered with a random noise texture and set on a black background.

The output for this case is not as good as for the previous test cases. Although the outer surface of the torus has been correctly interpreted as slanting away radially, the inner surface has been missed completely. The region of lowest APF has been interpreted as covering the entire center of the surface, but this is not the pattern that one would predict for this kind of surface. The inner surface should contain higher APF values than the area of the torus between the inner and outer slopes, and therefore should not be considered part of the area closest to the viewer. Thus, the results for this case are only partially correct—the general shape of a circularly-symmetric surface is visible in the output, but portions of the surface are incorrectly interpreted.

In an effort to understand the cause of the problem, I again examined the set of $\max_{\theta}[O_2(x, y\,; \theta)]$ values for this input. Figure 4.24 on page 177 presents the average of two cross-sections through the set of values. (Note that examining this type of cross-section is only useful for circularly-symmetric surfaces covered with isotropic textures, for which the set of values bears some similarity to the eventual shape estimate. It is not useful for more general surface types.) The cross-section reveals some interesting features.

First, there are two peaks in the set of $\max_{\theta}[O_2(x, y\,; \theta)]$ values on either side of the

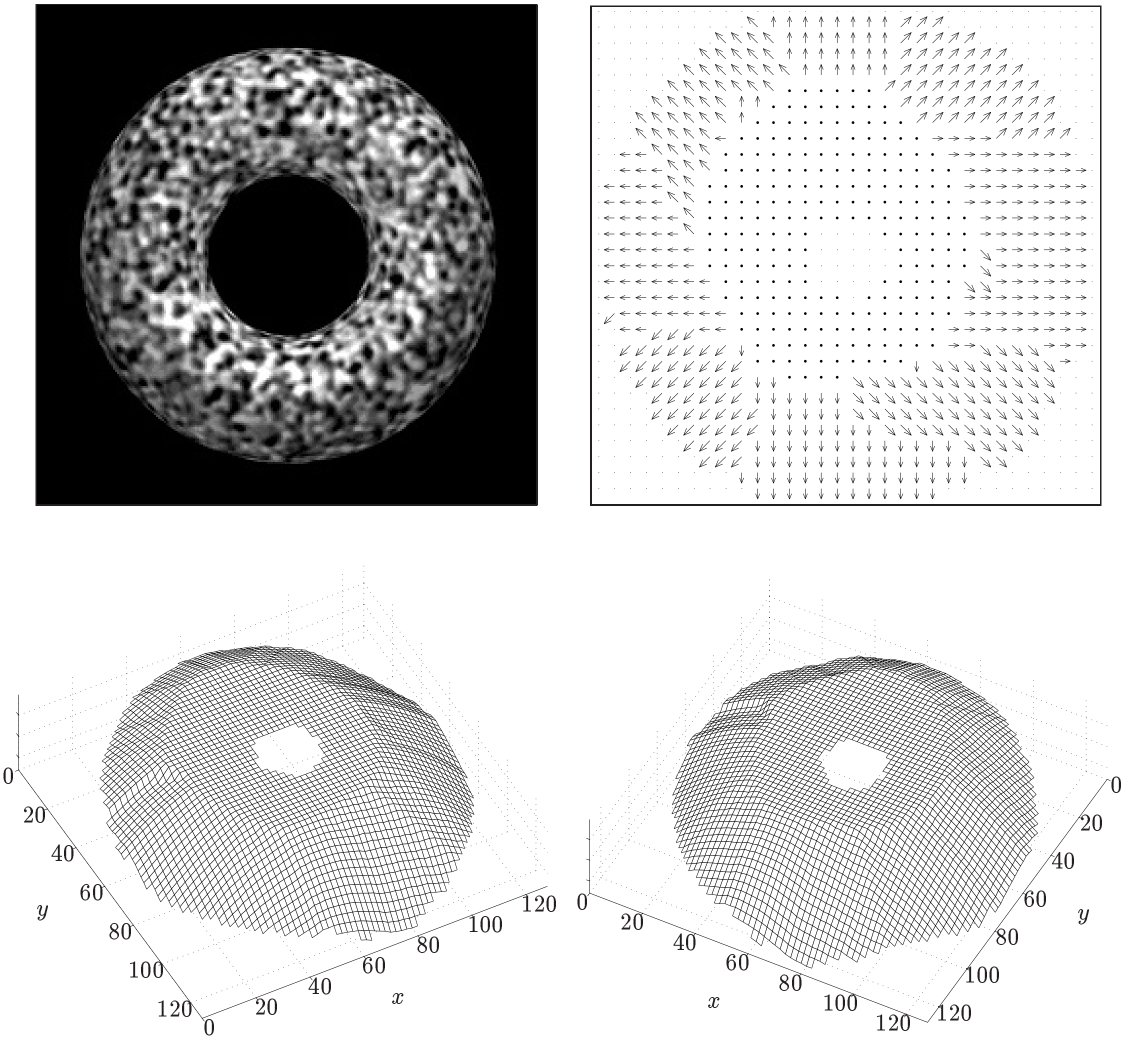


Figure 4.23: Shape estimation results for test image J.

center. These peaks are indicated by $P1$ and $P2$ in Figure 4.24. In-between the peaks is an area of lower values. Qualitatively, this is the pattern of values expected for a surface of this type. Each peak represents higher APF values and thus represents a region of the surface that is farther from the viewer than the area in-between; this corresponds to the inner and outer slopes of the toroidal surface.

Unfortunately, the difference in magnitude between the two peaks $P1$ and $P2$ and the lower area between them is small, and the separation between the two peaks is narrow. This makes the peaks difficult to detect. It is possible that the various averaging operations performed by the model render the two peaks indistinguishable at the current parameter settings. In addition, the two peaks are bordered on each side by a gradual decline towards lower APF values. This is unexpected because the borders of the surface

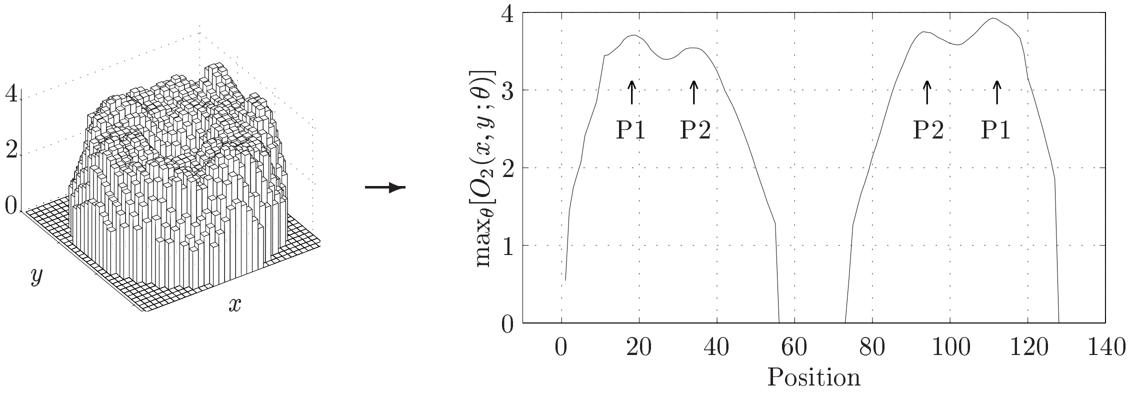


Figure 4.24: Cross-section through the landscape of $\max_\theta[O_2(x, y\,;\theta)]$ values for the image of the textured torus. The plot on the right shows the average of the values of the two cross-sections through the middle of the landscape along the x- and y-axes. $P1$ indicates the peak through the APF values closest to the outside surface of the torus, and $P2$ indicates the peak closest to the inside surface. In-between the peaks is an area of lower unnormalized APF values. The separation between the peaks is small, and the change in values between the peaks and the lull between them is also small. This means that the inner and outer sides of the surface are difficult to differentiate in the APF landscape. An additional problem is the gradual slope away from $P1$ and $P2$ on either side. This indicates that the APF values tend towards lower frequencies.

next to the blank regions are actually areas of highly compressed texture; these should therefore show up as higher APF values without a gradual decline to lower values. The gradual decline is due to the same problem discussed for previous input images: complex cells tuned to low spatial frequencies produced responses relatively far from the true borders of the textured surface. In the previous test cases, the shape estimation model was able to ignore these undesirable low APF values, but it is possible that in this case the model was led stray in the center of the torus.

Because it is possible to produce results of vastly different quality by adjusting the parameters in the model, I decided to investigate whether another set of parameter values might not produce better output for this test case. I performed a systematic search through a region of the parameter space summarized in Table 4.3 on the following page. Altogether, I examined over $1,000$ combinations of values, and succeeded in finding a region of the parameter space that yielded satisfactory results on test image J. Table 4.3 summarizes the values which gave the best subjective results. The output produced by the model using these parameter values is shown in Figure 4.25 on the next page.

The results in Figure 4.25 show substantial improvement over the first attempt. This time, both the inner and outer portions of the torus are sloped in rough agreement with human perception. The tilt direction map still displays some inaccuracies, especially in the

Table 4.3: Alternate parameter values explored for test image J

Parameter	Range Tested	Original Value	Alternate Value
r_i	5 – 22	22	14
r_s	5 – 22	22	14
l_p	24 – 32	30	24
c_1	1.2 – 3.5	3.0	1.5
c_2	-0.5 – 0.1	-0.25	-0.15
c_l	1.5 – 3.0	2.25	2.54

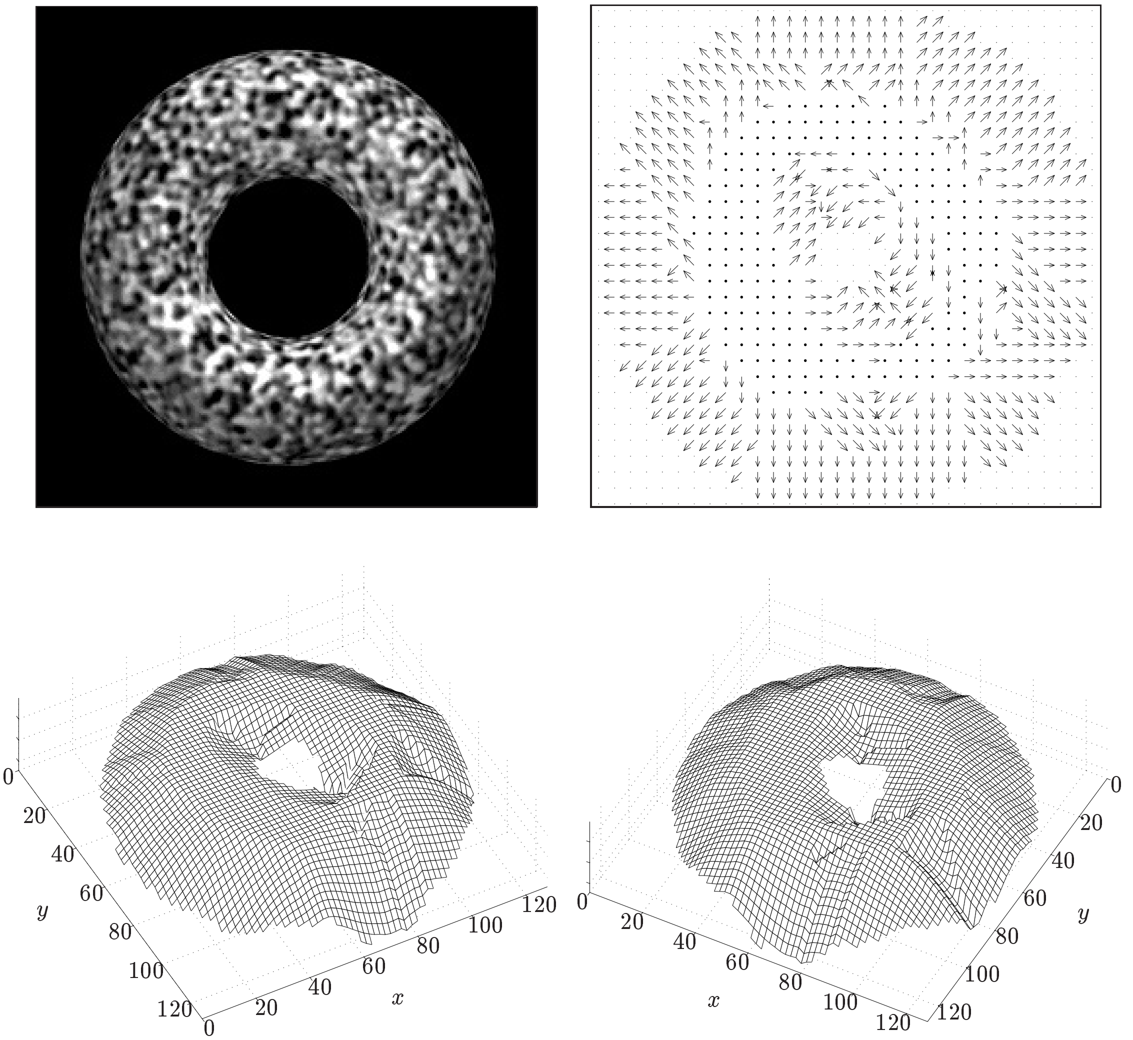


Figure 4.25: Shape estimation results for test image J using the alternate parameter values shown in Table 4.3.

middle area of the torus, and the reconstructed shape is admittedly rough. Nevertheless, the overall results are a fair approximation to the shape of the surface in the image, and they are significantly more correct than the previous results of Figure 4.23.

Does this mean that the original parameter values shown in Table 4.1 on page 157 were in fact unsuitable, even for the other test cases? To answer this question, I retested the other input images using the alternative parameter values shown in the right column of Table 4.3 on the page before. Although the model produced acceptable results for most of the inputs, the results were significantly rougher than those obtained using the original parameter settings. The differences were a matter of degree; none of the shape estimates using the alternative parameters were wildly inaccurate. However, it was clear that the original parameter values are more suitable for those other test cases.

These findings imply that the parameter values used for the other test cases may not be ideal for all types of inputs. It is especially noteworthy that the other cases all contain large contiguous textured surfaces. By contrast, the torus in the present case has a relatively narrow surface bounded by untextured areas on all sides (because of the hole in the middle). It is not surprising, then, that the parameter values that proved to be more suitable for this case (Table 4.3 on the preceding page) were generally lower in magnitude than the original parameter values. For example, it was necessary to decrease the averaging radii r_i and r_s. This is consistent with the fact that the surface width is much smaller. Larger averaging may cause the model to smooth over the subtle differences in the APF peaks shown in Figure 4.24 on page 177.

Test Image K

The results for the last test case are shown in Figure 4.26 on the following page. The image is a photograph of a roll of string.

Surprisingly, the model failed to produce a reasonable shape estimate for this input. The tilt direction map in Figure 4.26 reveals why: it shows that large portions of the outer region of the surface have been interpreted as being areas containing the lowest spatial frequencies. But this is contrary to expectations; the borders are in fact farther away from the viewer than the middle of the surface, and therefore should be marked by higher average spatial frequencies. This clearly confused both the interpretation of tilt directions and the integration process. Why did this happen?

This test case provides an example of some of the conditions that must be satisfied by an input image for successful application of the shape estimation model developed in this chapter. A close inspection of the texture on the surface in the image reveals that it exhibits two problems:

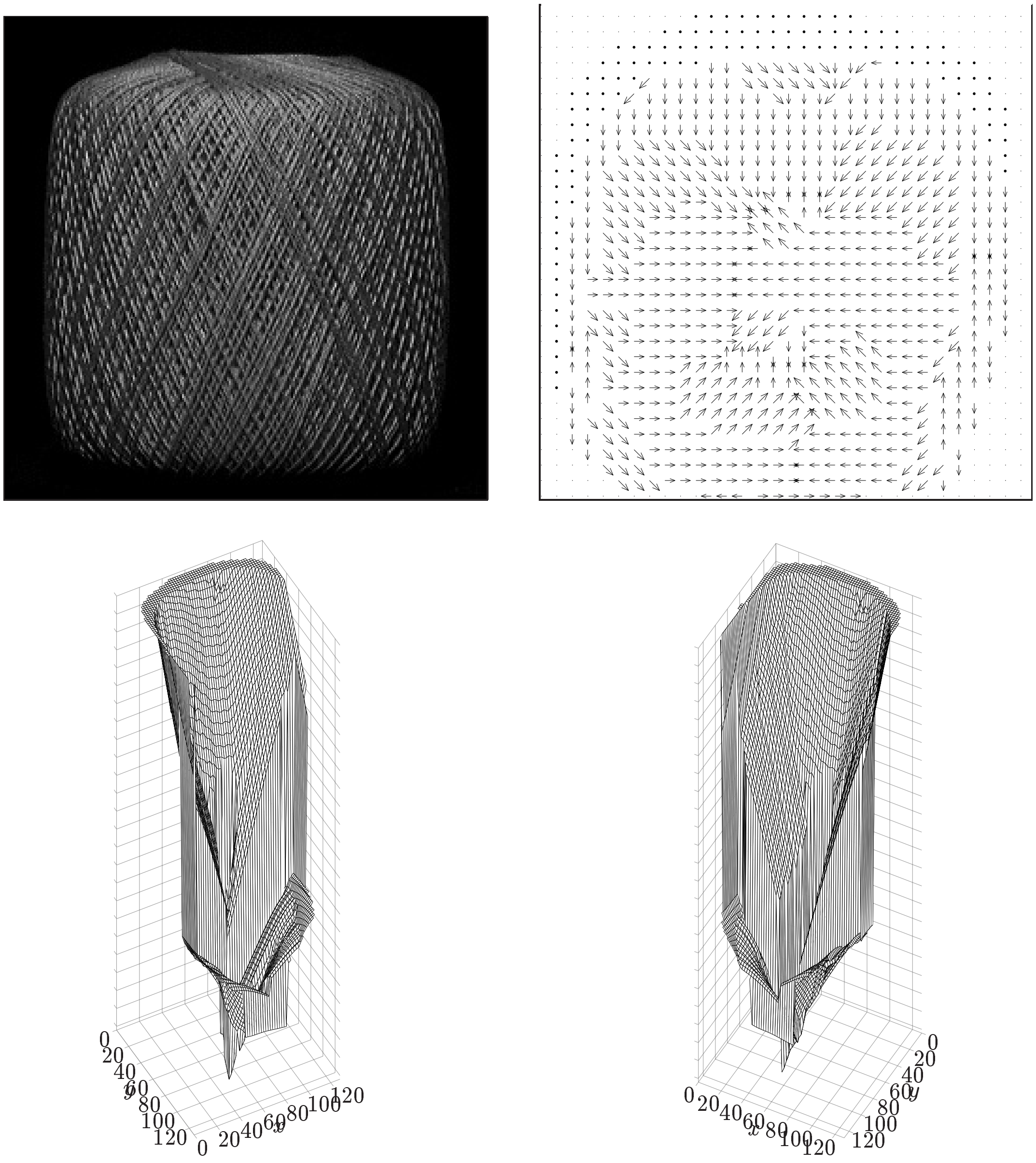


Figure 4.26: Shape estimation results for test image K.

- The texture is fine-grained almost everywhere. It is only slightly larger-grained in the middle of the surface than towards the outside. This results in a too-small amount of change in spatial frequencies when moving from central areas to peripheral areas on the surface. In fact, along the left and right sides of the cylinder, the sizes of the texture elements (which appear as short, light-colored line segments) remain nearly constant. Even the spacing between the texture elements changes little up to the edges of the surface in the image.

- The texture is poorly visible on the upper portion of the roll of string. The surface there appears nearly smooth. A smooth surface induces responses in complex cells tuned to low spatial frequencies, which results in low APF values in those areas. This in turn is interpreted by the shape estimation process as indicating that a region is closer to the viewer. In the image, the upper portion of the surface is in fact receding from view. But the lack of visible texture on portions of the surface in the image makes this impossible to tell based on texture alone.

To illustrate the effects of these factors, Figure 4.27 on the next page shows a contour plot of the maximum unnormalized APF values for this input. (This is the same kind of plot used above in the discussion of test image I.) The contour plot illustrates the pattern of APF values for this test case. It shows that the regions of the surface that are most slanted did not lead to high APF values. This makes it impossible for the shape estimation process to perform properly.

It is interesting to ask why humans do not have difficulty perceiving the shape of the surface pictured in this input image. The probable reason is that the input image offers a number of non-texture cues to its shape:

- *Contour cues*: the surface is composed entirely of line elements, and these line elements produce curves that can be interpreted to give shape information.

- *Shading cues*: the surface has shading effects, and these can be used to infer shape.

- *Object recognition*: humans easily recognize the object whose surface is pictured in the input image. This allows our visual systems to go beyond the information given in the input, and fill in whatever aspects of the shape may not be directly visible.

Thus, although the shape estimation model failed to give a satisfactory result for this test case, it may be impossible to make a shape interpretation from this input based on texture alone.

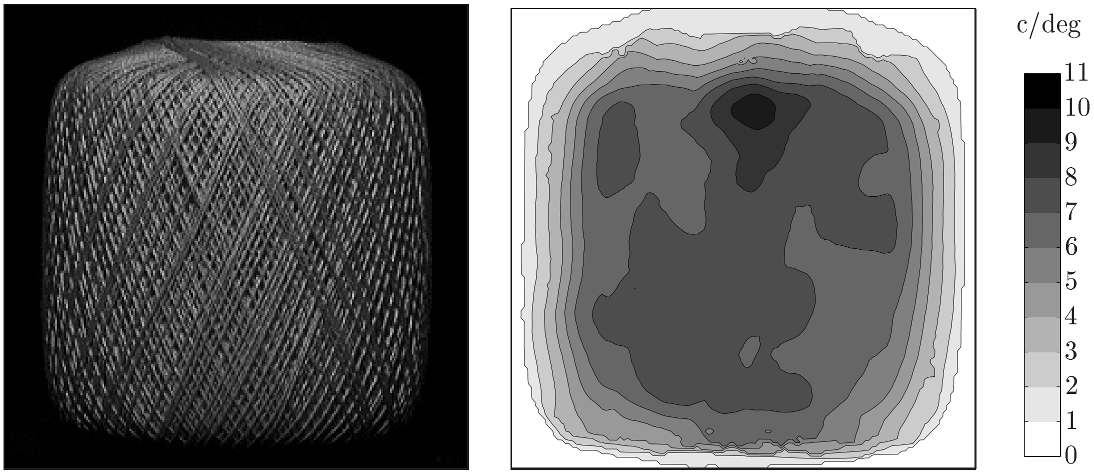


Figure 4.27: (Left) test image K from Figure 4.12, used in the results shown in Figure 4.26 on page 180. (Right) Contour plot of the the maximum unnormalized APF values across all eight orientations in the image. The values shown here were computed using the formula $\max_\theta[O_2(x, y\,; \theta)]$, where $O_2(x, y\,; \theta)$ represents the results from Stage Two of the model as described in Section 4.3.2. The colorbar to the right of the plot indicates the shades of gray corresponding to different average peak frequencies indicated by the contours of the plot. The contour plot shows that, contrary to expectations, the highest spatial frequencies appear near the middle of the surface rather than towards its edges.

4.4.3 Discussion of Experiment Two

Overall, the results of this experiment demonstrate the viability of this model of texture-based shape estimation. The model can produce perceptually plausible representations of the general shape of textured surfaces. Some of the test cases used in this experiment resulted in rougher shape estimates than others, but with the exception of the last case, all the outputs of the simulation displayed approximately the correct overall surface shapes.

Conditions for Best Results

The model described in this chapter can work with a variety of surface shapes and textures, subject to certain conditions about the nature of the input. These requirements have been alluded to elsewhere in this chapter, but it is worth discussing them in greater detail in the context of the experimental results. They can be summarized as follows:

1. Homogeneity of texture;

2. Visibility of texture;

3. Sufficient surface width; and

4. Absence of reversals in the tilt direction.

The first condition, *homogeneity of texture*, is one of the core assumptions behind Sakai and Finkel's (1995) model as it is implemented here. In order for the method to work, the texture on a surface must be everywhere the same, or close to the same. The less homogeneous a surface texture is, the less likely that the shape estimation method will interpret the spatial-frequency spectrum in a way that reflects the actual surface shape. For this reason, all of the textures used in this experiment have generally been highly homogeneous.

Several of the mechanisms and associated parameters in the model provide some degree of immunity to inhomogeneities in textures. In particular, large values of the smoothing parameters r_a, r_i, and r_s help increase tolerance to variations in the surface texture. The tradeoff with increased averaging is a decrease in the sensitivity to surface shape variations, and a tendency to flatten shapes.

The second condition, *visibility of texture*, relates to a problem illustrated in the results for test image K in Figure 4.26 on page 180. It is vital for the shape estimation model that the texture on a surface be visible everywhere on the surface, especially near borders facing away from the viewpoint. If the texture is not sufficiently visible, the surface will appear smooth in the image, which will result in complex cells tuned only to the lowest spatial frequencies to be activated. This is especially problematic when a surface is slanted away from the viewer, a situation that should result in a high degree of texture compression and consequently the activation of complex cells tuned to high spatial frequencies. If instead the texture becomes invisible and the surface appears smooth, the model will interpret that area of the surface as being closer rather than farther away, contradicting the true surface shape.

The third condition, *sufficient surface width*, concerns the need for the radii of surface regions in the image to be at least as wide as the various smoothing radii used in the model (that is, parameters r_a, r_i and r_s). If a surface or a portion of a surface is too small, the variations in spatial frequencies across the surface may be so close together that the smoothing processes in the model average over them. This can cause the variations in spatial frequencies to be obliterated and the surface to appear flat. Figure 4.28 on the next page illustrates these issues for a cone-shaped surface. The tip of the cone is narrow and after averaging ends up being interpreted as flat and facing the viewer.

The fourth condition, *absence of reversals in the tilt direction*, concerns the problem encountered with test image C in the results shown in Figure 4.15 on page 165. In that input image, the surface in the upper right-hand corner is reversed in slope; this led to the tilt direction being interpreted as facing inward. This in turn caused further problems for the shape estimation method: the tilt directions facing each other caused a sharp

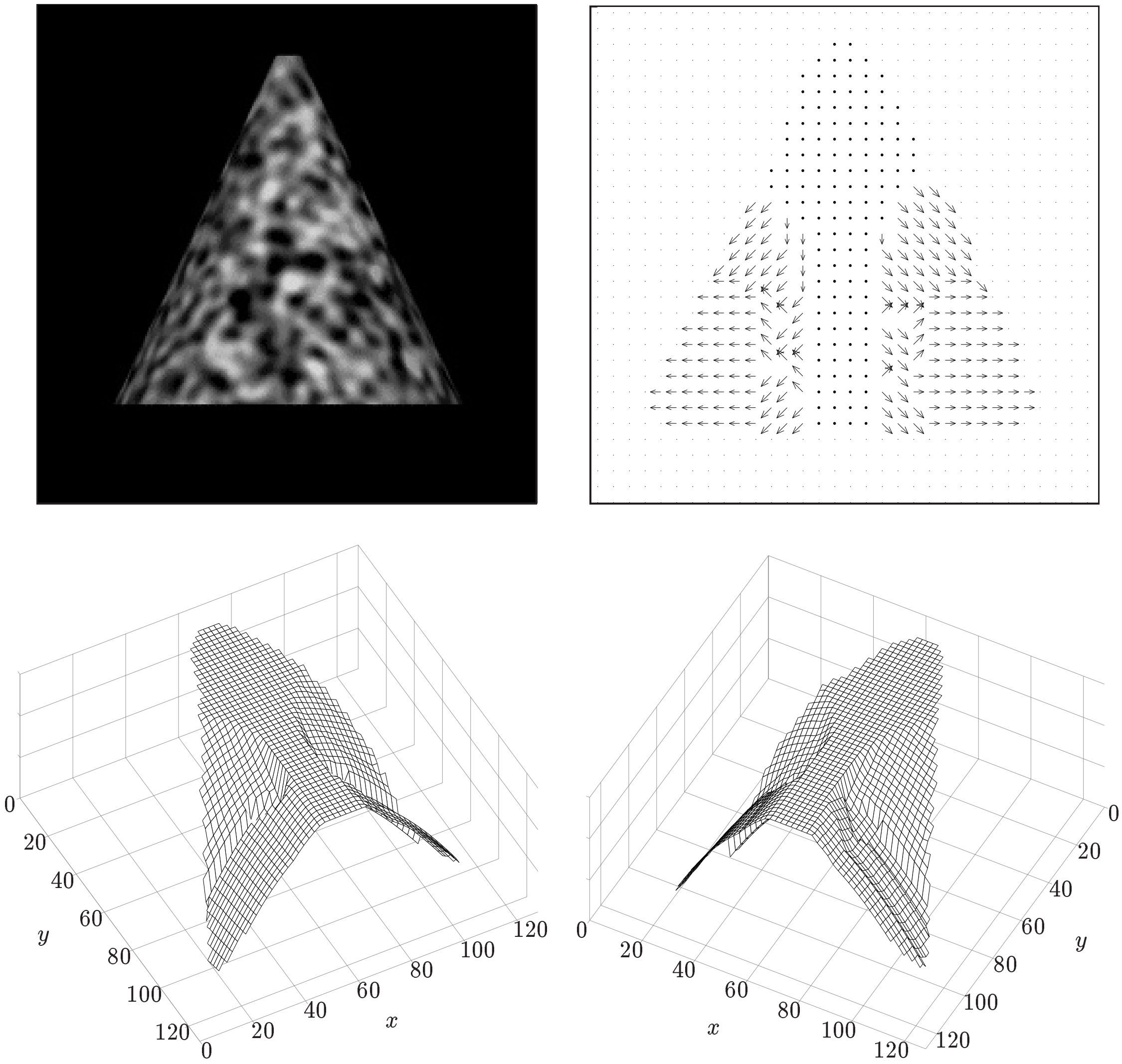


Figure 4.28: Results of shape estimation for an image of a conical surface covered with a noise texture. Note that the point of the cone is included in the area interpreted to be closest and facing the viewer.

discontinuity in the estimated surface shape. The model described in this chapter has no way of coping with this situation; it must assume that a separate segmentation component will divide an image region at locations where the surface gradient reverses itself.

Problem Areas

The shape estimation model works well for producing an approximate analysis of the overall shape of a surface. Still, there remain a few problem areas that can stand further improvement. These are:

1. *Avoiding an excessively flat region in the shape rendition corresponding to where the physical surface is interpreted as being closest to the viewer.*

   As discussed above, the flat areas in the surface renditions arise because the integration method does not compute depths in the surface region interpreted as being closest to the viewer. That region is fixed and serves as an "anchor" for the slant integration process. Although the model requires some way of anchoring the integration process, there may be a better way of establishing the region of lowest APF without introducing excessively large flat regions.

2. *Finding the best parameter settings for different input cases.*

   The case of the torus-shaped surface (test image J) demonstrates that parameter values developed based on large continuous surfaces may not be the most suitable for surfaces having small or medium widths. At minimum, this indicates that when multiple regions are dealt with in Chapter 6, the parameter values may need to be adjusted again. It would be good to find a method that can automatically scale the parameters based on some aspect of the input, such as perhaps the number of active complex cells.

3. *Overcoming the spill-over of complex cell responses at surface region borders.*

   All of the images containing single surfaces bounded by untextured areas produced complex cell responses that extended past the borders between the textured and untextured areas. This is undesirable and can lead to errors in the shape estimation process. An important area of further research is to find a way of modifying or post-processing the complex cell responses so that they are better-attuned to the true extent of textured surfaces.

4.5 Summary

In this chapter, I have presented a model for the texture-based estimation of surface shape. The model begins with complex cell responses as a way of estimating the localized spatial-frequency content across an input. The shape estimation principle is based on interpreting the pattern of changes in the spatial-frequency spectrum in order to infer changes in the shape of the physical surface. The model and much of the implementation is based on work by Sakai and Finkel (1995).

Including the initial complex cell component, there are four stages in Sakai and Finkel's model and the implementation presented in this chapter:

1. Computation of the responses of simulated complex cells at each point in an input image. Because cortical cells are selectively responsive to limited ranges of spatial frequencies, their responses can be used to estimate the spatial-frequency content at different locations in an input.

2. Computation of the average peak spatial frequency at each point in the input. This involves finding the strongest-responding complex cell at each location, tagging the location with the cell's preferred spatial frequency, and then averaging the frequency values in a spatial neighborhood surrounding each location.

3. Normalization of the average peak frequency followed by lateral inhibition between values in orthogonal orientations. The output of this stage is a distributed representation of the estimated local slant and tilt at each location in the input.

4. Integration of surface slant along the surface gradients. This combines the local estimates of slant and tilt into an estimate of the overall surface shape.

In the process of creating a new implementation of Sakai and Finkel's (1995) model starting with the complex cell simulation described in Chapter 3, I developed several modifications intended to improve the performance of the model. Briefly, the enhancements are:

1. The use of a method of averaging, described in Section 4.3.2, that discounts zero values. It has proven to be essential to use this method in the place of more common averaging methods in all calculations that call for an averaging step (e.g., Stages Two and Four of the implementation). This modification should be retained regardless of whether an integration process is to combine the distributed estimates of slant and tilt into an overall shape rendition.

2. The introduction of several additional mechanisms into Stage Four. These include averaging steps to help smooth over some of the roughness of the underlying complex cell responses; a step to fill in zeros in the maps of responses prior to integration; and several other mechanisms, such as the use of an inverted Gaussian weighting scheme described by Equations 4.27 and 4.28 on page 152. Individually, the effects of these changes are small, but collectively they significantly improve the performance of the integration stage of the model. However, these changes are less important to the underlying theory of shape estimation, because as discussed in Section 4.2.3, the integration process itself may not need to be performed in a biological visual system.

The simulation described in this chapter can take images of textured surfaces as inputs and produce representations of their approximate shapes as output. The input images must satisfy certain assumptions made in the model:

- The image projection must be orthographic;
- The input must contain a single, continuous surface that is convex;
- The texture on the physical surface must be homogeneous; and
- The portion of the surface having the lowest APF must be the portion of the surface closest to and facing the viewer.

Within these limitations, the results of Experiment Two in Section 4.4 on page 155 demonstrate that the model works fairly well for a variety of textures and surfaces. Nearly all test cases produced outputs that agreed qualitatively with the interpretations that a human might make of the input images.

Experiment Two also revealed some problem areas that would benefit from further research. First, many of the shape estimate results suffered from a pronounced area of flatness corresponding to the region interpreted as being closest to and facing the viewer. Second, some of the parameters in the model, especially c_l, do not work optimally at a single value for all cases; it would be better if they could be rescaled automatically based on some characteristic of the input. And finally, inputs containing transitions between textured and untextured areas demonstrated that the complex cell responses tended to spill over into the untextured region, an undesirable effect.

The next step in this research is to explore the use of the simulated complex cells as a basis for texture-based segmentation. That is the topic of the following chapter. In Chapter 6, the shape estimation component developed in the present chapter is combined with the segmentation component to produce the final system in this dissertation.

Chapter 5

A Model of Texture-Based Segmentation

Two main components of the overall model have been presented so far in this dissertation: an initial processing network composed of visual filters based on cortical complex cells, and a shape estimation mechanism that uses the model cell responses to estimate surface shape from texture. The third and last major component left to be developed is a texture-based segmentation mechanism.

As mentioned in Chapter 1, the model of texture-based segmentation pursued here casts segmentation as a process of settling a network of units into a stable configuration. For simulation purposes, this is represented as a mathematical function that expresses how the processes of region interpolation and boundary detection compete during the segregation of an input image into regions. Segmentation is achieved after the function—the sum of the competing forces—is iteratively settled to a minimum value.

The formulation that is developed in this chapter is based on the *coupled-membrane model* (Lee, 1993, 1995; Lee et al., 1991, 1992). This model does not assume that texture regions are defined by spatially-homogeneous filter responses; it can therefore segment visual textures that arise from slanted or curved surfaces. I show in this chapter that the coupled-membrane model can also be modified so that, rather than using complex cell responses directly, it begins with an intermediate result produced by the shape estimation component described in Chapter 4. This reduces the computational complexity of the segmentation process, and also forges a link between the segmentation and shape estimation processes.

This chapter is organized as follows. Section 5.1 presents an overview of the weak membrane model. In Section 5.2, I discuss the coupled-membrane model and offer a new, modified version of this segmentation approach. Section 5.3 fills out the practical details of implementing and using the model. Section 5.4 then presents Experiment Three, in which I tested the behavior and performance of the segmentation system. Finally, Section 5.5 summarizes the main points of this chapter.

5.1 Overview of the Weak Membrane Model

The goal of segmentation in the most general sense is to localize regions and their boundaries in a set of data. The data may be the two-dimensional array of pixel intensity values of an image, or they may be values from some other domain; for example, the data may be measurements derived from an image by using the responses of spatial filters resembling cortical cells.

The segmentation approach described in this chapter is a variation on a class of segmentation approaches known as *weak membrane models* (Blake & Zisserman, 1987; Geman & Geman, 1984; Mumford & Shah, 1985). The following analogy can provide some intuition into the basic operation of this segmentation model. It is illustrated in Figure 5.1 for the simple case of a two-dimensional set of data, $D(x, y)$. The data values can be imagined to form a landscape, where the height at every x, y location is determined by the value of $D(x, y)$ at that point. A thin, elastic sheet is then stretched over this landscape. In order to make the sheet conform to the hills and valleys, we can further imagine attaching small springs at the top of each bar in the landscape to the corresponding location on the elastic sheet. The springs can alternately push or pull as needed, but are constrained to only move up or down. The collective action of these many springs will

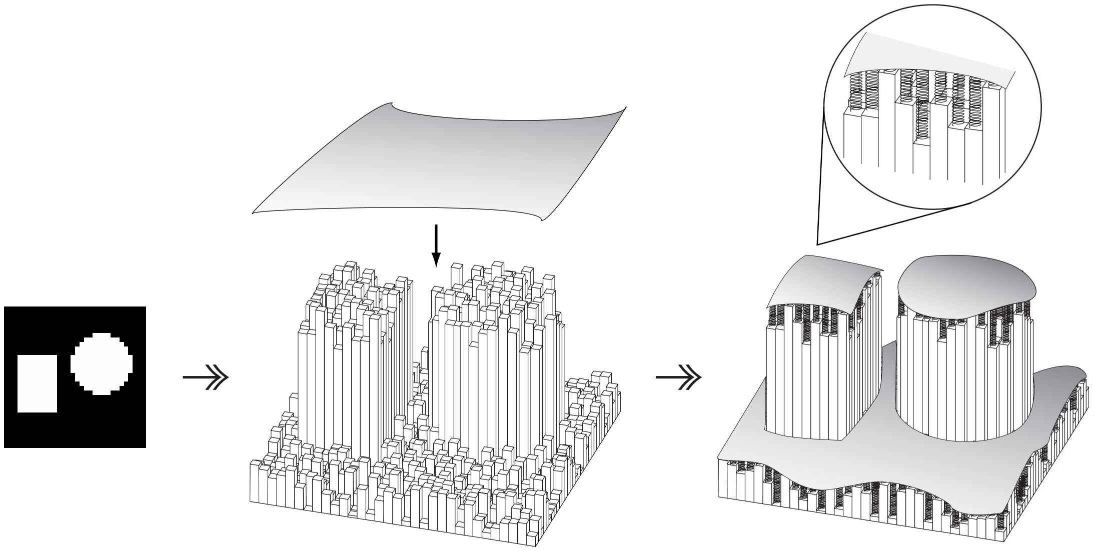


Figure 5.1: The basic analogy of the weak membrane. A two-dimensional array of data, in this case consisting of noisy light intensity readings, can be viewed as a landscape of values where the height of each bar represents the value $D(x, y)$ of the data at that point. A thin, elastic sheet is stretched over this data landscape, fixed to the top of each bar by a small spring that can push or pull but only move up and down. Where the data vary abruptly, the springs will pull and push on the sheet hard enough to break it. The set of breaks in the sheet indicates boundaries between regions in the data $D(x, y)$.

pull the sheet closer to the hills and valleys, making it conform to the landscape. In places where there are abrupt changes in the height of the landscape, the springs may pull on the sheet with such force that the sheet may break, forming boundaries between otherwise continuous areas. Between the elastic resilience of the sheet and the springs pulling on it, the sheet will tend towards a stable configuration. There may be more than one such stable configuration—like pressing on a mattress, giving the sheet a push may cause the different tensions to shift, changing the overall shape. But eventually, the system will settle into some compromise.

How can the best configuration of the system be predicted? One attractive approach is to simulate the behavior of the sheet using a rectangular grid of computing units. Each unit needs to communicate with only a few neighbors, and each can function by continually updating its output value based on some formula involving its current state and its neighbors' inputs. Using a network of this kind, the configuration of the elastic sheet can be found by having each unit compute the tensions between the sheet and the springs pulling on it. First each unit is initialized with a starting estimate of the sheet's height at that point. Then each unit adjusts its own, local estimate by replacing its output value with the average of the values of its four neighbors. This is repeated continually. Eventually the network as a whole settles into a state where the individual values stop changing; the result is a stable state of the system. It is easy to imagine this approach as a parallel process involving a large number of simple computational units such as neurons.

There are several ways of expressing this idea in more specific terms for purposes of analysis and computer simulation. The methods include statistical formulations (Geman & Geman, 1984; Geman et al., 1990; Marroquin, 1984, 1985a, 1985b; Nielsen, 1997; Zhu & Yuille, 1996), mean field theory (Bilbro, Snyder, Garnier, & Gault, 1992; Geiger & Girosi, 1991; Geiger & Yuille, 1991; Hofmann, Puzicha, & Buhmann, 1997), and energy functions (Blake & Zisserman, 1986a, 1986b, 1987; Lee, 1993, 1995; Lee et al., 1991, 1992; Morel & Solimini, 1995; Mumford & Shah, 1985; Schnörr & Sprengel, 1994; Shah, 1990, 1991, 1992a, 1992b, 1996a, 1996b). The methods can be implemented not only in software, but in analog VLSI hardware as well (Harris, Koch, & Luo, 1989; Harris, Koch, Staats, & Luo, 1990; Harris, Koch, & Luo, 1990; Koch, Marroquin, & Yuille, 1986; Lumsdaine, J. L. Wyatt, & Elfadel, 1991).

In the present research, I use the energy function formulation and mainly follow the treatment given by Blake and Zisserman (1987). The basic idea behind the energy function formulation is to cast the segmentation problem as one of minimizing a function. The function represents the state of the system as "energy", in analogy to the elastic energy in a mechanical system under tension. (This use of the term "energy" is different from its use in Chapter 3.) The function encodes two aspects of the problem: interactions between the region boundaries and the data, and constraints on the characteristics of the segmentation results. Minimizing the function is designed to lead to a best-fit labeling of the boundaries between regions, as well as a smoothed representation of the regions themselves.

In more formal terms, the goal of the weak membrane segmentation process is to find a piecewise-smooth function $U(x, y)$ that is a good fit to the data $D(x, y)$. The function $U(x, y)$ represents the elastic sheet in the analogy given above. The behavior of the sheet is defined by a function E having three main terms,

$$E = E_d + E_s + E_p. \tag{5.1}$$

E represents the elastic energy of the membrane. The process of refining the function $U(x, y)$ to fit $D(x, y)$ is performed by searching for values of $U(x, y)$ that will settle the membrane into a preferred state, a configuration where its elastic energy (the value of E) is at a minimum. The terms in the equation have the following meanings:

E_d: The *data term*, a measure of the faithfulness of the function $U(x, y)$ to the data $D(x, y)$. The general form of this component is

$$E_d = \iint [U(x, y) - D(x, y)]^2 dxdy. \tag{5.2}$$

This is a measure of the squares of the errors between the function $U(x, y)$ and the data $D(x, y)$. The integral is evaluated over the area where the data function $D(x, y)$ is defined.

E_s: The *smoothness term*, a measure of how severely the function $U(x, y)$ is deformed. The general form of this component is

$$E_s = \lambda^2 \iint [\nabla U(x, y) \cdot \nabla U(x, y)] \, dxdy. \tag{5.3}$$

This is a measure of the rate of change of $U(x, y)$ in the x and y directions. The integral is again evaluated over the area where the data are defined. The constant λ is a positive real number that controls the weight given to this term; E_s effectively tries to keep $U(x, y)$ as flat as possible, and λ controls the elasticity or willingness to deform of $U(x, y)$.

E_p: The *penalty term*, the sum of penalties levied for each break in the function $U(x, y)$. The general form of this component is $E_p = \alpha L$, where the function L is a measure of the set of contours created by discontinuities in $U(x, y)$. Several different definitions are possible; the simplest is to make L be a measure of the total contour length (Blake & Zisserman, 1987). Minimizing the function E then also acts to minimize the contour length. This form of L leads to

$$E_p = \alpha \int dl, \tag{5.4}$$

where l is the length of a discontinuity, and the integral is evaluated over the length of all discontinuities. The cost of creating a break in $U(x, y)$ is thus α per unit length of discontinuity, where α is a positive real number. The parameter α therefore controls how easily segmentation boundaries may be introduced in the final result.

Putting this all together yields the general, continuous version of the basic weak membrane equation,

$$E = \iint_A [U(x,y) - D(x,y)]^2 dxdy + \lambda^2 \iint_{A-B} [\nabla U(x,y) \cdot \nabla U(x,y)]\, dxdy + \alpha \int_B dl, \tag{5.5}$$

where A is the domain of the data and B is the set of piecewise contours that divide up the domain into separate regions. The influences of the parameters α and λ are discussed in more detail in Section 5.3.3.

Once the function E is minimized, $U(x, y)$ represents an approximation to the data $D(x, y)$. It is a piece-wise smoothed representation of $D(x, y)$, with discontinuities introduced between regions having borders with sharp transitions in the data. The discontinuities—the boundaries between smoothed regions—can be extracted from the final l values, which act as labels. Going back to the analogy of stretching an elastic sheet over a landscape of data, the smoothed result embodied by $U(x, y)$ represents the relatively smooth regions of the membrane, bounded by breaks in the membrane labeled by l.

These are the essential aspects of the weak membrane segmentation model. A few more elements are needed in order to use this in practice: the continuous formulation must be translated into a discrete form for implementation on a digital computer, and a method must be found for minimizing the discrete form of E. These issues are covered in Section 5.3.

5.2 The Coupled-Membrane Model and Its Modification

The basic weak membrane model is designed for two-dimensional data, such as intensity values in an ordinary gray-scale image. One approach to using this technique for segmenting images containing texture is the following: first compute a two-dimensional array of texture descriptors derived from the image intensity values, and then perform weak membrane segmentation on it. Although this is viable, it is difficult to do in practice because it requires devising a descriptor summarizing texture characteristics as a single, point-wise measure. The discussions in Section 2.1 make it clear that texture is inherently a multiscale and ill-described phenomenon difficult to characterize in any single way.

A more flexible approach is to extend the weak membrane model to include more than two dimensions. This is exactly what Lee (1993, 1995; Lee et al., 1991, 1992) did. He expressed texture segmentation in probabilistic terms as an inference process designed to find surfaces and surface boundaries in the visual world. The goal of this segmentation approach is to accomplish two tasks simultaneously: grouping areas of similar texture together, and labeling boundaries between dissimilar areas of texture. Lee achieved this by extending the weak membrane model to the case of a four-dimensional data set, consisting of the responses of visual filters resembling cortical cells at different spatial scales and orientations. The model includes couplings between all four dimensions, and so Lee dubbed this new formulation the *coupled-membrane model*.

The initial processing component of Lee's (1993, 1995; Lee et al., 1991, 1992) system consists of a bank of spatial filters resembling cortical complex cells. The underlying receptive field operators of the cells in that model are modeled after the Gabor function (Daugman, 1985; Kulikowski et al., 1982; Marčelja, 1980; Watson, 1983). The model combines two full-squared linear receptive field operators in quadrature at each spatial location, and normalizes their outputs by the sum of receptive field outputs at all frequencies and orientations. This approach is similar to Heeger's (1991) contrast-normalization model of complex cells discussed in Chapter 3. Lee used cells tuned to three peak spatial frequencies and eight orientations, with fixed spatial-frequency bandwidths of one octave and fixed orientation bandwidths of 22.5°. This four-dimensional set of twenty-four response maps is used in Lee's model as the input data $D(x, y, f, \theta)$ into the coupled-membrane model of segmentation.

5.2.1 The Coupled-Membrane Model

Lee (1993, 1995; Lee et al., 1991, 1992) formulated the segmentation process as the problem of finding a piecewise-continuous function $U(x, y, f, \theta)$ that is a smooth estimation of

$D(x, y, f, \theta)$, with the segmentation energy function defined as

$$\begin{aligned} E = & \iiiint_{A \times P} \left[U(x, y, f, \theta) - D(x, y, f, \theta) \right]^2 d(\log f) d\theta dx dy \\ & + \iiiint_{(A-B) \times P} \left[\gamma_f^2 \left(\frac{\partial U}{\partial \log f} \right)^2 + \gamma_\theta^2 \left(\frac{\partial U}{\partial \theta} \right)^2 + \lambda^2 \left(\frac{\partial U}{\partial x} \right)^2 + \lambda^2 \left(\frac{\partial U}{\partial y} \right)^2 \right] d(\log f) d\theta dx dy \\ & + \alpha \int_B dl \end{aligned} \tag{5.6}$$

where A is the two-dimensional spatial domain; P is the two-dimensional spectral domain (spatial frequency and orientation); $B \subset A$ is the set of boundary contours that divide up A into regions; the parameters λ, γ_f, and γ_θ control smoothing across space, spatial frequency and orientation, respectively; and α is the penalty for creating discontinuities in $U(x, y, f, \theta)$. The integration over the frequency dimension is done with $d \log f$ because spatial frequency is represented in logarithmic form. The parameters γ_f and γ_θ are positive real numbers.

The equation above has the same form as the weak membrane function, $E = E_d + E_s + E_p$, but the second term introduces new couplings in spatial frequency and orientation. Under the control of the parameters γ_f and γ_θ, shifts in spatial frequency and orientation are permitted in the function $U(x, y, f, \theta)$ as it tries to conform to the complex cell responses of $D(x, y, f, \theta)$. This is what allows for texture gradients and distortions in the input: as long as the shifts in frequency and orientation are not too great, a given region of texture will be treated as a continuous region. The final result of the segmentation process is a set of labels that divide the spatial domain A of the image into a set of regions, $A_1, \dots, A_n$, which are the connected components of $A - B$.

Lee (1993, 1995; Lee et al., 1991, 1992) demonstrated this approach on images containing multiple regions of textures with gradients. The simulation produced reasonable results on a variety of images of natural scenes, capturing most boundaries between regions of texture in the inputs.

5.2.2 The Modified Coupled-Membrane Model of Segmentation

The coupled-membrane function in Equation 5.6 includes gradients in space, spatial frequency, and orientation. Suppose that instead of using the raw cortical filter responses as the data $D(x, y, f, \theta)$, the coupled-membrane function were modified to use the unnormalized average peak frequency (APF) values described in Chapter 4. The unnormalized APF values are computed using Equation 4.12 on page 137; this essentially collapses the complex cell responses across frequencies, producing one response map for each orientation. Each such map is organized spatially; that is, each point in a map corresponds to a spatial location in the input image.

The segmentation energy function for this *modified coupled-membrane model* can then be written as

$$\begin{aligned} E = & \iiint_{A\times\theta} \left[U(x,y,\theta) - D(x,y,\theta)\right]^2 d\theta dx dy \\ & + \iiint_{(A-B)\times\theta} \left[\gamma^2\left(\frac{\partial U}{\partial\theta}\right)^2 + \lambda^2\left(\frac{\partial U}{\partial x}\right)^2 + \lambda^2\left(\frac{\partial U}{\partial y}\right)^2\right] d\theta dx dy \\ & + \alpha \int_B dl. \end{aligned} \tag{5.7}$$

In this equation, $D(x,y,\theta)$ is the set of unnormalized APF maps (one per orientation θ); $U(x,y,\theta)$ is the piecewise-continuous function that is fitted to $D(x,y,\theta)$; A is the two-dimensional spatial domain; $B \subset A$ is the set of piecewise boundary contours that divide up A into regions; the parameters λ and γ control the smoothing of the result across space and orientation, respectively; and α is the penalty for creating discontinuities in $U(x,y,\theta)$. As before, α, γ and λ are positive real numbers.

A change in APF values from one spatial location to another can indicate a change in surface orientation, following the principles outlined in Chapter 4. *Normalized* APF values are directly proportional to the estimated local surface slant at a given point and particular orientation, because a change in normalized APF is equated with a change in texture compression across a surface. However, normalization requires having a region in which to normalize the APF values, and unfortunately, information about the extent of regions is unknown at the segmentation stage—the whole point of segmentation is to find the texture regions. The *unnormalized* APF values used in the equation above are therefore only indirectly related to surface shape. Nevertheless, the gradient terms in Equation 5.7 can be interpreted loosely in terms of changes in surface orientation because of their roles in tracking texture compression:

- The term $\partial U(x,y,\theta)/\partial x$ is related to the local, instantaneous rate of texture compression in the x-direction in the image;

- The term $\partial U(x,y,\theta)/\partial y$ is related to the local, instantaneous rate of texture compression in the y-direction in the image.

In other words, these two components together are related to (though not synonymous with) change in compression at a given point and in a given orientation θ. Change in compression in turn is proportional to local surface slant. The relationship is not a perfect one, but it does give some intuition about the role of these gradient terms in the segmentation process.

By using the APF values computed by the texture-based shape estimation component, the segmentation process can reuse an intermediate result that must be computed anyway, increasing overall efficiency. More importantly, the dimensionality of the modified coupled-membrane function is reduced by one, from four to three dimensions, leading to a substantial reduction in computation during the segmentation process. And finally, by doing this, the segmentation process is linked more closely to the shape estimation process, connecting with arguments by Krumm and Shafer (1994) that segmentation should make use of information about surface orientation.

Has anything been lost by collapsing one of the dimensions in the coupled-membrane model? Will the resulting segmentation system still perform well? These questions are examined in Experiment Three below, which presents segmentation results for both the original coupled-membrane model and the modified coupled-membrane model.

5.3 Implementation and Operation of the Modified Coupled-Membrane Model of Segmentation

The discussion thus far has focused on general aspects of membrane models of segmentation. This section covers several practical aspects of turning the continuous formulation of the modified coupled-membrane model (Equation 5.7 on the preceding page) into a working software implementation. Three main topics serve as the focus:

1. Translating the continuous form of the segmentation function into a discrete format suitable for software implementation;

2. Minimizing the resulting discrete segmentation function; and

3. Understanding the roles of the parameters in the modified coupled-membrane model.

5.3.1 The Discrete Formulation of the Modified Coupled-Membrane Model

The weak membrane model is simpler to understand than the coupled-membrane model, so here I discuss the discretization of the weak membrane function first, followed by the formulation for the modified coupled membrane. The results presented here are based in large part on the derivations provided by Blake and Zisserman (1987).

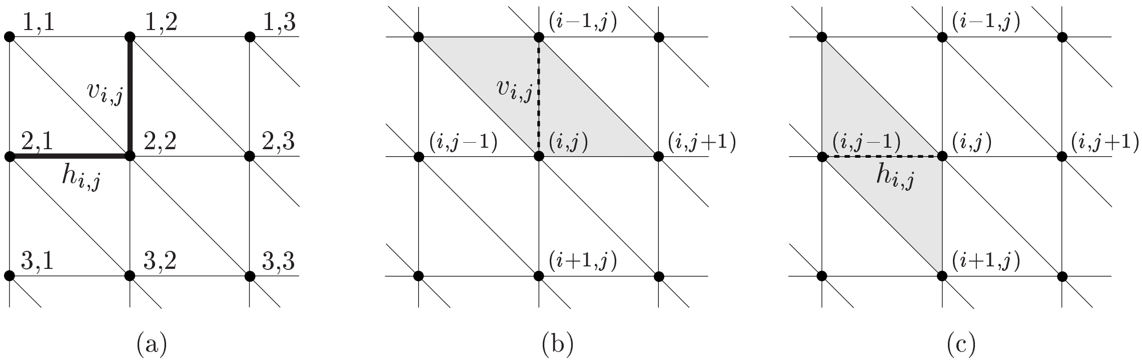


Figure 5.2: Arrangement of line variables in the two dimensional case. The convention I use here is that of Blake and Zisserman (1987), with a minor change to the ordering of the horizontal elements. (a) The array of $u_{i,j}$ and $d_{i,j}$ nodes is numbered using the matrix convention (i, j), in which the origin is the upper left. Boolean variables $h_{i,j}$ and $v_{i,j}$ are used to label breaks between nodes. (b) A value of $v_{i,j} = 1$ for some i, j indicates that a break exists between nodes at locations i, j and $i - 1, j$. This signifies that the membrane energy component represented by the two "northerly" triangles, shown shaded in the figure, is disabled. (c) A value of $h_{i,j} = 1$ for some i, j indicates that a break exists between nodes at locations i, j and $i, j - 1$. This signifies that the membrane energy component represented by the two "easterly" triangles, shown shaded in the figure, is disabled.

The Weak Membrane Model

The standard approach to discretizing the weak membrane function is to use the finite element method (Blake & Zisserman, 1987; Strang & Fix, 1973). In this approach, the continuous domain of x, y in Equation 5.5 on page 192 is divided into $n_x \times n_y$ points on a rectangular grid as illustrated in Figure 5.2. The discrete versions of function $U(x, y)$ and data $D(x, y)$ are then defined at each point or *node* of the grid. The nodes are indexed (using matrix numbering convention rather than x, y convention) such that $u_{i,j} = U(j, i)$ and $d_{i,j} = D(j, i)$, where i and j are indexing variables.

The discontinuity variable, l, in Equation 5.5 needs to be given a discrete form as well. In a one-dimensional case, this is fairly simple: a set of Boolean-valued variables $\{l_i\}$ is defined such that l_i (known as a *line variable*) indicates whether a discontinuity is present between two nodal values i and $i - 1$. This approach was first introduced by Geman and Geman (1984). In the two-dimensional case, it becomes slightly more involved because there are several possible arrangements. A true finite element approach would typically use triangular elements, with the function value constant over the triangle and the discontinuity variable labeling to the body of the triangle. However, Blake and Zisserman (1987) found that this arrangement is not conducive to implementing an efficient minimization method. Instead, Blake and Zisserman introduced an arrangement in which line variables serve to label breaks in two adjacent triangles. Their approach, with a slight change in numbering convention, is illustrated in Figure 5.2. It uses two sets of line variables, $v_{i,j}$ and $h_{i,j}$. The first set labels breaks between nodes in a vertical direction, and the second set labels breaks between nodes in a horizontal direction. The Boolean labeling variables are defined so that $v_{i,j} = 1$ indicates the two triangles shown in Figure 5.2(b) are broken, whereas $h_{i,j} = 1$ indicates the two triangles shown in Figure 5.2(c) are broken.

The discrete variables $u_{i,j}$, $d_{i,j}$, $h_{i,j}$ and $v_{i,j}$ are combined to create the discrete forms of the three components of Equation 5.5,

$$\begin{aligned} E = \sum_{i,j}(u_{i,j} - d_{i,j})^2 + \lambda^2 \sum_{i,j}\left[(u_{i,j} - u_{i+1,j})^2(1 - v_{i,j}) + (u_{i,j} - u_{i,j+1})^2(1 - h_{i,j})\right] \\ + \alpha \sum_{i,j}(v_{i,j} + h_{i,j}). \end{aligned} \tag{5.8}$$

The parameters λ and α are the same in Equation 5.7 on page 195, and the summations are taken over all i, j where the data are defined. The above is the discrete formulation of the segmentation energy function for the weak membrane model.

It is useful to note that the i, j indexing convention, which is common in work involving matrices and in image processing, is slightly at odds with the x, y convention used in other contexts. This is an unfortunate collision of conventions. In x, y notation, the first component serves to index elements in the horizontal direction, and the origin is in the lower left of an array. Conversely, in i, j notation, the first component serves to index elements in the vertical direction, and the origin is in the upper left of an array.

The Modified Coupled-Membrane Model

The discrete form of the modified coupled-membrane model is closely related to the weak membrane formulation. There are two differences. First, the data values in the modified model are not image intensities but rather unnormalized APF values computed using Equation 4.12 on page 137. Second, there is a third dimension representing orientation.

The addition of a third dimension leads to a space of data which can be visualized as a stack of two-dimensional layers. The values of $D(x, y, \theta)$ in a given layer (a particular orientation) constitute the unnormalized APF map for that orientation. This continuous domain can be discretized in an analogous way to the approach taken in the case of the weak membrane: the continuous domain is divided into $n_x \times n_y \times n_\theta$ nodes in a cuboid grid, where n_x, n_y and n_θ are the number of elements in the x, y, and orientation dimensions, respectively. The discrete values of $D(x, y, \theta)$ and $U(x, y, \theta)$ are defined at each node such that $u_{i,j,k} = U(x, y, \theta)$ and $d_{i,j,k} = D(x, y, \theta)$, where i, j, and k are indexing variables. The line variables $v_{i,j}$ and $h_{i,j}$ are confined to the two spatial dimensions.

The discrete form of the modified coupled-membrane function (Equation 5.7 on page 195) is given by

$$\begin{aligned}E = &\sum_{i,j,k}(u_{i,j,k} - d_{i,j,k})^2 \\ &+ \sum_{i,j,k}\big[\gamma^2(u_{i,j,k} - u_{i,j,k+1})^2 \\ &\qquad + \lambda^2(u_{i,j,k} - u_{i+1,j,k})^2(1 - v_{i,j}) + \lambda^2(u_{i,j,k} - u_{i,j+1,k})^2(1 - h_{i,j})\big] \\ &+ \alpha\sum_{i,j}(h_{i,j} + v_{i,j}),\end{aligned} \tag{5.9}$$

where i, j, k are integer index variables for y, x, and θ, respectively; the summations are performed over the full range of nodes defined in the three-dimensional set of data; and the parameters α, λ and γ are the same as in Equation 5.7 on page 195.

Within each two-dimensional layer specified by a given orientation θ, the coupled-membrane model above acts partly like a weak membrane trying to fit the landscape of APF values. All the weak membranes are coupled together through both the first and second terms of the right-hand side of Equation 5.9; they do not act independently but rather concurrently, taking into account data from all orientations. Parameter γ in the formula sets the degree of smoothing across orientations. Higher values of γ impose a higher cost on gradients in values across orientations.

The equation above is essentially identical in form to the discrete coupled-membrane model given by Lee (1993, 1995; Lee et al., 1991, 1992), with that model's spatial-frequency dimension removed and the $u_{i,j,k}$'s equal to the unnormalized APF values for an input (computed as described in Section 4.3.2).

5.3.2 Minimizing the Segmentation Energy Function

Segmentation using the modified coupled-membrane model, as well as the weak membrane and coupled-membrane models, requires that the discrete segmentation function E be minimized over the given data and parameter values. This turns out to be difficult in practice.

A function that has a single, global minimum value is known as *convex*. A convex function has a bowl-shaped landscape of values, with its lowest energy at a unique point in the bottom of the bowl. Finding the minimum of a convex function is straightforward: one simply starts anywhere in the bowl and follows the direction of the gradient of the values. Unfortunately, the weak membrane function and its relatives are nonconvex because of the weak continuity constraints imposed by the third term in Equation 5.1 (Blake, 1983; Blake & Zisserman, 1986b, 1987). This means that the segmentation energy functions may have many local minima—intermediate stable states into which the functions may fall but that do not represent the best, most stable state of the system.

Simple, direct minimization algorithms such as ordinary gradient descent cannot be used to find the minimum of E, because these are prone to getting trapped in local minima. Thankfully, alternative methods that overcome this problem are available. Simulated Annealing (Kirkpatrick, Jr., & Vecchi, 1983) is one minimization approach; it has been applied to a weak membrane type of model by Geman and Geman (1984), and to the coupled-membrane model by Lee (1993, 1995; Lee et al., 1991, 1992). Another approach is dynamic programming (noted by Blake & Zisserman, 1987). Nielsen (1997) has developed a method based on Gaussian convolution. The use of genetic algorithms for energy minimization has been explored by Bhandarkar and Zeng (1997). Yet another alternative is to use a Hopfield neural network (Hopfield, 1984; Hopfield & Tank, 1985), which has the advantage that it can be extended to an analog VLSI implementation (Koch et al., 1986).

Blake (1983; Blake & Zisserman, 1986b, 1987) developed a specialized, deterministic algorithm for minimizing the weak membrane function. His method is the *Graduated Nonconvexity* (GNC) algorithm. The method is more efficient for this problem than general-purpose approaches such as Simulated Annealing. Lee (1993, 1995; Lee et al., 1991, 1992) has adapted it for the coupled-membrane model, and it can easily be adapted to work for the modified coupled-membrane model as well. The GNC algorithm is the minimization method I use in this research.

Overview of the Graduated Nonconvexity Algorithm

The GNC algorithm proceeds as follows. For some nonconvex function F, the first step is to construct a convex approximation to the original function and find its minimum. Let F^* be this approximation. This function, being convex, has a unique minimum that can be found deterministically using any one of a number of direct methods.

The second step is to define a sequence of functions $F^{(p)}$ for $1 \geq p \geq 0$. Each successive function is allowed to become "less convex" and more closely approximate the original:

$$\begin{aligned}
&F^{(1)} = F^* &&\text{(i.e., the convex approximation)}\\
&F^{(1/2)} &&\\
&F^{(1/4)} &&\\
&\vdots &&\\
&F^{(0)} = F &&\text{(the original nonconvex function)}
\end{aligned}$$

The GNC algorithm then proceeds to minimize iteratively the sequence of functions $F^{(p)}$, namely $F^{(1)}, F^{(1/2)}, F^{(1/4)}, F^{(1/8)}, F^{(1/16)}, \ldots$, using the results of one minimization as the starting point for the next. Since the first function, $F^{(1)} = F^*$, is convex, any starting point can be used to accomplish its minimization. The successive approximations $F^{(p)}$ gradually approach the global minimum of F as $p \to 0$. The actual number of values of p iterated depends on other parameters as described below, but in practice, a total of five to seven iterations are common for the sizes of images used in this project.

How does one go about creating a convex approximation F^* and the sequence of approximations $F^{(p)}$? Blake (1983; Blake & Zisserman, 1986b, 1987) developed a technique suitable for functions having the form of the membrane models of segmentation. The first step in the approach involves rewriting the energy function. The formula for the energy of the modified coupled-membrane (as well as that for the weak membrane and the original coupled-membrane) is a function, $E(u_{i,j,k}, v_{i,j}, h_{i,j})$, of both the real-valued variables $u_{i,j,k}$ and the Boolean-valued line variables $v_{i,j}$ and $h_{i,j}$. In order to make the GNC algorithm work, the Boolean-valued variables must be removed to produce a one-parameter version of the energy function. Blake had the insight that the minimization with respect to $v_{i,j}$ and $h_{i,j}$ in a membrane-type function can be done in advance, yielding a new energy function $\hat{E}(u_{i,j,k})$ that depends only on the $u_{i,j,k}$'s. The GNC algorithm can be used on this new function. Once $\hat{E}(u_{i,j,k})$ is minimized, the line variables can be recovered easily from the result.

The derivation of this simplified energy function for the case of the modified coupled-membrane is described in the next section, and the actual details of the GNC algorithm are described in the section after that.

Preliminary Step: Simplifying the Segmentation Energy Function

Blake and Zisserman (1987) provide a derivation of the function $\hat{E}(u_{i,j})$ for the weak membrane function. The version for the modified coupled-membrane, $\hat{E}(u_{i,j,k})$, can be derived similarly to this as follows.

First, following Lee (1993), the summations over orientations θ in Equation 5.9 can be encapsulated in separate functions. Let

$$Z_v(i,j) = \sqrt{\sum_k (u_{i,j,k} - u_{i+1,j,k})^2}, \tag{5.10}$$

$$Z_h(i,j) = \sqrt{\sum_k (u_{i,j,k} - u_{i,j+1,k})^2}, \tag{5.11}$$

where the subscripts v and h signify differences taken in the vertical and horizontal directions, respectively, on the i,j grid. Then the energy function $E(u_{i,j,k}, v_{i,j}, h_{i,j})$ can be rewritten in the form

$$E(u_{i,j,k}, v_{i,j}, h_{i,j}) = E_d + \sum_{i,j} H[Z_v(i,j), v_{i,j}\,; \alpha, \lambda] + \sum_{i,j} H[Z_h(i,j), h_{i,j}\,; \alpha, \lambda], \tag{5.12}$$

where

$$H(t, l\,; \alpha, \lambda) = \lambda^2 t^2 (1-l) + \alpha l \tag{5.13}$$

and E_d encapsulates the data term and the difference term involving the orientation dimension,

$$E_d = \sum_{i,j,k} (u_{i,j,k} - d_{i,j,k})^2 + \sum_{i,j,k} \gamma^2 (u_{i,j,k} - u_{i,j,k+1})^2. \tag{5.14}$$

The effect of these manipulations is to rewrite $E(u_{i,j,k}, v_{i,j}, h_{i,j})$ in a form where the dependence on the line variables have been moved into the terms involving the function $H(t, l\,; \alpha, \lambda)$. The particular form of $H(t, l\,; \alpha, \lambda)$ is one used by Blake and Zisserman (1987) and Lee (1993, 1995; Lee et al., 1991, 1992), although the notation I use here is slightly different to avoid conflicts with other notations used in this dissertation.

The minimization problem is now

$$\begin{aligned} &\min_{\{u_{i,j,k}, v_{i,j}, h_{i,j}\}} \Big\{ E_d + \sum_{i,j} H[Z_v(i,j), v_{i,j}\,; \alpha, \lambda] + \sum_{i,j} H[Z_h(i,j), h_{i,j}\,; \alpha, \lambda] \Big\} \\ &= \min_{\{u_{i,j,k}\}} \Big\{ E_d + \min_{\{v_{i,j}, h_{i,j}\}} \sum_{i,j} \big(H[Z_v(i,j), v_{i,j}\,; \alpha, \lambda] + H[Z_h(i,j), h_{i,j}\,; \alpha, \lambda] \big) \Big\}. \end{aligned} \tag{5.15}$$

The inner minimization in the right-hand side of this equation can be performed first.

Doing so reduces the problem to one of finding $\min_{\{u_{i,j,k}\}} \hat{E}(u_{i,j,k})$, where

$$\hat{E}(u_{i,j,k}) = E_d + \sum_{i,j} \hat{H}[Z_v(i,j)\,;\alpha,\lambda] + \sum_{i,j} \hat{H}[Z_h(i,j)\,;\alpha,\lambda]. \tag{5.16}$$

The line variables $v_{i,j}$ and $h_{i,j}$ have been absorbed into the new $\hat{H}(t\,;\alpha,\lambda)$ terms, leaving an equation that depends only on $u_{i,j,k}$. The function $\hat{H}(t\,;\alpha,\lambda)$, which is a modified version of $H(t,l\,;\alpha,\lambda)$, acts as a neighbor interaction function between the nodes $u_{i,j,k}$. For the modified coupled-membrane model, it is defined as

$$\begin{aligned} \hat{H}[Z_v(i,j)\,;\alpha,\lambda] &= \min_{l\in\{0,1\}} H[Z_v(i,j), v_{i,j}\,;\alpha,\lambda] \\ &= \begin{cases} \lambda^2[Z_v(i,j)]^2 & \text{if } |Z_v(i,j)| < \sqrt{\alpha}/\lambda \\ \alpha & \text{otherwise.} \end{cases} \end{aligned} \tag{5.17}$$

$$\begin{aligned} \hat{H}[Z_h(i,j)\,;\alpha,\lambda] &= \min_{l\in\{0,1\}} H[Z_h(i,j), h_{i,j}\,;\alpha,\lambda] \\ &= \begin{cases} \lambda^2[Z_h(i,j)]^2 & \text{if } |Z_h(i,j)| < \sqrt{\alpha}/\lambda \\ \alpha & \text{otherwise.} \end{cases} \end{aligned} \tag{5.18}$$

The function $\hat{E}(u_{i,j,k})$ is the sought-after, reduced version of the original membrane energy function, $E(u_{i,j,k}, v_{i,j}, h_{i,j})$. In the GNC algorithm, it is subjected to minimization in place of the original function. Once this is accomplished, the line variables $v_{i,j}$ and $h_{i,j}$ can be recovered from the result using the formulas

$$v_{i,j} = \begin{cases} 1 & \text{if } |Z_v(i,j)| > \sqrt{\alpha}/\lambda, \\ 0 & \text{otherwise,} \end{cases} \tag{5.19}$$

$$h_{i,j} = \begin{cases} 1 & \text{if } |Z_h(i,j)| > \sqrt{\alpha}/\lambda, \\ 0 & \text{otherwise.} \end{cases} \tag{5.20}$$

The formulas above need to be applied to each grid point in the results of the minimization of $\hat{E}(u_{i,j,k})$. At those locations where $h_{i,j} = 1$ or $v_{i,j} = 1$, a break is labeled in the segmentation output. Thus, the segmentation result is in the form of a labeling of region borders in the input image.

The GNC Algorithm

The simplified version $\hat{E}(u_{i,j,k})$ of the energy function $E(u_{i,j,k}, v_{i,j}, h_{i,j})$ derived above is still nonconvex; it is merely a rewritten form of the latter function in which the line variables have been temporarily removed. As mentioned above, the first step in the GNC algorithm (Blake, 1983; Blake & Zisserman, 1986b, 1987) is to define a convex approximation to $\hat{E}(u_{i,j,k})$ and minimize it. The second step is to define a sequence of functions $\hat{E}^{(p)}(u_{i,j,k})$, each closer to the desired function (and thus increasingly nonconvex) and eventually ending with $\hat{E}(u_{i,j,k})$ itself, and minimize each of these functions in turn.

Blake and Zisserman (1986b, 1987) define the family of one-parameter functions $\hat{E}^{(p)}$ as follows (adapted here for the case of the modified coupled-membrane model):

$$\hat{E}^{(p)}(u_{i,j,k}) = E_d + \sum_{i,j} \hat{H}^{(p)}[Z_v(i,j)\,;\alpha,\lambda] + \sum_{i,j} \hat{H}^{(p)}[Z_h(i,j)\,;\alpha,\lambda] \tag{5.21}$$

where

$$\hat{H}^{(p)}(t\,;\alpha,\lambda) = \begin{cases} \lambda^2 t^2, & \text{if } |t| < q \\ \alpha - c(|t| - r)^2/2, & \text{if } q \leq |t| < r \\ \alpha, & \text{if } |t| \geq r, \end{cases} \tag{5.22}$$

$$c = 1/(4p), \tag{5.23}$$

$$r^2 = \alpha(2/c + 1/\lambda^2), \tag{5.24}$$

$$q = \alpha/(\lambda^2 r). \tag{5.25}$$

The parameters λ and α are the same as in the previous equations of the modified coupled-membrane, and the parameter p is varied during the minimization process, as described below.

The equation for $\hat{E}^{(p)}(u_{i,j,k})$ above is similar to Equation 5.16 on the page before, but the neighbor interaction function $\hat{H}^{(p)}(t\,;\alpha,\lambda)$ has a different form than $\hat{H}(t\,;\alpha,\lambda)$. The new neighbor interaction function is in fact the key to making the GNC algorithm work. It is designed so that $\hat{E}^{(1)}(u_{i,j,k})$ is convex, and successive versions of the segmentation function—that is, $\hat{E}^{(1/2)}(u_{i,j,k})$, $\hat{E}^{(1/4)}(u_{i,j,k})$, etc.—increase in degree of nonconvexity in a slow and controlled manner (Blake & Zisserman, 1987). As $p \to 0$, the function $\hat{H}^{(p)}(t\,;\alpha,\lambda)$ becomes more like $\hat{H}(t\,;\alpha,\lambda)$ and $\hat{E}^{(p)}(u_{i,j,k})$ becomes more like $\hat{E}(u_{i,j,k})$.

Given the definition of a one-parameter family of cost functions $\hat{E}^{(p)}(u_{i,j,k})$, the rest of the GNC minimization algorithm is simple. In the first iteration, $p = 1$ and the function $\hat{E}^{(1)}(u_{i,j,k})$ is minimized. Once the minimum is found, in the next iteration, the value of p is decreased by one-half of the previous p, and the result of the previous iteration is used as the starting point for the minimization of the next $\hat{E}^{(p)}(u_{i,j,k})$.

Minimizing $\hat{E}^{(p)}(u_{i,j,k})$

The last remaining detail is the question of how to minimize the individual $\hat{E}^{(p)}(u_{i,j,k})$ functions. Most approaches to doing this rely on the same basic idea: minimization is accomplished by making successive, gradual adjustments to the individual $u_{i,j,k}$ nodes in an effort to reduce the value of the overall function $\hat{E}^{(p)}(u_{i,j,k})$ to a minimum. Blake and Zisserman (1987) provide several example algorithms for doing this.

The simplest approach is *direct descent*; it involves making a small change to a node $u_{i,j,k}$, testing the resultant value of $\hat{E}^{(p)}(u_{i,j,k})$, and accepting the change to $u_{i,j,k}$ if it leads to a reduction in the value of $\hat{E}^{(p)}(u_{i,j,k})$. However, this method is exceedingly time-consuming. The cost of computing the value of $\hat{E}^{(p)}(u_{i,j,k})$ is extremely high for the three-dimensional case of the modified coupled-membrane model, making the simulation run too slowly even for research purposes.

A more efficient class of approaches is based on *gradient descent*. These methods derive information from the gradient of the energy function, $\partial\hat{E}^{(p)}(u_{i,j,k})/\partial u_{i,j,k}$, to guide the adjustments to $u_{i,j,k}$. For the present work on the modified coupled-membrane model, I adapted the *nonlinear successive over-relaxation* (SOR) algorithm described by Blake and Zisserman (1987). SOR is a deterministic gradient descent approach that can be implemented for parallel-processing or serial-processing computer hardware. (The simulations described in this dissertation were all implemented on serial-processing computers.)

SOR is a surprisingly simple method. It is an iterative approach, where at each iteration, the value

$$\frac{w}{T}\frac{\partial\hat{E}^{(p)}(u_{i,j,k})}{\partial u_{i,j,k}} \tag{5.26}$$

is subtracted from each node $u_{i,j,k}$. This update is performed sequentially, from the first $u_{i,j,k}$ to the last. The operation is performed repeatedly until certain criteria (discussed below) are met. When these criteria are achieved, the function $\hat{E}^{(p)}(u_{i,j,k})$ is considered to have reached its minimum.

The constant T in Equation 5.26 is an upper bound on the value of the second derivative of $\hat{E}^{(p)}(u_{i,j,k})$; its value is found from

$$T \geq \frac{\partial^2\hat{E}^{(p)}(u_{i,j,k})}{\partial u_{i,j,k}^2} \quad \forall u_{i,j,k}. \tag{5.27}$$

The constant w in Equation 5.26 has value $0 < w < 2$ and is called the *SOR parameter*; it controls the speed of convergence of the SOR algorithm. Its optimal value for weak-membrane types of models was found by Blake and Zisserman (1987) to be

$$w = \frac{2}{1 + 1/(\lambda\sqrt{2})}. \tag{5.28}$$

An important question is when to stop the minimization of each $\hat{E}^{(p)}(u_{i,j,k})$ function. Even in the case of the first (convex) function in the series, reaching the true minimum can be time-consuming, requiring hundreds of iterations in the case of the modified coupled-membrane model. Numerical roundoff errors may prevent reaching the true minimum altogether, and in any case, most of the benefit of the minimization is reached early on, with results converging asymptotically and with only small additional improvements in the long tail of the process. It is therefore useful to establish a stopping threshold based in some way on the convergence properties of the minimization algorithm.

Blake and Zisserman (1987) suggest using the infinity norm, $||\mathbf{x}||_\infty = \max_i |x_i|$, for measuring progress towards the goal. Progress towards convergence can be estimated from the quantity

$$||\mathbf{u}^{(n+1)} - \mathbf{u}^{(n)}||_\infty, \tag{5.29}$$

that is, the difference between the values of all the $u_{i,j,k}$'s at some iteration n and their values at the next iteration. Blake and Zisserman offer an analysis of convergence of SOR and other schemes for the continuous form of the weak membrane, but not the discrete form (which is what is actually minimized in GNC).

As a practical matter, I have found that the following combination of stopping criteria work well enough for minimizing the modified coupled-membrane model. The idea is to measure both the quantity in Equation 5.29 and the number of iterations in a given pass of SOR, and stop the current pass when either quantity reaches a specified threshold. For the convergence criterion, I use a threshold based on the largest $d_{i,j,k}$ value,

$$\delta_{\min} = c_g \max_{\{i,j,k\}} (d_{i,j,k}) \tag{5.30}$$

where c_g is a small constant (set to 0.00005 in the simulations reported here). The parameter c_g establishes how small the difference between node values must be on successive iterations of the inner GNC loop before the function is considered approximately minimized. When, during minimization of $\hat{E}^{(p)}(u_{i,j,k})$ at a given p, the norm in Equation 5.29 drops below the value of $\delta_{\min}$, the functional value is deemed close enough to final convergence and the loop ended.

The second stopping criterion, based on number of iterations, is designed to avoid overly long minimization runs. I have found that the results of minimizing the modified coupled-membrane function in the present implementation do not improve noticeably when allowed to run longer than about 800 iterations. For the simulations reported in this chapter, the iteration stopping criterion was set to 800.

Figure 5.3 summarizes the main loop of the GNC algorithm when combined with a serial SOR minimization scheme and the stopping criteria outlined above. The algorithm in Figure 5.3 consists of two nested loops. The stopping criterion on the inner loop is based on the two criteria described above. When either of the stopping criteria is reached, the inner loop is stopped, p is set to the next value, and the inner loop is executed again. For the simulations described in Experiment Three below, with the value of λ usually being 18, the outer loop is iterated six times.

Extracting Borders From the Minimization Results

Once the GNC/SOR minimization process is completed, the borders can be extracted from the resulting $u_{i,j,k}$ values using Equations 5.19 and 5.20 on page 203. The procedure simply consists of detecting where the values of $Z_v(i,j)$ and $Z_h(i,j)$ exceed a threshold, and marking a border between the corresponding i, j locations in the input. Figure 5.4 summarizes the process.

5.3.3 The Roles of the Parameters in the Modified Coupled-Membrane Model

There are three parameters in the modified coupled-membrane model: α, γ, and λ. These represent the three degrees of freedom in the segmentation function, and together they control different aspects of the behavior of the function in detecting texture regions.

In the case of the weak membrane model, which has only λ and α, the roles of the parameters are well understood (Blake & Zisserman, 1987). Unfortunately, obtaining analytical relationships between α, γ, λ, and their effects on the modified coupled-membrane model is another matter: the additional couplings in that formulation greatly increase the difficulty of analyzing the interplay between the different parameters. Still, some headway is possible by examining certain restricted cases and by using the prior results of Lee (1993), who analyzed the original coupled-membrane model in some detail. In the paragraphs below, I discuss first the restricted case in which $\gamma = 0$ and there is only one orientation, and then the general case of $\gamma > 0$ and multiple orientations.

```
function $GNC\_SOR\_MAIN\_LOOP(d_{i,j,k})$
    $w := 2/\big(1 + 1/(\lambda\sqrt{2})\big)$
    $\delta_{\min} := c_g \max_{\{i,j,k\}} (d_{i,j,k})$
    forall $i, j, k$
        $u_{i,j,k} := d_{i,j,k}$
    end forall

    for $p := \{1, 1/2, 1/4, 1/8, \dots, 1/\lambda\}$
        $n := 0$
        repeat
            $n := n + 1$
            forall $i, j, k$
                $u_{i,j,k} := u_{i,j,k} - \frac{w}{T} \frac{\partial \hat{E}^{(p)}(u_{i,j,k})}{\partial u_{i,j,k}}$
            end forall
        until $(||\mathbf{u}^{(n)} - \mathbf{u}^{(n-1)}||_\infty < \delta_{\min}$   or   $n > n_i)$
    end for
    return new $u_{i,j,k}$'s
end function
```

Figure 5.3: Summary of the main GNC loop for serial SOR minimization (Blake & Zisserman, 1987), with modifications for implementing the stopping criteria described in the text. The first part of the function consists of initializations; the second part contains the GNC loop. The term $\mathbf{u}^{(n)}$ represents the vector of all $u_{i,j,k}$ values at inner loop iteration number n; $||\mathbf{x}||_\infty = \max_i |x_i|$ is the infinity norm; $\delta_{\min}$ is the threshold for accepting a minimization inner loop sequence as being close enough to ultimate convergence; n_i is a constant representing an upper bound on the number of inner loop iterations permitted; and T is set as discussed in the text.

```
function $EXTRACT\_BORDERS(u_{i,j,k})$
    initialize border storage array to empty
    forall $i, j$
        if $|Z_v(i,j)| > \sqrt{\alpha}/\lambda$
            mark a border segment between $i, j$ and $i - 1, j$
        end if
        if $|Z_h(i,j)| > \sqrt{\alpha}/\lambda$
            mark a border segment between $i, j$ and $i, j - 1$
        end if
    end forall
    return representation of borders
end function
```

Figure 5.4: Summary of the border extraction procedure. This process takes the $u_{i,j,k}$ values produced by *GNC_SOR_MAIN_LOOP* from Figure 5.3, and produces a labeling of the borders in the input image.

Table 5.1: Behavioral aspects of the basic weak membrane model

Name	Symbol	Description	Formula
Sensitivity threshold	h_0	Threshold for detecting isolated, steep gradients in response maps	$h_0 = \sqrt{2\alpha/\lambda}$
Gradient limit	g_l	Threshold for detecting shallow gradients in response maps	$g_l = h_0/(2\lambda)$
Adjacent step threshold	h_s	Threshold for detecting two adjacent steep gradients	$h_s = h_0\sqrt{\lambda/w}$

Based on Blake and Zisserman (1987).

Restricted Case: $\gamma = 0$ *and One Orientation*

If there is only one class of orientation responses in the modified coupled-membrane model, and $\gamma = 0$, then Equation 5.7 on page 195 simplifies to

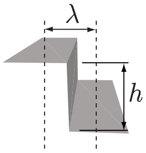
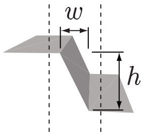
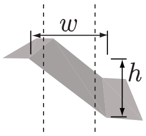
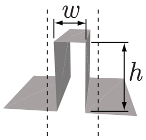
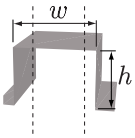
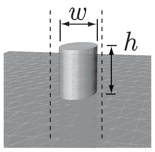
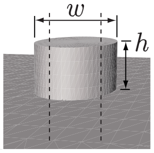
$$\begin{aligned} E = & \iint_A \left[U(x,y,\theta_1) - D(x,y,\theta_1)\right]^2 dxdy \\ & + \iint_{(A-B)} \left[\lambda^2\left(\frac{\partial u}{\partial x}\right)^2 + \lambda^2\left(\frac{\partial u}{\partial y}\right)^2\right] dxdy \\ & + \alpha \int_B dl. \end{aligned} \tag{5.31}$$

This is identical in form to the basic weak membrane (Equation 5.5), except that here, the data function $D(x, y, \theta_1)$ refers to APF values in a single orientation θ_1, rather than (say) image intensity values. This limited case of the modified coupled-membrane model obeys the same rules as the weak membrane, and so it is worth reviewing the weak membrane model's behavior here because it offers some insights into the functioning of the more general case.

Table 5.1 summarizes certain aspects of the weak membrane model's behavior that are controlled by quantities derived from parameters α and λ. The behavioral aspects are the sensitivity threshold, commonly denoted by h_0; the gradient limit, g_l; and the threshold for detecting adjacent steps, h_s. Blake and Zisserman (1987) provide analyses and derivations of these quantities. Table 5.2 on the next page summarizes their results, showing how these quantities are related to certain basic kinds of inputs.

The first case shown in Table 5.2 is a step transition in the input data. This kind of step transition might occur where there is a sharp border between a coarse texture and a fine texture. The result is a sudden change in the spatial frequencies registered in the

Table 5.2: Behavior of the weak membrane model for certain idealized input data

Feature	Data	Detection Condition	Comments
Step edge		$h > h_0$	• Step edge is isolated when separation from nearest other step edge is large compared to λ • Situation changes if two edges are near compared to λ—see case of adjacent steps
Steep gradient		$h > h_0$	• Gradient looks like a step edge for $w \ll \lambda$, so same threshold applies • May get double step if $h \gg h_0$ • Situation changes if $w \gg \lambda$ (see next case)
Shallow gradient		$\dfrac{h}{w} > g_l$	• Gradient is shallow if $w \gg \lambda$ • May get multiple breaks if $h/w \gg g_l$
Adjacent steps (near)		$h > h_s$	• Two steps are near if $w \ll \lambda$ • Solution has two breaks • Detection threshold increased by $\sqrt{\lambda/w}$ compared to case of isolated step
Adjacent steps (far)		$h > h_0$	• If step separation large compared to λ, the steps do not interfere with each other • Solution has two breaks
2-D top-hat (narrow)		$h > h_s$	• 2-D circular step with $w \ll \lambda$ is same as case of 1-D adjacent steps with $w \ll \lambda$
2-D top-hat (wide)		$h > h_0$	• 2-D circular step with $w \gg \lambda$ is same as case of 1-D adjacent steps with $w \gg \lambda$

unnormalized APF map, corresponding to a difference in the types of complex cells that are activated by the different texture scales in the input image. When the step transition is far enough away from other features in the data that it can be considered to be isolated (i.e., when the separation from the nearest other step edge is larger than λ), then the segmentation result will have a break at the location of the step edge if the difference in data values across the step is greater than h_0.

Similar to the case of a step transition is the case of a steep gradient in APF values. A gradient is considered steep if the width of the transition, w, is less than the value of λ. The weak membrane will signal a border if the difference h in values across the step is greater than h_0. Note that in the limit of $w = 1$, the gradient is equivalent to a step edge. Conversely, if the gradient in APF values is shallow compared to λ, the membrane will detect a border if the difference in values across the step (measured as $g = h/w$) is greater than g_l. If the difference g is significantly greater than g_l, then multiple breaks may be introduced in the membrane at the location of the gradient.

If two step transitions in the data appear close to each other (whether in an extended linear border, or in a two-dimensional "top-hat" shape), then they interact. The threshold for detection then becomes h_s—the steps are detected if the difference in values across the step is greater than h_s. Conversely, if the adjacent steps, or the edges of a two-dimensional top-hat shape, are far enough away from each other (far compared to the value λ), then the detection threshold is h_0, the same as for an isolated step.

The examples in Table 5.2 on the preceding page illustrate the sense in which parameter λ acts as a *scale* parameter. Discontinuities and other transitions in the input data are treated as separate, isolated events when the distance between them is large compared to the value of λ. When the data transitions are close in distance compared to λ, the discontinuities interact, and the threshold for detecting them is increased.

The idealized examples presented in Table 5.2 are noise-free. Resistance to noise in the input data is determined by parameter α, which determines the cost of introducing discontinuities in the segmentation output. Higher values of α make it more difficult to introduce a break in the membrane, which in turn reduces the likelihood that fluctuations due to noise in the data will be treated as discontinuities.

General Case: $\gamma > 0$ *and Many Orientations*

In contrast to the restricted case discussed above, the general case of $\gamma > 0$ and multiple orientations in $D(x, y, \theta)$ leads to much more complicated relationships between the parameters α, γ and λ in the coupled-membrane models.

Lee (1993) has analyzed a simplified, two-dimensional version of his four-dimensional coupled-membrane model. This two-dimensional version is given by

$$\begin{aligned} E = & \int_A \int_P \left[U(x,\psi) - D(x,\psi)\right]^2 d\psi dx \\ & + \int_{A-B} \int_P \left[\gamma^2 \left(\frac{\partial U}{\partial \psi}\right)^2 + \lambda^2 \left(\frac{\partial U}{\partial x}\right)^2\right] d\psi dx \\ & + \alpha \int_B dl. \end{aligned} \tag{5.32}$$

In this simplified formula, A is a one-dimensional spatial domain indexed by variable x and P is a one-dimensional spectral domain indexed by variable ψ. For Lee's formulation, ψ can represent either spatial frequency or orientation; the analysis applies equally to both. In the modified coupled-membrane model, the variable ψ corresponds to the orientation θ. The quantity $B \subset A$ is the set of boundary contours that divide up A into regions; λ and γ control the smoothing of the result across the x and ψ dimensions, respectively; and α is the penalty for creating discontinuities in $U(x,\psi)$.

In his analysis of this formula, Lee (1993) examined three cases: a step edge across ψ in $D(x,\psi)$, an infinitely long gradient across ψ, and a finitely long gradient across ψ. The case of the step edge and the finite gradient are similar to the step edge and shallow gradient cases, respectively, shown in Table 5.2 on page 210. However, there is the important difference that the data variations under consideration here are across the spectral variable ψ rather than across the spatial variable, x.

Lee's (1993) analysis concentrates on finding the critical values of α. These are the values that determine whether a segmentation break will be introduced at a given location, based on the input data and the other parameter values. The simplest case to analyze is an ideal step edge along the ψ-axis. For the modified coupled-membrane model, this corresponds to input data that contain a sharp change in the orientation dimension θ; Figure 5.5 on the following page provides an illustration of this. Lee (1993) has shown that the case of the step edge leads to the following expression for the critical value α^* for α:

$$\alpha^* = \frac{\lambda}{\pi\gamma}\left[1 - \frac{\Delta\theta}{\gamma} K_1\left(\frac{\Delta\theta}{\gamma}\right)\right]. \tag{5.33}$$

Here, $\Delta\theta$ is the change in the θ-dimension across the step edge, and K_1 is a modified Bessel function of the second kind (Abramowitz & Stegun, 1970, their Equation 9.6.25). The equation above expresses the theoretical minimum value of α that will prevent a segmentation break at a given location in the input. If, at some location, the combination

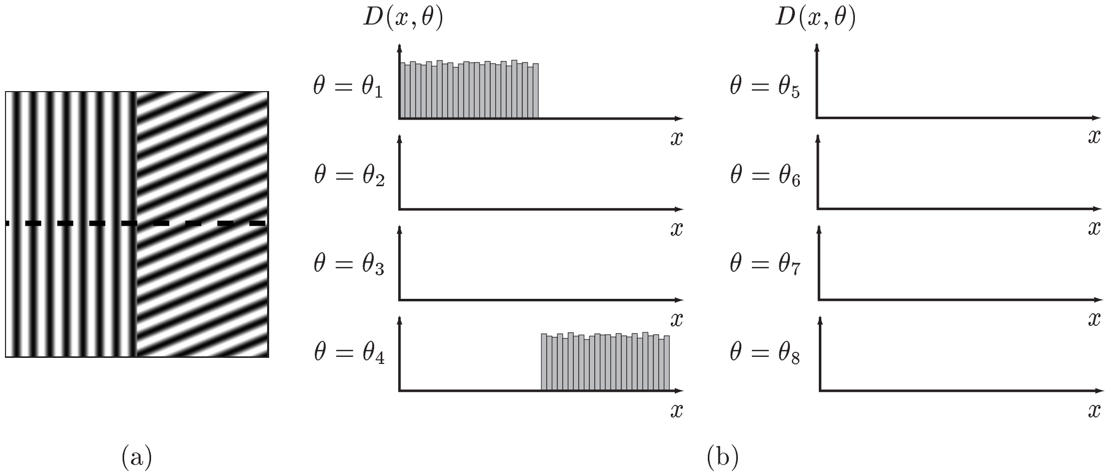


Figure 5.5: Illustration of an ideal step change across ψ in $D(x, \psi)$, for the case where ψ refers to orientation angle θ. (a) An example of an input image that contains a step change in orientation. (b) Suppose that APF values are computed for the input image shown in (a), and then a one-dimensional slice is taken horizontally through the middle of the resulting data (the dotted line). Each stylized bar graph shown here represents an idealized plot of $D(x, \theta)$ for a particular value of θ. One of the orientations shows a strong response over the first half of the image, but no response over the rest. A different orientation shows a strong response over the second half of the input. The change in orientations is $\Delta\theta$.

of $\Delta\theta$ and γ leads to $\alpha^* > \alpha$, then a segmentation break is introduced; if $\alpha > \alpha^*$, then no break is created. This result shows that the critical value of α is related in a nonlinear way to several quantities: the change across the step edge, $\Delta\theta$, and parameters λ and γ.

In practice, a true step edge in the orientation content of an input is nearly impossible to obtain. Recall that the complex-cell-like filters used to produce $D(x, y, \theta)$ are broadly tuned in orientation: they respond not just at a single orientation, but to a lesser degree, to neighboring orientations as well. This means that changes in the orientation responses to an input tend to be gradual rather than abrupt. Therefore, it is important to examine the cases of gradients across θ.

An example of a gradient in the values of $D(x, \theta)$ is illustrated in Figures 5.6 on the next page. This example involves a gradual shift in input data values across a limited spatial distance w along the x-dimension. If the gradient were infinitely long in the x-dimension, then, according to Lee (1993), the critical value of α would be determined from

$$\alpha^* = \frac{c^4 \lambda^3 \tan^3 \mu}{\pi\gamma} \left[\frac{b^3}{c(b^2 - c^2)^{3/2}} \tan^{-1}\left(\frac{\sqrt{b^2 - c^2}}{c} \right) - \frac{1}{b^2 - c^2} \right], \tag{5.34}$$

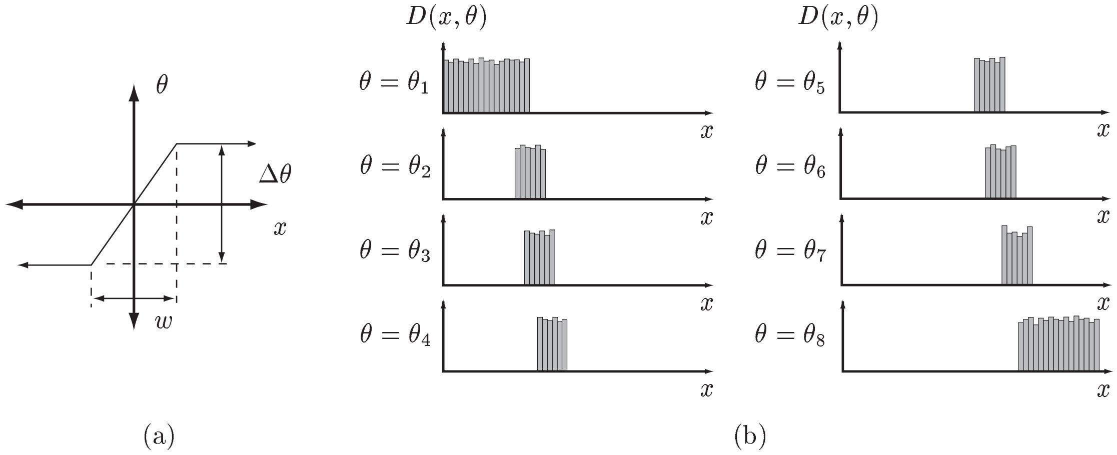


Figure 5.6: Illustration of a finite gradient across ψ in $D(x, \psi)$, for the case where ψ refers to orientation angle θ. (a) Graph of θ versus x. The width of the gradient refers to the distance w along the x-axis. The steepness of the gradient is determined by $\tan(\Delta\theta/w)$. (b) Suppose APF values are computed for an input image containing the kind of gradient shown in the graph at left, and then a one-dimensional slice is taken horizontally through the middle of the resulting data. Each stylized bar graph represents an idealized plot of $D(x, \theta)$ for a particular value of θ.

where

$$\tan\mu = \frac{\Delta\theta}{w} \tag{5.35}$$

$$b = 1/\gamma \tag{5.36}$$

$$c^2 = \frac{1}{\lambda^2 \tan^2 \mu + \gamma^2}. \tag{5.37}$$

As before, if $\alpha < \alpha^*$, a segmentation break is introduced; conversely, if $\alpha > \alpha^*$, no break is created.

Infinitely long gradients are not realistic for actual input data. Unfortunately, if the gradient is only finitely long rather than infinitely long, the resulting equations are exceedingly difficult to solve analytically. Lee (1993) presents example solutions derived numerically for certain combinations of parameter values. For more general cases, Lee points out that the solution for the case of the finitely long gradient can be approximated based on following principles:

- The solution for the finitely long gradient converges to the solution for the step edge as $\Delta\theta/w$ grows large. This happens when $w \to 0$, or in other words, when the width of the gradient becomes so narrow that it approaches a step edge.

- When $\Delta\theta/w$ is small (i.e., a shallow gradient), the solution to the finitely long gradient approaches the solution for the infinitely long gradient. This happens

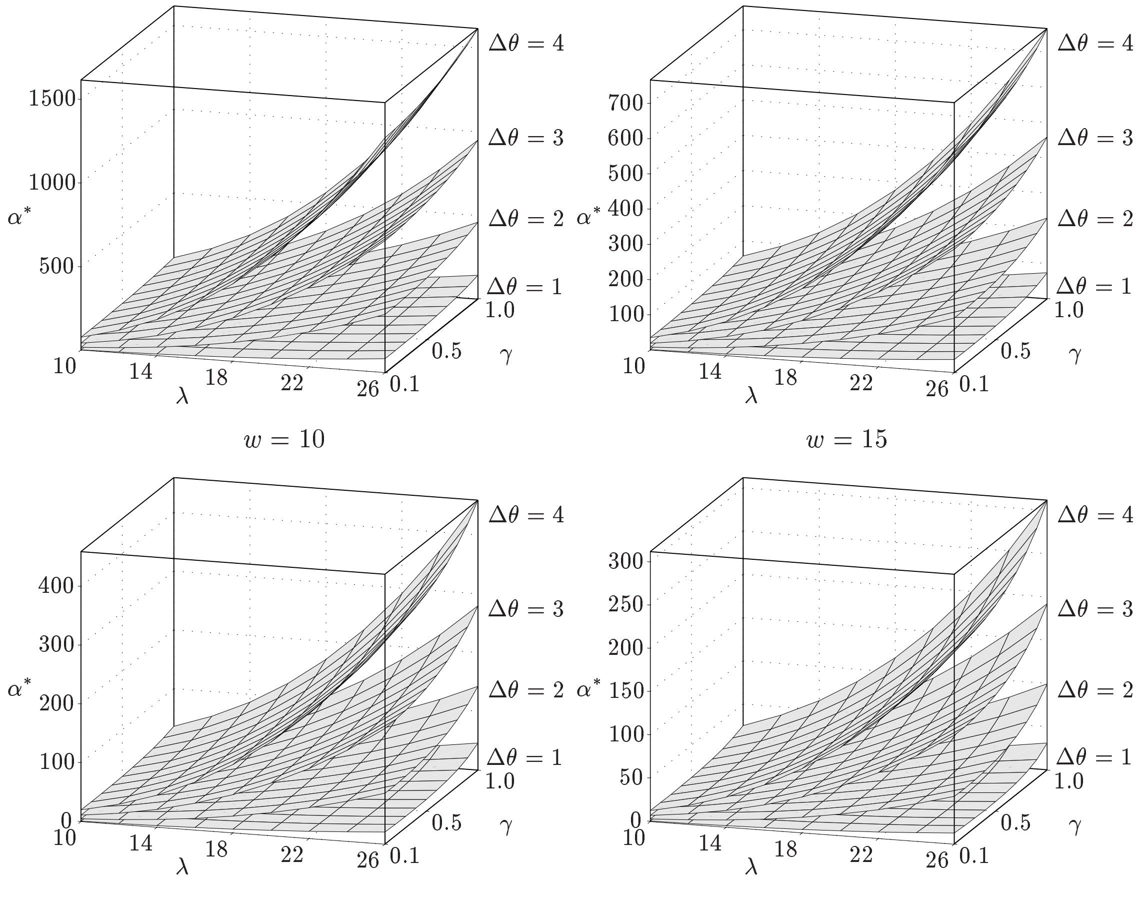
> when w is much larger than $\Delta\theta$. Therefore, for small values of $\Delta\theta/w$, Equation 5.34 on page 213 can be used as an approximation to the relationship for the finitely long gradient.

In the case of the modified coupled-membrane model, the typical width w of a gradient (which is measured in the spatial domain, i.e., in terms of spatial grid locations) tends to be significantly larger than the value of $\Delta\theta$. This is because the maximum value of $\Delta\theta$ for the modified coupled-membrane model as implemented here is $\Delta\theta = 4$. (The largest possible orientation difference occurs for orthogonal orientations, that is, those that differ by 90°. Each value of the orientation index θ in the model equals 22.5°; therefore, four is the largest $\Delta\theta$ for the implementation described here.) With typical values of λ ranging from 18–22, the corresponding range of $\Delta\theta/w$ is from 0.22–0.18, which is small enough to use the infinite gradient approximation.

To give a sense of the range of critical values of α, Figure 5.7 on the next page shows plots of α^* using Equation 5.34 and a variety of parameter values. Each of the four sets of plots shown in the figure represents a set of surfaces of Equation 5.34 for one of four values of the variable w. Within each plot, each surface represents values of α^* for a given value of $\Delta\theta$, plotted against variations in λ and γ. The meaning of α^* is as before: if $\alpha < \alpha^*$, a segmentation break will be created across the gradient, whereas if $\alpha > \alpha^*$, no segmentation break will be created.

Figure 5.7 illustrates that α^* is much larger for more abrupt gradients in orientation (i.e., for shorter widths w of the gradient). This can be seen from the upper left-hand plot, in which the maximum value of α^* is several times larger than the maximum value in the lower right-hand plot. The significance of this is that the shallower the gradient in orientation is, the easier it is to avoid a segmentation break. Relatively small values of α will prevent a break. Conversely, the narrower a gradient is, the easier it is to produce a segmentation break—it takes extremely high values of α to avoid it. Figure 5.7 also shows that the critical value of α increases with increasing orientation difference $\Delta\theta$. An increase in the difference across the orientation dimension of the input is simply another way of creating a more abrupt transition in orientation; thus, the value of α^* increases.

The figure also shows that the critical value of α does not vary much over the range of γ from 0.1 to 1.0, but it does vary significantly with changes in λ. The relationship between α^* and λ is an inverse one: for a given value of α, increasing λ makes it more likely that a segmentation break will be introduced. At first, this may seem to be at odds with the role of λ as a scale parameter: it may seem as though a larger value of λ should make it more difficult to introduce a segmentation break. But in truth, these two parameters always interact, in both the weak membrane and the coupled-membrane



$w = 10$

$w = 15$

$w = 20$

$w = 25$

Figure 5.7: Plots of surfaces produced using Equation 5.34 on page 213. Each of the four sets of surfaces shown above correspond to a different value of the distance variable w. Within each of the four boxes, each surface corresponds to the value of α^* that results when using a different value of $\Delta\theta$. The plots show in a graphical way the relationships between α^* (the critical value of α) and the other parameters in the model. The meaning of α^* is: if $\alpha < \alpha^*$, a segmentation break will be created across the gradient, whereas if $\alpha > \alpha^*$, no segmentation break will be created.

models. The pattern of behavior in Figure 5.7 is consistent with the inverse relationship between λ and the step detection threshold in the basic weak membrane (h_0 in Table 5.2 on page 210) as well as the relationship embodied by Equation 5.34 on page 213 for the coupled-membrane models.

To illustrate some of the effects of these parameters in practice, Figure 5.8 on the next page presents the results of applying the modified coupled-membrane model to an example image consisting of three regions (two containing texture, one blank). Shown in the figure are the results using a variety of settings of α, γ and λ.

The pattern of results in Figure 5.8 confirms the relationships implied by the plots of α^* in Figure 5.7. Small values of α result in many spurious segmentation breaks all over

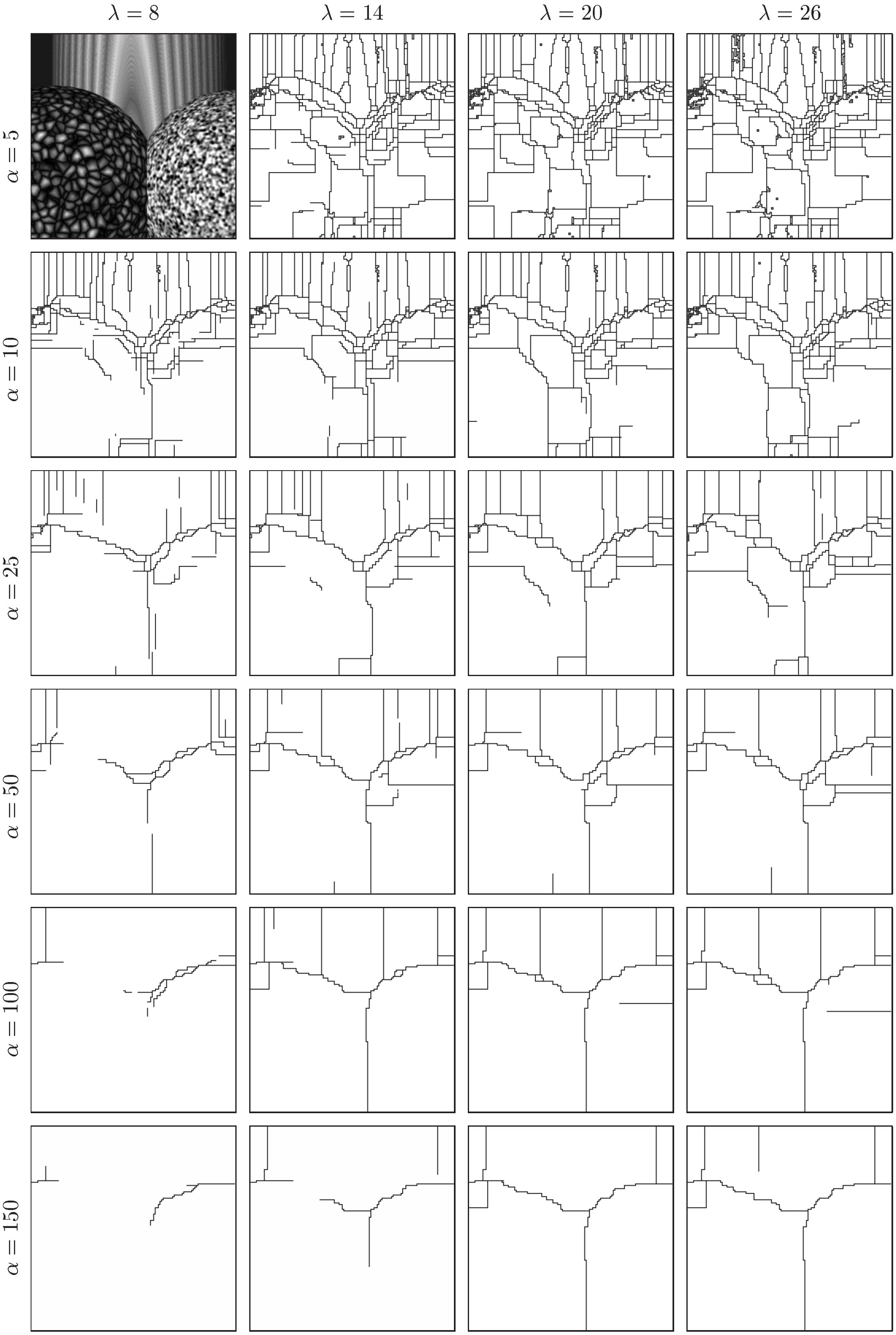


Figure 5.8: Illustration of the effects of varying the parameter values in the modified coupled-membrane model. Shown here are the results for $\gamma = 0.30$. (Continued on the next page.)

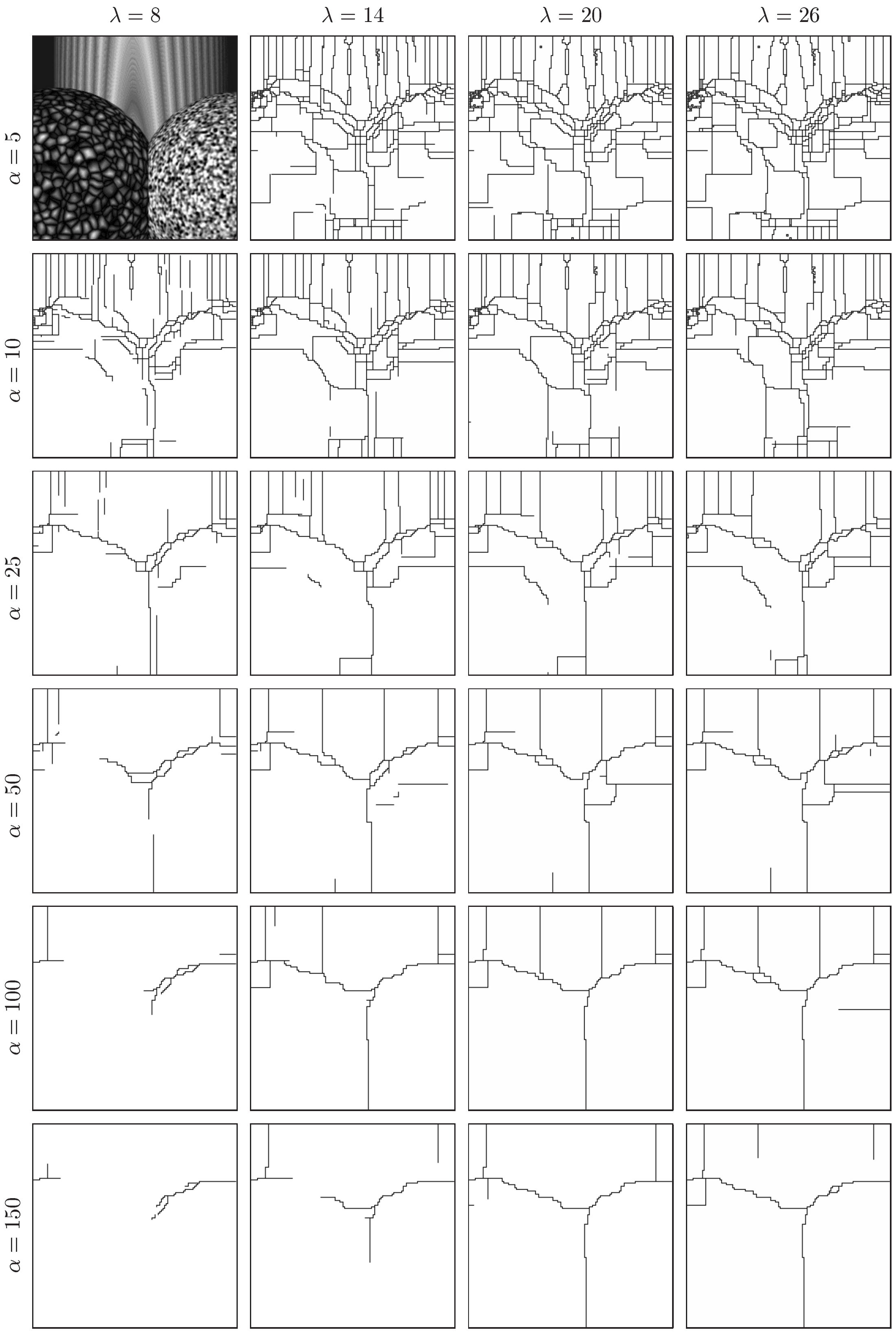


Figure 5.8: (Continued from the previous page.) Illustration of the effects of varying the parameter values in the modified coupled-membrane model. Shown here are the results for $\gamma = 0.70$.

the input image. Increasing α for a given combination of λ and γ reduces the number of segmentation breaks and tends to only leave borders between major texture regions. The effects of changing γ are somewhat harder to discern, but comparing the results for identical settings of α and λ reveals a slight tendency for the segmentation method to pay more attention to the dominant orientation in a given region when γ is increased. Compare, for example, the outputs for $\alpha = 25$ and $\lambda = 8$ at $\gamma = 0.3$ and $\gamma = 0.7$: at the setting of $\gamma = 0.3$, the vertical breaks in the lower middle area are more confused than at $\gamma = 0.7$.

Interpretations and Practical Implications

The results above show that the precise effects of the three parameters α, γ, and λ on the modified coupled-membrane model are considerably more complex than the effects of α and λ on the basic weak membrane. Still, the roles of the parameters are broadly similar:

- Parameter α imposes a cost on introducing segmentation breaks between regions of texture. The higher the value of α for a fixed combination of λ and γ, the less likely that a segmentation break will be introduced across a particular texture transition in the input. As such, α also controls immunity to noise.

- The scale parameter λ acts as a characteristic distance affecting interactions between discontinuities across space in the input data (where the data in this case are maps of APF values). Large values of λ help smooth over fluctuations in the data, thus avoiding segmentation boundaries across small-scale changes in the texture characteristics of the input image. However, λ interacts with α: an increase in λ may need to be accompanied by an increase in α, because higher values of λ raise the critical value α^* of α in Equations 5.33 and 5.34.

- Parameter γ controls the coupling between the weak membranes in different orientations θ. It modulates the value of α, primarily according to the relationship in Equation 5.34 on page 213.

In addition to these three control parameters in the model, there are two additional parameters in the simulation: c_g and n_i. These two control the minimization process according to the algorithm in Figure 5.3 on page 208. Table 5.3 on the next page provides a summary of all five free parameters in the modified coupled-membrane model.

Table 5.3: Free parameters in the modified coupled-membrane model

Parameter	Symbol	Description	Equation
Noise tolerance	α	Cost for introducing a segmentation break; controls likelihood of putting a border between texture regions and controls resistance to noise	5.7
Scale	λ	Characteristic distance affecting interactions between discontinuities across space in the data; large values smooth over fluctuations in the input	5.7
Orientation coupling	γ	Controls degree of coupling between responses at different orientations	5.7
Convergence factor	c_g	Factor used to set the maximum difference between node values during GNC/SOR minimization for calling the result approximately minimized	5.30
Maximum iterations	n_i	The maximum number of iterations of the inner GNC loop permitted during GNC/SOR minimization	n/a

5.4 Experiment Three: Segmentation of Images Containing Multiple Textures

This section presents the results of experimental testing of the segmentation model described in the previous section. The goal of this experiment is to verify that the modified coupled-membrane model works as intended, as well as to explore its general performance and compare it to that of the original coupled-membrane model. (In the discussions that follow, I use the acronyms *MCM* and *OCM* to stand for "modified coupled-membrane model" and "original coupled-membrane model", respectively.)

At the outset, it is worth reiterating that the segmentation component is only intended to produce an *approximate* description of regions in the input image based upon patterns of texture. The results do not need to be perfect. It is sufficient if the segmentations produced by the MCM model clearly capture the major regions in the input. Precise boundary shapes are not required.

Given this goal and other issues discussed in this chapter, I formulated three predictions for this experiment:

1. The MCM model should be able to segment a wide variety of textures. This includes flat textures viewed from the front (that is, the kinds of inputs typically used in tests of energy models of segmentation, as discussed in Chapter 2), as well as images

of textures on curved surfaces, and mixtures of regions of texture and no texture.

2. The segmentation results produced by the MCM model should be comparable in quality to the results obtainable using the OCM model. Because the OCM model has more degrees of freedom (it uses four data dimensions separately, instead of three as in the MCM model), it is possible that the OCM model will generally achieve more accurate segmentations. However, the differences should be limited to such aspects as smoothness of boundaries and precise localization of boundaries; there should not be differences in the number of major regions found by each model.

3. The simulation run-times for the MCM model should be significantly shorter than the run-times for the OCM model, because the MCM model is computationally less demanding.

The results of testing the predictions are discussed below.

5.4.1 Methods

I implemented the simulation of the modified coupled-membrane model as an addition to the complex cell and shape estimation simulations described in Chapters 3 and 4, respectively. As in previous chapters, the implementation consisted of a mixture of code written in the language C (Kernighan & Ritchie, 1988) and the Matlab environment (MathWorks, 1998a).

All inputs were 256-level gray-scale images 512 × 512 pixels in size. The set of test images is shown for reference in Figure 5.9 on the following page. I picked test cases that covered the classes of images mentioned in the predictions for this experiment. Image A contains two patches of orthogonally-oriented line elements, created using an ordinary computer drawing program. Image B contains two natural wood bark textures. Image C consists of a closeup of grass in the left half and a patch of textile in the right half. The textures in B and C are from the Brodatz (1966) album. Image D contains a closeup of raffia weave inset in a larger image of the same texture, while image E contains a patch of raffia weave inset inside an image of cloth; both of these images were kindly provided by Trygve Randen from the database at Høgskolen i Stavanger (Randen, 1997). Image F is identical to the golf ball used in Experiment Two in Chapter 4. Images G, H and I contain multiple textured surfaces; they were generated in software using a ray-tracing rendering package (POV-Ray™ by POV-Team™, 1997). The textures on the surfaces in images G, H and I are filtered random noise, artificial wood, and (in the case of the left-most spherical shape in H) random tessellations. Image I is intended to emulate in a rough way

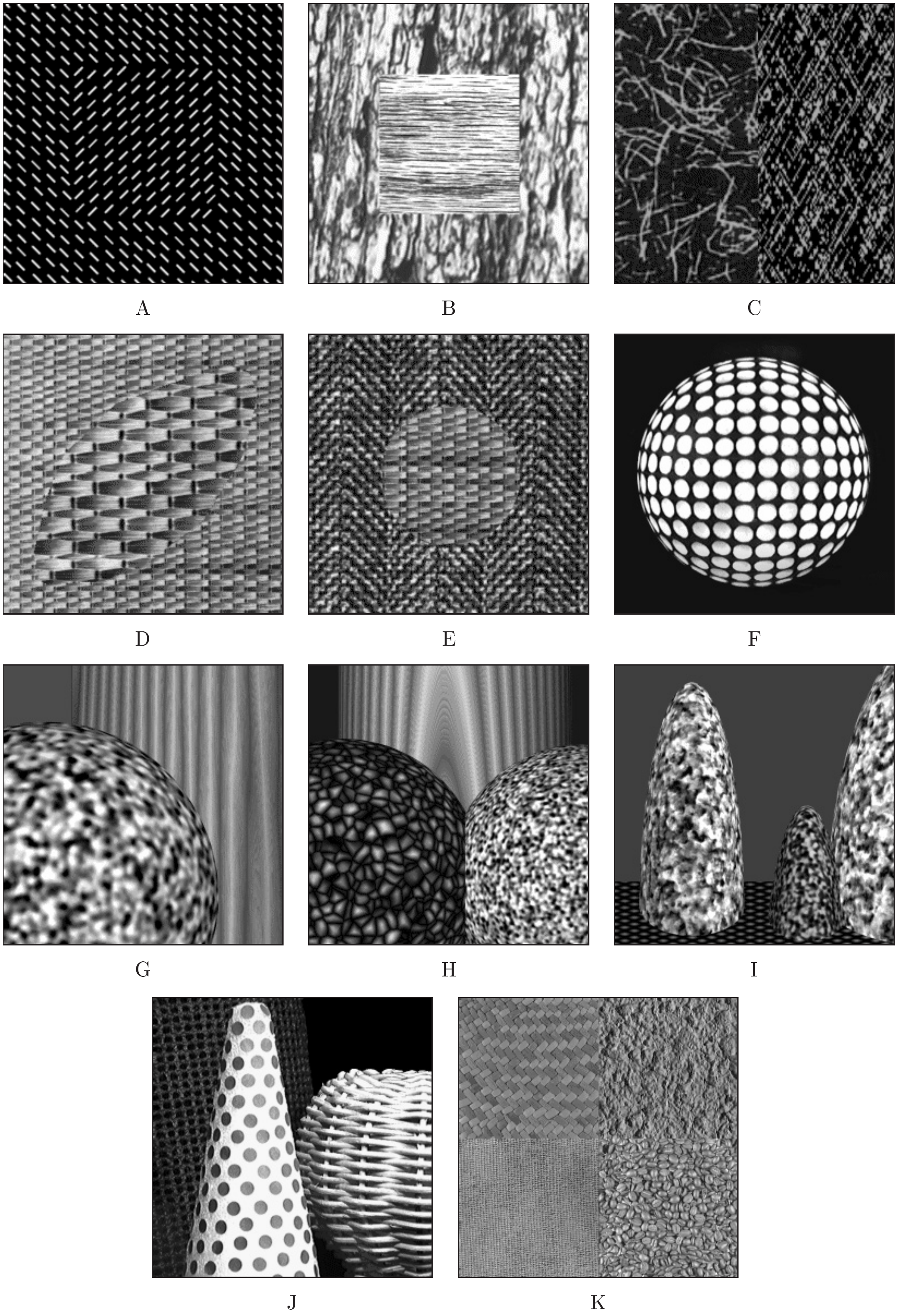


Figure 5.9: Images used as inputs for Experiment Three. More information about the images and how they were created is provided in the Methods section.

the real scene pictured in Figure 1.1 on page 2; the real scene contains textures that are too fine to be usable in this simulation. Image J is a collection of objects photographed using the digital camera setup described in Section 4.4.1. Finally, image L is a collage of four images of flat texture patches: raffia weave, concrete, coffee and canvas; the four images were obtained from the VisTeX database (Picard et al., 1995).

I performed each simulation run of the MCM model in two steps. For each image, I first generated unnormalized APF maps for each complex cell orientation by using Equation 4.12 on page 137 and the first part of the simulation described in the previous chapter. Because computing the complex cell responses is a time-consuming step, I stored the unnormalized APF maps in hard disk files in preparation for the next phase of the simulation. (Another benefit of this approach comes from the fact that once the unnormalized APF maps for a given image are computed, they do not need to be recomputed for additional runs of the segmentation component.) In the second step of each simulation run, I executed the simulations of the MCM model using the unnormalized APF maps as inputs. Each test case typically required performing a large number of runs using different parameters, as described below.

Simulating the OCM Model

For purposes of comparing the results of the OCM and MCM models, I implemented a version of the OCM model in the same framework as the MCM model. I used essentially identical implementations for both models, with the exception that, in the case of the OCM model, the inner loop in the GNC/SOR algorithm in Figure 5.3 on page 208 had an additional subloop to index over the spatial-frequency dimension.

To produce complex cell response maps for the OCM model with the same size as the unnormalized APF maps used as inputs for the MCM model (that is, to reduce the complex cell response maps from size 512×512 to 128×128), the response maps must be reduced by a factor of four in each spatial direction. The reduction process used in the computation of unnormalized APF maps involves a significant amount of averaging prior to subsampling (see the discussion surrounding Equation 4.12 on page 137 as well as Section 4.3.2). However, the OCM model formulated by Lee (1993, 1995; Lee et al., 1991, 1992) does not involve averaging the cell response maps. On the one hand, to average the raw complex cell response maps by the same amount as used in constructing the APF maps would undoubtedly introduce a degree of smoothing that is not in keeping with the spirit of the OCM model. On the other hand, *some* averaging *is* necessary prior to subsampling the response maps, in order to avoid aliasing. I resolved this conflict by performing the map size reductions using two levels of Gaussian pyramid reduction (Burt,

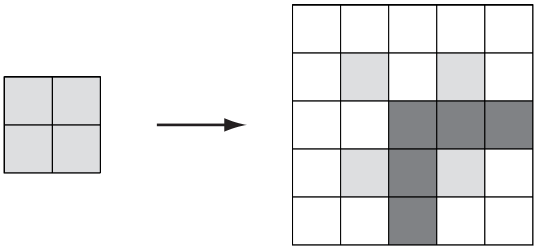


Figure 5.10: The *supergrid* representation (Jain et al., 1995). (Left) A closeup of a sample subset of data. (Right) Representing the data on a supergrid involves introducing additional grid locations in-between the original locations. Segmentation breaks are then displayed using short 3×1 line segments (shown here as the darkest-colored squares) between the actual data locations.

1981, 1984). This is equivalent to performing two successive steps, each step consisting of first smoothing the response maps with a small 5×5 Gaussian filter, then subsampling the result by two in each spatial direction. This approach seemed to result in map size reductions with a minimum of excess averaging.

Representing the Segmentation Results

The unnormalized APF maps have size 128×128 cells. The segmentations computed by the both the MCM and OCM models are represented in terms of breaks that actually fall in-between the cell locations. To display the results, I used a *supergrid* representation (Jain et al., 1995), in which additional grid locations are inserted between each cell location along the x and y directions. Each segmentation break is represented in this format as a short line segment placed between cell locations. Figure 5.10 illustrates the basic idea. The supergrid representation has size $(2n_x + 1) \times (2n_y + 1)$, where n_x and n_y are the widths of the original data.

The use of a supergrid is merely a convenient device for representing the segmentation results; it is not intended to imply the need for an equivalent neural structure. It does, however, introduce a small complication in the final, combined processing model, when the border-based segmentation output is converted into a region-based representation. The conversion issue is covered in the next chapter.

Setting Parameter Values

The most difficult issue that must be faced when running each of the two segmentation models is choosing values for the free parameters. There are five parameters in the MCM model: α, λ, γ, c_g, and n_i. There are six parameters in the OCM model: α, λ, γ_f, γ_θ, c_g, and n_i.

Table 5.4: Default parameter values used for simulations reported in this section

Modified Coupled Membrane		Original Coupled Membrane	
Parameter	Value	Parameter	Value
α	(variable)	α	(variable)
λ	18	λ	30
γ	0.60	γ_f	2.0
		γ_θ	0.6
c_g	0.00005	c_g	0.000000005
n_i	800	n_i	800

In order to find general ranges of parameter values that produced good segmentations for each type of model, I performed trial-and-error testing in preliminary experiments using both the set of images shown in Figure 5.9 and additional images of a similar kind (not shown). I sought parameter values that led to segmentations containing continuous breaks only between the major regions of texture in each input image. Table 5.4 shows the final values used. Except where noted, all segmentation results discussed in this section for the MCM and OCM models were obtained using the parameter values shown in the table.

The most difficult parameters to set are α, λ, and γ for the MCM model, and α, λ, γ_f and γ_θ for the OCM model. I attempted to maintain the same values over all input images, but this proved to be impossible: no acceptable segmentation could be reached by fixing all three (or four, in the OCM model) of these parameters for all input images. However, I succeeded in finding values for two of the parameters, λ and γ for the MCM model, that worked reasonably well for all of the input images in the tests discussed below. Similarly, I found values for three of the parameters in the OCM model, λ, γ_f and γ_θ, that worked for all of the test images. This left one parameter, α, that needed to be adjusted manually for each input image for both models. I obtained the segmentation outputs shown in the results below by running the segmentation models with different values of α for each image, and selecting the best segmentation for each input.

Clearly, the use of manual intervention to produce the final segmentations is undesirable. It means that the segmentation model as implemented here is supervised rather than unsupervised. I discuss this problem further in Section 5.4.3.

Setting the minimization control parameters c_g and n_i is a considerably easier task. The goal in setting these parameters is to balance the number of iterations performed during minimization against the quality of the minimization results. Ideally, the process should perform the fewest number of iterations such that running more iterations does not change the results appreciably. Experimenting with different input images and different settings of c_g and n_i makes it possible to eventually settle on values that balance these goals. Once set, the parameters need not change.

Finally, the difference in the values of c_g shown in Table 5.4 between the MCM and OCM models is due to the difference in the ranges of input values to the two implementations. For the MCM model, the inputs consist of maps of average peak frequencies, and the input values range from zero to a maximum of 22.6. For the OCM model, the inputs are complex cell responses and the input values range from zero to a maximum of approximately 1.0. These differences affect the value of the threshold $\delta_{\min}$ of Equation 5.30 on page 206 that must be used for the two models; the values of c_g reflect these differences in the ranges of input values.

5.4.2 Results

The first two predictions (that the MCM model should be able to segment a wide variety of textures, and that the results should be comparable in quality to those obtained using the OCM) should ideally be examined together for each of the test cases. Unfortunately, on the sizes of data being used here, the OCM model takes a tremendous amount of time to compute results (an average of 48 hrs per run). Finding the initial ranges of parameter values that produce acceptable results requires making hundreds of simulation runs, which made it impossible to create a detailed comparison between the OCM and MCM models within the time constraints of this project. Instead, I resorted to comparing the two models on a subset of the input images. The results for the predictions are therefore presented separately: first I present the segmentation results of the MCM model, then a comparison between the MCM and OCM models on a subset of the test images, and finally an examination of the differences between the two models in simulation running times.

Segmentation Results for the Modified Coupled-Membrane Model

For each test case, I examined the outputs of the simulation visually and evaluated them on the basis of how accurate and how smooth the segmentations appeared.

MCM Segmentation Results for Test Image A

Figure 5.11 on the following page shows the segmentation output of the MCM model when presented with the first test case. The input image contains a region of short line segments oriented at 45°, set inside a larger region of orthogonally-oriented line segments.

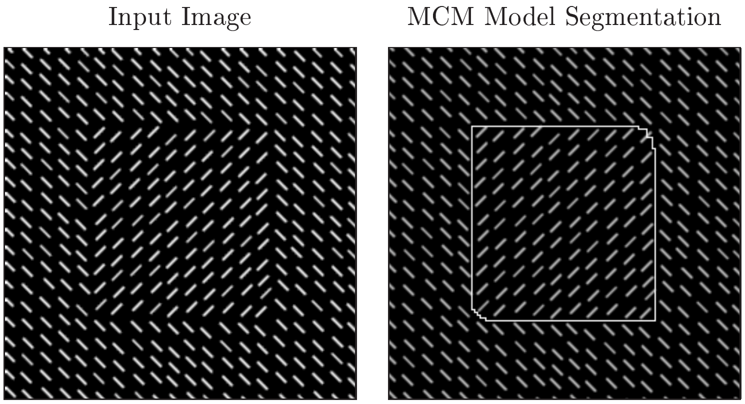


Figure 5.11: Segmentation results for test image A. The input image is shown in the left-hand column; the segmentation output is shown in the right-hand column, superimposed on a dimmed version of the input image. Segmentation produced with $\alpha = 36$.

The segmentation results are displayed as white lines superimposed on a dimmed version of the original image.

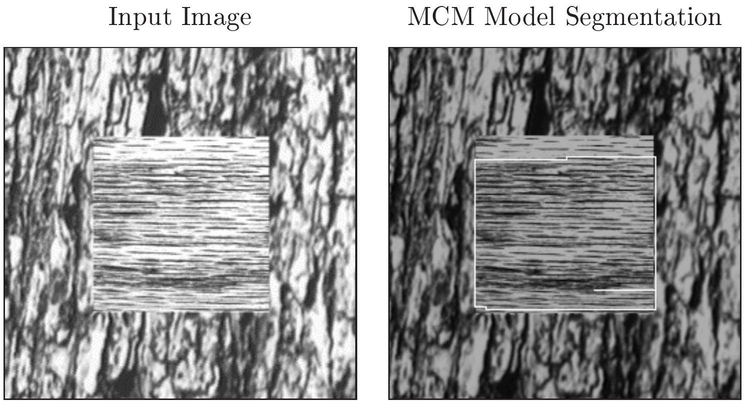
Figure 5.11 shows that the MCM model achieved excellent results for this simple test case. The simulation placed the boundaries between the inner and outer patches of textures with fairly high accuracy. The lines representing the borders fall almost exactly where the true region borders appear in the image, and there are no stray lines. Overall, the results are nearly perfect.

There are small artifacts in the corners of the segmentation boundaries. The errors are trivially small and inconsequential for the larger purpose of this dissertation. However, it is useful to note that the texture borders in the input image are in fact irregular—the borders do not form straight lines, and so it is not surprising that the resulting segmentation does not form a perfect square.

MCM Segmentation Results for Test Image B

Figure 5.12 on the next page presents the segmentation results of the MCM segmentation model when presented with the second test image. The image contains two wood textures, each having a different grain and different orientation.

The segmentation result of the MCM model for this test case is good, though certainly not perfect. Three of the four borders are located all mostly exactly where the true borders are, but there is significant localization error in the top border. The segmentation algorithm estimated the top border to be noticeably further into the middle square than the true texture border. The reason for this is not clear. It is possible that the horizontal grains of the wood texture influenced the estimated position of the texture border, but



Input Image

MCM Model Segmentation

Figure 5.12: Segmentation results for test image B. Segmentation produced with $\alpha = 493$.

this answer is not satisfactory given that the horizontal grain is more pronounced in other locations inside the middle texture patch. Alternatively, it is possible that the slight brightening at the upper part of the inner texture square somehow misled the segmentation process. The brighter area could have have given the impression that the border transition is lower within the inner texture square than it really is.

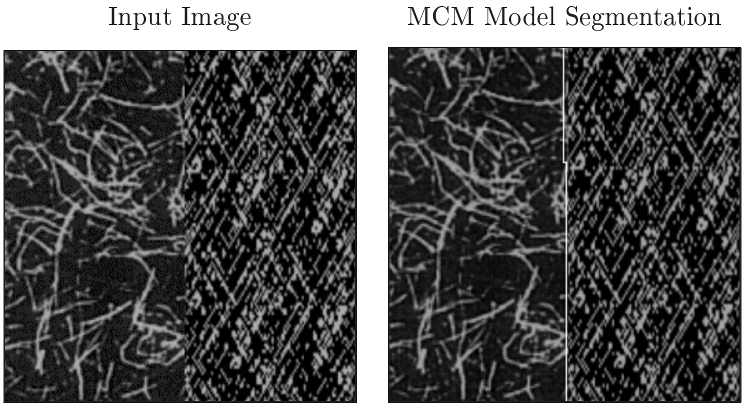
There is also a short, stray line segment in the lower right of the inner texture region. Although stray line segments such as these are undesirable, ultimately they will not affect the combined segmentation and shape estimation model. Chapter 6 describes certain clean-up steps that are performed on the segmentation output in the combined model; one of these steps eliminates unclosed line segments.

On balance, these errors are relatively small, and the segmentation results for MCM for this case agree with expectations.

MCM Segmentation Results for Test Image C

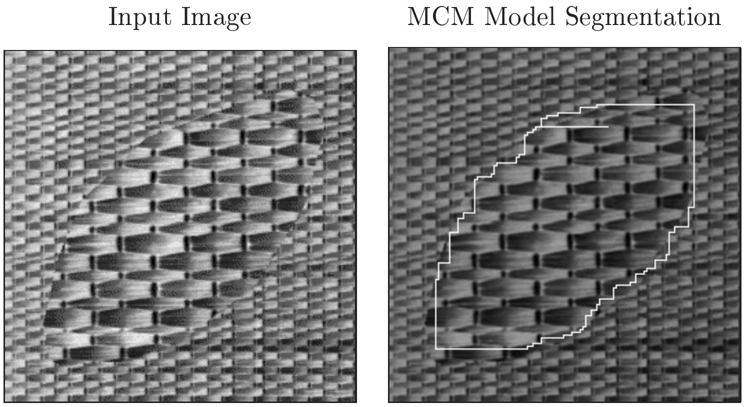
Figure 5.13 on the following page presents the segmentation results for the third input image. The two side-by-side textures pictured in the image are both dominated by bright lines on a dark background, and both have roughly the same spatial scale. The two texture patches differ mainly in their orientation qualities.

The results for the MCM model for this simple case are nearly perfect, with only a small crook in the border between the two regions. Except for this small crook, the border falls almost exactly where it should.


Input Image

MCM Model Segmentation

Figure 5.13: Segmentation results for test image C. Segmentation produced with $\alpha = 166$.


Input Image

MCM Model Segmentation

Figure 5.14: Segmentation results for test image D. Segmentation produced with $\alpha = 73$.

MCM Segmentation Results for Test Image D

Figure 5.14 presents the segmentation results for the MCM model when given the fourth input image. The two textures in this image are both raffia weave, and differ primarily in the scale of the texture elements. Unlike the previous three cases, this input contains a curved border between the textures rather than a straight-line border.

The results show that the model successfully captured the general aspects of the texture borders. The texture borders have a rather rough quality, and the segmentation method appears to have had difficulty closely tracking the curved contours of the inner texture patch. This difficulty is probably compounded by the strong vertical and horizontal components of the textures in the image. Despite this roughness, the results are actually quite good, considering that it is difficult even for human observers to discern the exact location of the border in some parts of the image. Overall, the model's output can be considered acceptable as a rough segmentation of the input.

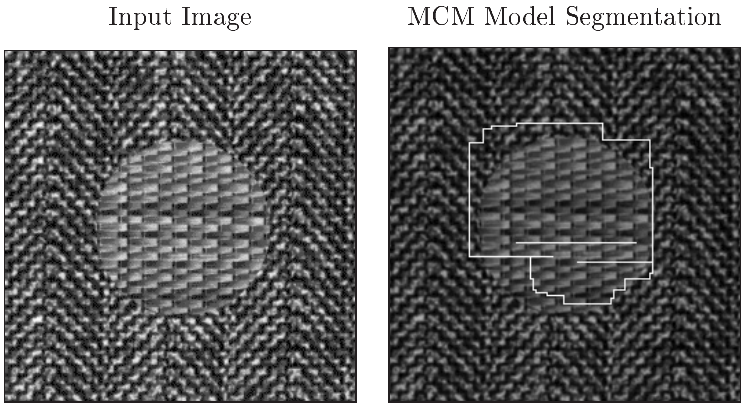


Figure 5.15: Segmentation results for test image E. Segmentation produced with $\alpha = 27$.

MCM Segmentation Results for Test Image E

Figure 5.15 shows the results of the MCM model on the fifth test case. This image, like the previous test case, also consists of two patches of texture with curved borders and strong orientation components. In this case, the two textures are more similar in spatial scale but more different in the orientations of the textures.

Although it was possible to obtain a single closed region with the MCM model, the result remains somewhat disappointing. The region borders are extremely blocky; indeed, it is not possible to tell from the output that the inner texture region is actually circular. In addition, the segmentation borders are not centered on the circular texture region; instead, they are slightly closer to the top border of the image than they should be. On the other hand, despite the roughness of the segmentation, the results are still reassuring because the essence of the input (a texture difference resulting from one patch inlaid inside another) is captured in the output.

Images with curved contours are known to be difficult for weak segmentation methods that use line processes constrained to the vertical and horizontal directions (Geiger & Yuille, 1991). The two line processes in the modified coupled-membrane model (the Boolean variables $h_{i,j}$ and $v_{i,j}$ in Equation 5.9) tend to encourage straight-line borders oriented vertically or horizontally. Thus, the circular texture patch in this test case represents a difficult class of inputs for this segmentation method—a circular texture border created by a weak difference between textures with strong linear components. In fact, the strong horizontal and vertical components in the textures themselves could have caused the segmentation method to tend towards the horizontal and vertical directions. These characteristics of the input may help explain some of the rough quality of the output in this case.

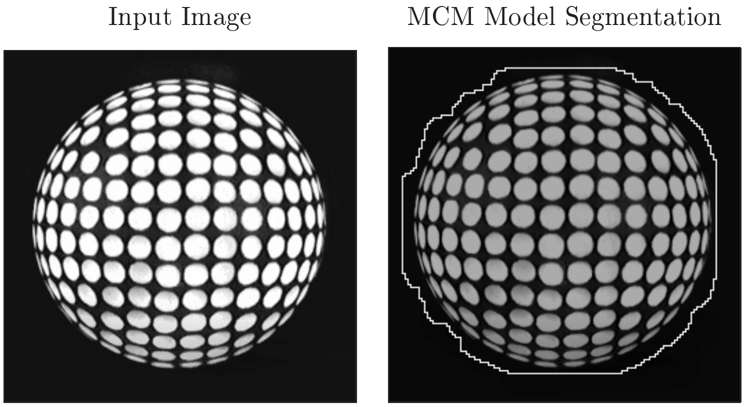


Figure 5.16: Segmentation results for test image F. Segmentation produced with $\alpha = 270$.

Another issue that could have contributed to the poor result is the particular settings of some of the averaging parameters. The averaging parameters (both in the computation of the APF maps and the parameter λ) may be large compared to the small diameter of the inner circle. This could have blurred the region borders and made it more difficult for the model to follow the circular outline.

MCM Segmentation Results for Test Image F

The sixth test case is a simple image designed primarily to check whether the segmentation model properly handles image regions devoid of texture. The image is a photograph of a painted golf ball. The results are shown in Figure 5.16.

The MCM model results for this case are reasonably correct. The output captured the single closed region in the middle. However, the segmentation border is slightly too far away from the surface border. The localization error is probably caused by one or both of the following problems. First, it may be a result of the spill-over of complex cell responses previously discussed in the context of Experiment Two. Recall that the complex cells tuned to lower spatial frequencies tend to show responses some distance away from the border between texture and nontexture; this could mislead the segmentation mechanism into placing the borders farther away from the true surface borders than it should. Second, it may simply be a result of the relatively large amount of averaging applied during the creation of the unnormalized APF maps. This averaging blurs the edges of surfaces, and could be what caused the segmentation model to slightly misplace the borders for the present test case.

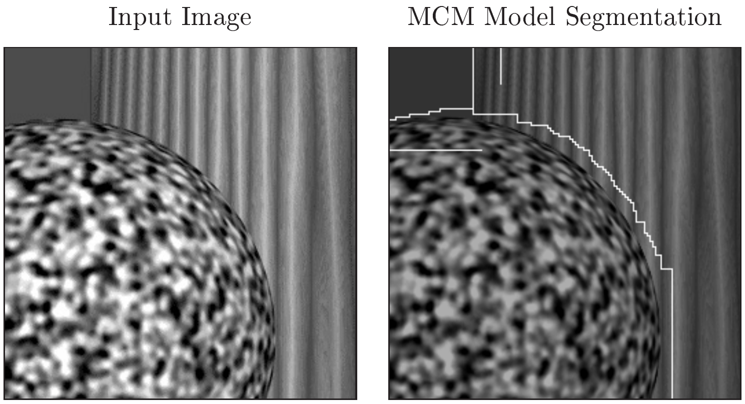


Figure 5.17: Segmentation results for test image G. Segmentation produced with $\alpha = 197$.

MCM Segmentation Results for Test Image G

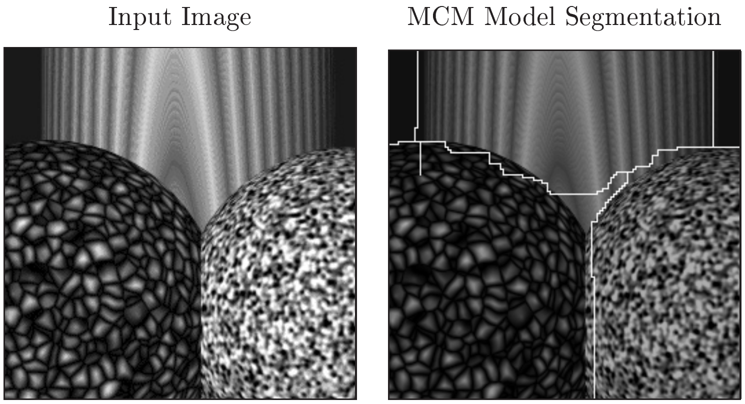
Figure 5.17 shows the results of the segmentation model for the seventh test case. There are two areas of texture in the image, both containing significant distortions introduced by surface curvature; there is also a third, blank area.

The results for the MCM model are good. The outlines of the texture regions are clearly visible in the output. There is a small stray line segment in the upper left of the circular region, but as mentioned above, such line segments will ultimately not be significant in the final model. Aside from that artifact, the only other error in the result is that the borders are slightly misplaced. The likely cause of this is the averaging performed in the early stages of the model, especially the computation of the APF maps. Despite this, the results are quite reasonable as an approximate segmentation of the input image.

MCM Segmentation Results for Test Image H

The next test case, shown in Figure 5.18 on the next page, is intended to provide a more complex input. It is similar to the previous test case, but contains one more texture region.

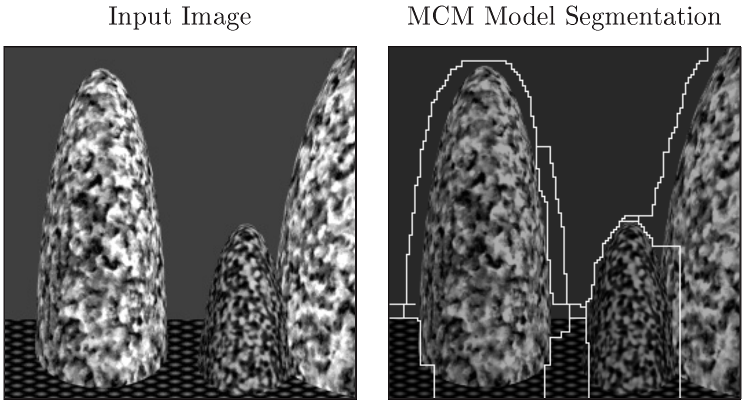
The output of the MCM model for this test case is again fairly good. Apart from a ragged corner in the middle of the image, and a short, stray border line in the upper left, the segmentation result agrees closely with all the major regions. The quality of the localization in this case is roughly comparable to that of the previous test case; the borders follow the region outlines fairly well, although in some places they fall a short distance away from the true region borders.



Input Image

MCM Model Segmentation

Figure 5.18: Segmentation results for test image H. Segmentation produced with $\alpha = 148$.



Input Image

MCM Model Segmentation

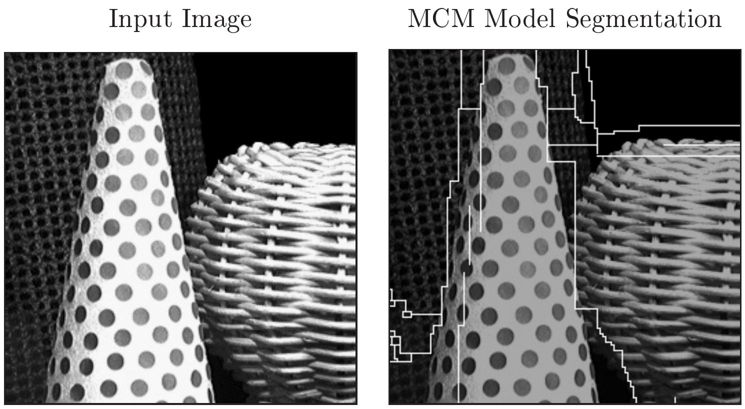
Figure 5.19: Segmentation results for test image I. Segmentation produced with $\alpha = 72$.

MCM Segmentation Results for Test Image I

Figure 5.19 shows the results of the MCM segmentation model for the ninth test case. The image is meant as an idealization of the example image used in Chapter 1 of this dissertation (Figure 1.1). It is a more complex image than the previous test cases.

The segmentation output from the MCM model is fair. It captures the major regions in the input, but also contains a few extraneous regions. These extra regions surrounding the main ones are small. Once again, the borders in the segmentation are located a short distance away from the true edges of the surfaces in the image, an effect already seen with test case F.

It is actually common for the coupled-membrane segmentation models to produce extra, small, closed regions such as those visible here. They occur for the next test case as well, and can occur for the previous test cases at certain settings of parameter α. This has not been an issue for images A through H, because the previous cases have all contained relatively large textured surfaces and only few borders. The presence of the extraneous



Input Image

MCM Model Segmentation

Figure 5.20: Segmentation results for test image J. Segmentation produced with $\alpha = 72$.

closed regions in the present case suggests that some additional processing on the segmentation results may be in order. In the next chapter, I discuss an additional processing step in which excessively small regions are removed from consideration for further processing. This has the effect of leaving the main regions intact, which is consistent with the goal of producing a segmentation that captures the main texture regions in the input.

MCM Segmentation Results for Test Image J

Figure 5.20 shows the results of the segmentation model for the tenth input image. This time the image is a photograph of several textured objects.

The MCM model results for this case show that it did capture some of the major region borders in the input, but overall, the model has failed to produce an adequate segmentation for this test case. There are many, extraneous, small- and medium-sized regions in the result, along with several stray line segments. In addition, the borders of the bottom of the cone-shape surface are not captured well; instead, the segmentation borders spread out significantly into neighboring regions (comparatively more so that in previous cases).

The MCM model output does show some promising aspects, however, and this suggests that perhaps some variation on the parameter settings could improve the results. I tested this hypothesis in a second set of experiments on this input image. I ran the MCM model with different settings of λ and γ, exploring the parameter space somewhat more widely than in the previous test cases. Increasing the value of λ did in fact improve the output, and by alternately increasing λ and readjusting α and γ, I found new parameter settings that produced a better segmentation of the input. The results using these alternate parameter settings are shown in Figure 5.21 on the following page.

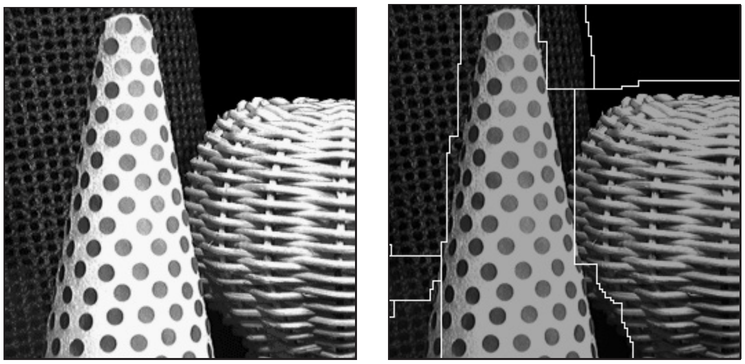


Figure 5.21: Segmentation results for test image J using alternate parameter settings. Parameter values: $\alpha = 128$, $\lambda = 34$, $\gamma = 0.5$.

Figure 5.21 shows that using $\alpha = 128$, $\lambda = 34$, and $\gamma = 0.5$, the segmentation method produced a better result. This time there were fewer extraneous areas, and the major texture regions were delineated in a more reasonable fashion. The main errors left in this output are the two medium-sized extraneous segments in the lower left-hand corner, and the mislocation of the lower right edge of the cone-shaped surface, where the segmentation erroneously includes the dark area between the cone and the basket as part of the cone region. It may be possible to improve the results further by exploring even higher values for λ, although I did not explore this possibility.

Thus, the segmentation model is in fact capable of producing an adequate result for test image J, but it requires different parameter settings. This points to the need for further research on methods for setting the parameters in the model.

MCM Segmentation Results for Test Image K

Figure 5.22 on the next page shows the results of running the MCM segmentation model on the eleventh input image. The input contains four equally-sized regions of different flat textures.

Figure 5.22 shows that with the settings of $\lambda = 18$ and $\gamma = 0.6$ used for the other test cases, the best output that could be achieved by the MCM model failed to represent the four texture regions of the input. Although the borders of three of the quadrants are approximately correct, there are far too many regions in the lower left quadrant of the image. Thus, the model did not produce an acceptable segmentation for this test case.

To see if any other parameter values would allow the MCM model to segment test image K correctly, I performed another small experiment to explore other areas of the parameter space. I searched through a region of the parameter space summarized in Table 5.5 on the following page. Altogether, I examined over 200 combinations of values.

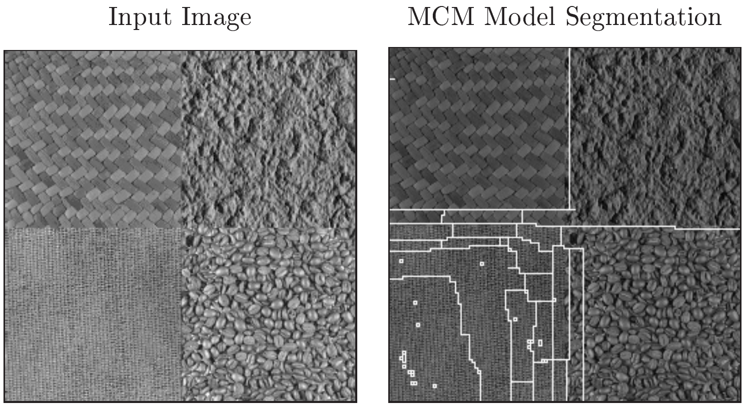


Figure 5.22: Segmentation results for test image K. Segmentation produced with $\alpha = 66$.

The combination of parameter values $\alpha = 353$, $\lambda = 40$ and $\gamma = 0.7$ produced the best results. Figure 5.23 on the next page shows the output produced by the model using these alternate parameter values.

Although the simulation results shown in Figure 5.23 represents an improvement over the original output, the model still failed to produce an adequate segmentation. There remain too many medium- to large-size regions in the lower left-hand quadrant of the output. Thus, even with the alternate parameter values, the MCM model failed to produce an adequate segmentation for this case.

I examine this failure in more detail in Section 5.4.3.

Segmentation Results for the Original Coupled-Membrane Model and Comparison to the MCM Model

Because of the time-consuming nature of the OCM model simulation, it was only possible to test the model on two images in the time available for this project. The two images that I selected were images A and H from Figure 5.9. The following paragraphs discuss the results for each image in turn.

Table 5.5: Alternate parameter values explored for input image K

Parameter	Range Tested	Original Value	Alternate Value
α	50 – 600	66	353
λ	18 – 68	18	40
γ	0.4 – 0.85	0.6	0.7

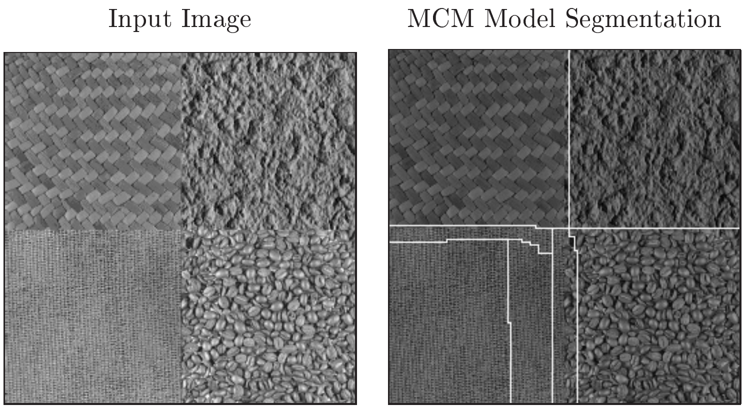


Figure 5.23: Segmentation results for test image K using alternate parameter values $\alpha = 353$, $\lambda = 40$, $\gamma = 0.7$.

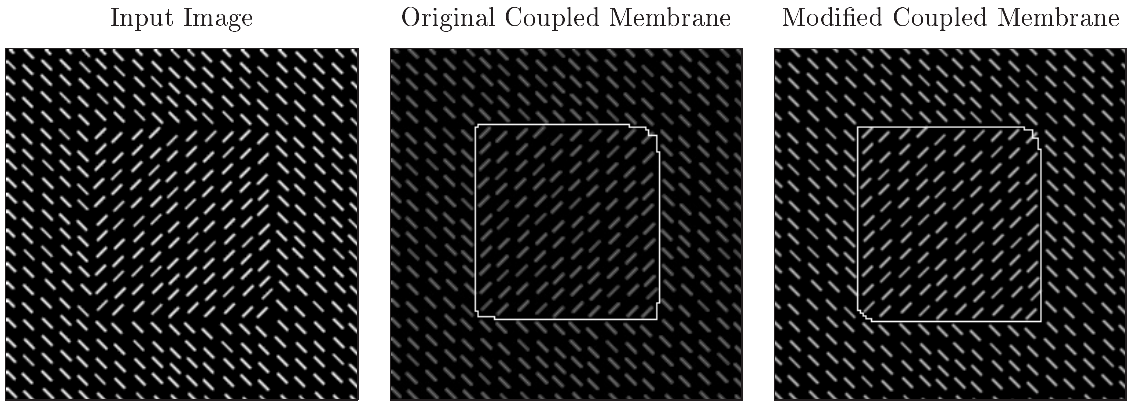


Figure 5.24: Segmentation results for the OCM model for test image A from Figure 5.9 on page 222. OCM parameter model settings: $\alpha = 0.03$. For comparison, the output from the MCM model from Figure 5.11 on page 227 is reproduced in the right-hand column.

OCM Segmentation Results for Test Image A

Figure 5.24 presents the segmentation results for the OCM model using input image A. The previously-shown results for the MCM model are also included in the figure for comparison.

The output that can be achieved using the OCM model on this test case is excellent. Figure 5.24 shows that the model placed the region boundaries accurately between the inner and outer patches of texture. As with the MCM model output, there are a few small artifacts in the corners of the segmentation boundaries, but these are not significant for the purposes of this research. Comparing the results between the OCM and MCM models shows that they are essentially identical in terms of quality and accuracy.

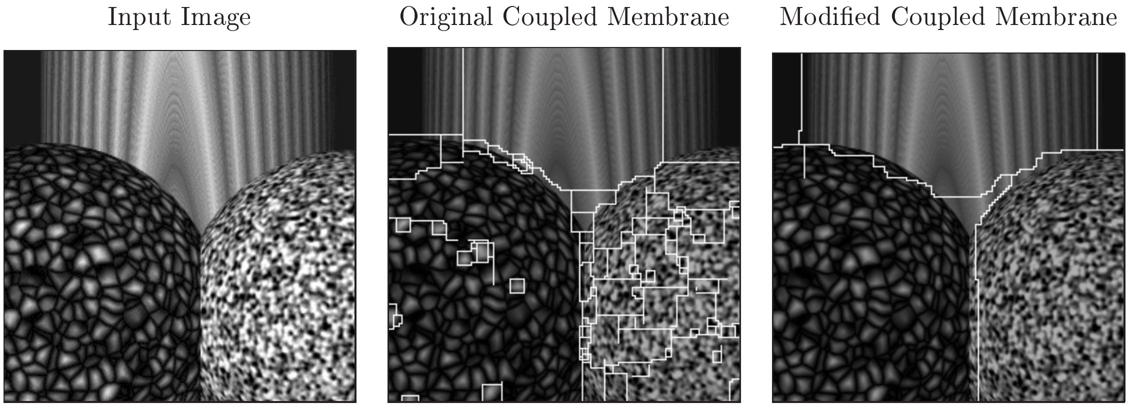


Figure 5.25: Segmentation results for the OCM model for test image H from Figure 5.9 on page 222. OCM parameter model settings: $\alpha = 0.0234$. For comparison, the output from the MCM model from Figure 5.18 on page 233 is reproduced in the right-hand column.

OCM Segmentation Results for Test Image H

Figure 5.25 presents the segmentation results from the OCM model when presented with input image H. There are three areas of texture in the image, all containing significant distortions introduced by surface curvature; there are also two blank areas.

The OCM model has faired much worse on this test case. Even the best segmentation that I could achieve contained a large number of stray border lines and stray regions in the segmentation result. By comparison, it was possible to achieve much cleaner results with the MCM model. The OCM model appears to be sensitive to small fluctuations within regions of texture. Further, the locations of the boundaries at the left and right sides of the upper middle surface are significantly misplaced in the OCM results from where they should be. The MCM model placed the vertical boundaries much closer to where a human observer would probably place them.

Thus, for this test case, the output from the simulation of the OCM model is disappointing. The model failed to produce acceptable results.

It is worth noting that since the simulations of the OCM model take so long, it was impossible to test as many variations in parameter settings as for the MCM model. It is therefore possible that the poor quality of the results for the OCM model in this case is simply due to not having found the range of parameter values that is best suited to this implementation.

Comparison of Simulation Run-Times Between the MCM and OCM Models

Actual program running times are generally not considered to be good indicators of algorithm performance. The absolute running time of a particular procedure is subject to many sources of variations, for example the quality of the implementation, the fit of the implementation to the particular computer architecture, the amount of core memory available to the algorithm at run-time, and countless other factors. A better type of comparison to make is in terms of time complexities or run-time *growth rates* (Aho, Hopcroft, & Ullman, 1983)

In comparing the original and modified coupled-membrane models, it is important to account for the difference in the way the inputs to the models are computed. The MCM model includes an additional step of calculating APF values from the complex cell outputs, and this must be included when comparing the run-time efficiencies of the OCM and MCM models. The complete set of run-time components for the OCM model simulation is

$$\boxed{\text{complex cells}} + \boxed{\text{reduction to } 128\times 128} + \boxed{\text{GNC}} + \boxed{\text{border extraction}}$$

For the MCM model, it is

$$\boxed{\text{complex cells}} + \boxed{\text{APF}} + \boxed{\text{reduction to } 128\times 128} + \boxed{\text{GNC}} + \boxed{\text{border extraction}}$$

The complex cell and border extraction stages are common to both models and can be ignored for purposes of comparing the models. The size reduction stages differ in the two models, but they both end up having the same upper bounds on run-times. The reason is the following. In the MCM model, the size reduction involves only subsampling, but in the OCM model it also involves convolution (because of the Gaussian pyramid reduction; see Section 5.4.1). In both cases, the running time of the subsampling operation is of order $O(n_w^2)$, where n_w is the width of a square input image ($n_w = 512$ in the simulations presented here). The extra convolutions performed in OCM always use the same constant-sized filter and are additive with the subsampling operation. Since the convolutions have order $O(n_w^2)$, the upper bound on the entire size reduction step in the OCM model remains $O(n_w^2)$. Thus, the size reduction stages can also be ignored for purposes of comparing the models. That leaves the differences in the segmentation procedure itself (GNC) and in the addition of the APF computation stage in the MCM model.

The worst-case running time of the APF creation stage is determined by the size of the input image; that is, the rate of growth is roughly of order $O(n_w^2)$, where n_w is the width of a square input image. (The averaging operation used in creating the APF maps introduces a large constant time factor, but does not change the upper bound on the time complexity.)

In the GNC/SOR procedures themselves, both the OCM and MCM models involve roughly the same constant number of numerical operations multiplied by the number of inputs to the two models. They differ in the sizes of the inputs. In the case of the

OCM model, there are $n_w^2 \times n_f \times n_\theta$ data points, where n_w is the width of the input to GNC/SOR (128 in this implementation), n_f is the number of frequency bands, and n_θ is the number of orientation bands. Thus, the complexity is of order $O(n_w^2 n_f n_\theta)$ for the OCM model. On the other hand, the MCM model takes only $n_w^2 \times n_\theta$ input data points, which means that its complexity is of order $O(n_w^2 n_\theta)$.

Putting these facts together, and using the rule that the overall time-complexity of a sequence of steps is given by the maximum of the time-complexities of the individual steps (Aho et al., 1983), the upper bounds are

$$O(n_w^2 n_f n_\theta) \tag{5.38}$$

for the OCM model, and

$$O\big(\max[O(n_w^2), O(n_w^2 n_\theta)]\big) = O(n_w^2 n_\theta) \tag{5.39}$$

for the MCM model. Therefore, the modified coupled-membrane model is indeed more efficient, in terms of running time growth rates, than the original coupled-membrane model, even after accounting for the extra computations involved in calculating the average peak frequency maps used as inputs.

These results are confirmed in practice. The original coupled-membrane model is extremely time-consuming to simulate on a serial-processing computer. Simulation run-times for inputs of the size used here (128 × 128 × 10 frequencies × 8 orientations) averaged around 48 hrs for the OCM model, even on the fastest computers at my disposal (e.g., an Sun Ultra™2 computer with one 300 Mhz UltraSPARC-II™ CPU). By comparison, on the same computer, the modified coupled-membrane model requires an average of 0.5 hr for the segmentation process and an additional 1.0 hr for creating the APF maps, for a total of 1.5 hrs in simulation running time.

5.4.3 Discussion

The results of this experiment show that the three predictions stated at the outset have mostly been met.

1. The modified coupled-membrane model developed in this chapter is able to segment inputs containing a wide variety of textures. For nearly all of the input cases tried in the experiment, the segmentation model was able to produce acceptable results. The exception was input image K (Figure 5.22 on page 236), for which the model captured the major texture regions but also reported additional regions not present in the input.

2. The segmentations produced by the modified coupled-membrane model are at least as good, and possible better than, those produced by the original coupled-membrane model. Due to the length of time required to run simulations of the OCM model, it has been impossible to provide a detailed comparison on all the test cases. However, based on the two test cases available, the results show that the MCM model actually performed better than the OCM model, in the sense that it was able to locate the major regions in the input without extraneous segmentation boundaries.
3. The simulation run-times for the MCM model, at least on a serial-processing computer, are much shorter than the run-times for the OCM model. Theoretical analysis of the upper bounds on the segmentation models also shows that the MCM model is significantly more efficient.

On the Failure to Segment Test Image K

The failure of the modified coupled-membrane model on test image K is disappointing. In order to try to understand the causes of this failure, I examined the set of unnormalized APF values for input image K. I followed the approach (previously used in Experiment Two) of plotting the quantity given by Equation 4.30 on page 174,

$$\max_{\theta}[O_2(x, y\,; \theta)],$$

where $O_2(x, y\,; \theta)$ represents the results from Stage Two of the shape estimation model (see Section 4.3.2). This permits visualizing the regions of high and low values in the inputs to the MCM model. Figure 5.26 on the following page presents a surface plot produced by applying the equation above to the unnormalized APF maps for image K.

The figure shows that the lower left-hand quadrant of the image is characterized by much higher APF values than the other three quadrants—the highest average peak frequencies in that region are nearly four times higher than the highest frequencies in any other region. The reason for the segmentation failure in Figure 5.22 on page 236 now becomes more apparent. To avoid the spurious segmentation breaks across the bottom left quadrant, λ and α would need to be set to relatively high values. But apparently at the kinds of large values of λ and α that are required to produce an acceptable segmentation across the bottom left quadrant, the segmentation process tends to miss the other texture borders in the image. Conversely, at the lower values of α and λ required to produce segmentation breaks between the other quadrants, the huge gradient around the lower left-hand quadrant leads to multiple segmentation breaks in that region. For this image, it proved impossible to satisfy these conflicting demands on the values of the parameters,

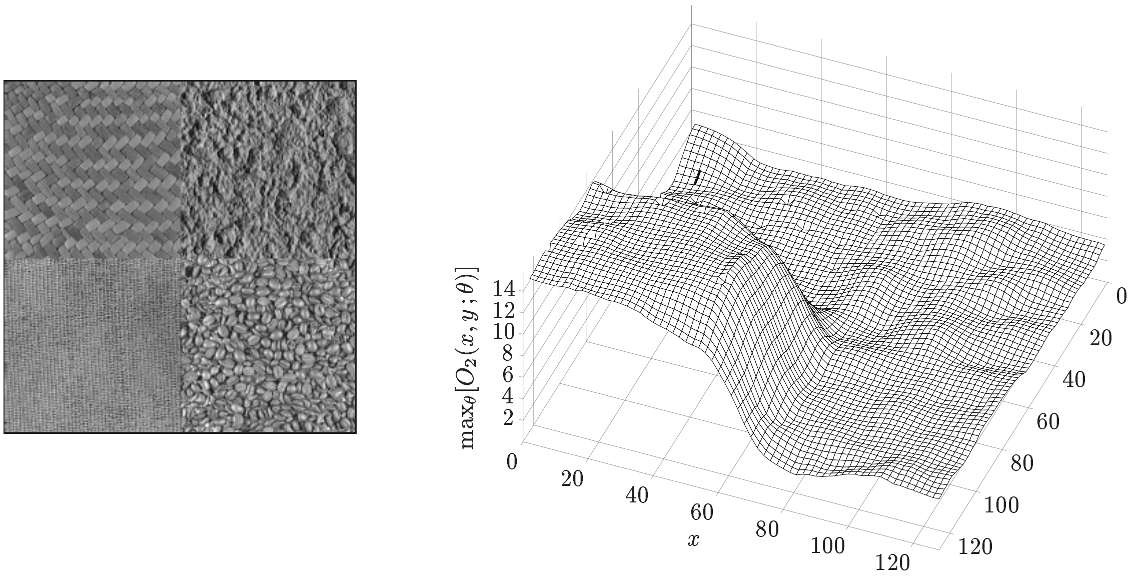


Figure 5.26: (Left) Input image K used in the results shown in Figure 5.22 on page 236. (Right) Surface plot of the the maximum unnormalized APF values across all eight orientations in the image. The values shown here were computed using the formula $\max_\theta[O_2(x,y\,;\theta)]$, where $O_2(x,y\,;\theta)$ represents the results from Stage Two of the model as described in Section 4.3.2. The raised region in the plot corresponds to the fine mesh texture in the input image. The surface plot shows that the maximum APF values in the lower left-hand quadrant are much higher than the highest APF values over any other quadrant; in fact, the difference is nearly a factor of four.

and therefore, even the best segmentation that could be achieved still had undesirable segmentation borders.

The root of the problem is that the value of parameter λ is maintained constant across the entire input image. An alternative approach that might succeed for a case such as image K is to allow λ to take on different values at different locations in the input. Then, λ could be set to a high value in the vicinity of the lower left-hand quadrant, and set to lower values over the other regions of the input. Blake and Zisserman (1987) mention this idea in the context of the weak membrane approach. More recently, Kubota and Huntsberger (1997) implemented a variation of Lee (1995)'s coupled-membrane model that also permits λ to vary depending on the local image statistics.

Thus, although the modified coupled-membrane model formulated in this chapter failed for test image K, there is enough work on the topic of adaptively setting parameter values to suggest that the deficiency can be corrected. I leave this improvement to future work.

On the Quality of the Segmentations

Some of the segmentation results for the MCM model, especially those for inputs containing curved texture boundaries, were relatively rough. The relatively large value of λ being used here may be one reason why the segmentation borders do not track the texture borders more closely in the case of curved textures. This parameter controls smoothing across space. The advantage of using a larger value is that small transitions in texture are ignored in favor of the major texture boundaries in the input, but the disadvantage is that the system does not track contours as closely as with a smaller value of λ. For the purposes of this research, I seek only a rough segmentation, so this is generally an acceptable compromise.

On the Problem of Manually Adjusting Parameters for Each Case

All of the segmentations in this experiment involved human intervention: I ran the models using different values of the α parameter for each test input, visually inspected each result, and chose the best output. The model *can* function with a single, set value for α for all inputs, but the results are not always of usable quality—for some images, the system may not find all texture regions, whereas for others, it may produce many spurious boundaries. Some test cases required substantial searching of the α parameter space before a good-quality segmentation could be had. By itself, then, the segmentation model described in this chapter is not complete. It requires supervision.

The need for human supervision indicates that the model is incomplete: it lacks a mechanism for automatically adjusting α to a value appropriate to each input. It is interesting to note that many published applications using the weak/coupled-membrane formalisms have also relied on manually adjusting the parameters to achieve adequate segmentations. This includes Lee's (1993, 1995; Lee et al., 1991, 1992) original formulation. The problem is therefore not limited to the texture-based segmentation model described in this dissertation, but rather is endemic to the class of weak/coupled-membrane approaches.

The type of mechanism that is missing is analogous to the mechanisms of light adaptation in the retina (Bruce et al., 1996; Kandel, Schwartz, & Jessell, 1991), as well as the contrast-normalization mechanism used in the model of simple and complex cells described in Chapter 3. These are just two examples of neural mechanisms that automatically adjust thresholds in the visual system, dynamically raising or lowering parameter values based on characteristics of a visual input. It seems likely that a similar mechanism could exist in the processes used for texture-based segmentation.

The need for such a mechanism in the weak/coupled-membrane class of models has been recognized by other researchers, and it is an active area of research. As mentioned above, Kubota and Huntsberger (1997) have examined this problem for the case of the original coupled-membrane model, and developed a method for estimating the model parameters based on the local statistics of the features in the input image. Their reimplementation of the coupled-membrane model functions without human intervention. Other work on models related to the weak membrane (e.g., Zhu & Yuille, 1995, 1996) has also addressed this limitation.

The use of manual intervention in the present work can therefore be justified as a step on the road towards a more complete model. Although I do not pursue it here, an obvious avenue for future exploration is adapting Kubota and Huntsberger's (1997) ideas to the MCM model.

5.5 Summary

In Chapter 2, I pointed out that most popular, biologically-inspired models of texture-based segmentation are designed for textures that are homogeneous in the image. This limits these models to working with textured surfaces that are flat and viewed from the front. In order to cope with more natural inputs, consisting of textures on curved or slanted surfaces, the segmentation method must permit distortions in textures projected onto an image.

The model of texture-based segmentation developed in this chapter is capable of handling these more general inputs. It is a variation of a model developed by Lee (1993, 1995; Lee et al., 1991, 1992), which itself is a variation on the weak membrane formalism (Blake & Zisserman, 1987; Geman & Geman, 1984; Mumford & Shah, 1985). One of the essential characteristics of Lee's coupled-membrane model is that it allows for gradients in the spatial frequency and orientation content of a texture. It is thereby able to segregate textures in an image even when those textures are distorted due to surface curvature or image projection effects. The *modified coupled-membrane* presented in this chapter preserves this flexibility, but uses a higher-level input instead of raw complex cell responses. This higher-level input is a set of measures that must be computed for the texture-based shape estimation process described in Chapter 4. Using these measures instead of the complex cell responses allows the segmentation process to be simplified, drastically reducing the number of computations necessary to segregate a visual input.

The simulation described here uses the *Graduated Nonconvexity* (GNC) algorithm by Blake (1983; Blake & Zisserman, 1986b, 1987) for minimizing the segmentation function in the model. The nonlinear successive over-relaxation (SOR) implementation that I use is a relatively simple method that performs reasonably well on serial-processing computers. Alternatives to GNC exist (e.g., Nielsen, 1997), but I did not investigate them. The weak membrane family of segmentation methods is also well-suited to parallel implementation. The modified coupled-membrane model could therefore be implemented using arrays of simple computing elements having only local interconnections. This suggests that a neural network could also serve as a substrate, and in fact several groups have pursued hardware VLSI implementations of networks for minimizing the weak membrane function (Harris et al., 1989, 1990, 1990; Koch et al., 1986; Lumsdaine et al., 1991).

Experiment Three demonstrated that the modified coupled-membrane model works well for generating approximate segmentations of inputs containing multiple regions of texture. By appropriately choosing the values of the parameters in the model, the segmentation system can be made to capture the major regions in an input.

Experiment Three also points out some limitations of the segmentation model:

1. It was not possible to find a set of values for all three parameters in the MCM model (α, λ, γ) that allowed the model to segment all inputs equally well. The best alternative was to fix two of the parameters (λ and γ), and manually adjust the third for each input image.

   This is admittedly not a good solution, especially since the goal is to model unsupervised texture segmentation. However, there is ongoing research on the topic of automatically setting the parameters in coupled-membrane types of models, and I believe it will eventually be possible to devise an automatic method for adjusting parameter α dynamically for each input image. It remains a topic for future work.

2. The MCM model failed to segment correctly one of the test cases (input image K, shown in Figure 5.9 on page 222). The result for that test case was partly correct, in that it captured all the major texture regions, but one area was partitioned by the model into too many subregions.

   This failure demonstrates that the model as formulated here has difficulty working with inputs that span a large range in the spatial-frequency spectrum. The failure for test image K occurred because one of the textures evoked responses in complex cells tuned to much higher spatial frequencies than any of the other textures in the image. It is likely that the problem could be overcome by modifying the segmentation method to allow parameter λ to be set adaptively across an image, instead of using a fixed λ value everywhere. The same kind of solution suggested for adaptively adjusting α (discussed above) may be useful for adjusting λ to cope with this problem. Research on this topic exists, but I have not attempted to incorporate such a feature in the present model. This problem, like the previous one, remains a topic for future work.

3. Computing segmentations with the coupled-membrane models (MCM and OCM) involves a heavily iterative process. Although the modified coupled-membrane model involves significantly fewer computations than the original coupled-membrane model, it still requires several thousand iterations through a set of nested loops. (See the summary of the GNC loop in Figure 5.3 on page 208.) Massively iterative processes such as this are unrealistic for a model of biological visual processing: the visual system is able to segment inputs on such short time scales that the method it uses could not involve iterating a loop several thousand times.

   It is *possible* that a parallel implementation would require fewer iterations to achieve a segmentation, by exploiting aspects of the minimization problem that are not available in a serial-updating implementation. However, the membrane model analogy of a large network of simple, locally-interconnected computing elements suggests that even a parallel implementation would have to use iteration in order to settle the network to a minimum state. A biologically-reasonable implementation would therefore have to substitute iterative settling with a process that minimizes the segmentation energy more directly.

   Thankfully, there is work on exactly this topic. In separate efforts, Harris et al. (1989), and Lumsdaine et al. (1991), have implemented the weak membrane approach in analog VLSI hardware. Their systems use networks of simple, electrically resistive elements arranged in a mesh; these resistive elements correspond to the computational units in the weak membrane analogy. The networks implement the membrane minimization process in a more direct, *continuous* fashion, avoiding the lengthy, discrete iteration loops that are necessary with the implementation discussed in this chapter. This type of implementation may therefore be a better analogy for how a biological network might implement a coupled-membrane model of segmentation.

Chapter 6

A Model of Combined Texture-Based Segmentation and Shape Estimation

The previous three chapters have presented a model of cortical complex cells, a model of texture-based shape estimation, and a model of texture-based segmentation. The final step in this research is combining these components into a system that takes the outputs of the complex-cell-like mechanisms and performs both segmentation and shape estimation. That is the topic of this chapter.

In truth, much of the work of readying the segmentation component for integration has already been done in the previous chapter. There, I described how the coupled-membrane model of segmentation can be adapted to use, as input, an intermediate result computed by the texture-based shape estimation component presented in Chapter 4. In the combined system, the segmentation results are used to guide the shape estimation component by determining in which image regions the remainder of the shape estimation process is performed.

Two tasks remain to be accomplished to bring everything together. First, the segmentation output needs to be processed further to produce a representation that is better suited to performing shape estimation. Second, the shape estimation component needs to be modified in several ways: it must detect and ignore regions lacking texture, it must detect and ignore regions corresponding to flat surfaces in the image, and it must be able to operate within the individual image regions delivered by the segmentation component.

This chapter is organized as follows. Section 6.1 summarizes additional changes to the segmentation component of Chapter 5 that are necessary for the final, combined model. Section 6.2 describes the modifications needed to the shape estimation method of Chapter 4 to enable it to work effectively within individual regions in a visual input. Section 6.3 then summarizes the entire model's architecture. In Section 6.4, I describe Experiment Four, designed to test the overall model. Finally, Section 6.5 summarizes the key points of this chapter.

6.1 Adapting the Segmentation Component

As described in Chapter 5, the segmentation portion of the overall model operates by settling a network of units into a stable configuration. This settling process is implemented in terms of minimizing a function that represents the interaction of competing goals in segmentation—the goals of growing larger regions and breaking regions into smaller pieces. The output of the process is a labeling of the boundaries between regions of texture in an input image.

Chapter 5 already described the primary modification to the segmentation component: changing the segmentation function of the original coupled-membrane model (Lee et al., 1991, 1992; Lee, 1993, 1995) in order to make it use the unnormalized average peak frequency (APF) maps produced by the shape estimation component of Chapter 4. However, the output produced by the modified coupled-membrane (MCM) segmentation method is still not ideal for the purposes of the overall model in this dissertation. The output of the model as designed in the previous chapter is a representation of the borders between texture regions. This format is awkward for performing shape estimation within the regions; what is needed instead is a representation of the *body* of each region.

This kind of representation could take the form of a tag or label associated with each location in the visual input, binding each location to a separate region. Theoretically, it should be possible to modify the coupled-membrane formalism described in Chapter 5 so that it produces a region-based representation directly. This would require undertaking the major task of reformulating the segmentation function (Shah, personal communication). For expediency in this research, I have followed a simpler course of retaining the border-based format of the MCM model, but adding a step for generating a region-based presentation based upon it. Region-based and border-based representations are essentially converses of each other, and it is straightforward to convert one to the other using common methods developed in computer vision.

Some additional cleanup operations are essential before performing shape estimation within each region. First, it is useful to inhibit processing near the region borders. Second, it is important to remove stray, unconnected border segments within region bodies. And finally, it is useful to ignore excessively small closed regions altogether, since it is not possible to perform any meaningful shape estimation if a region is too small.

These four processing steps are described in more detail in the following paragraphs.

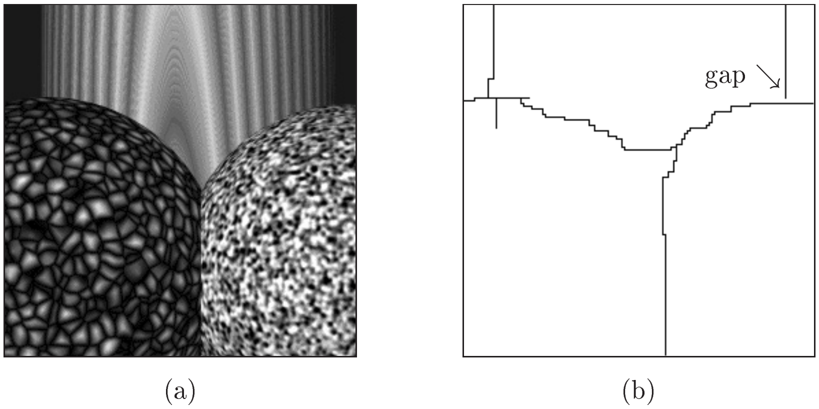
6.1.1 Inhibiting Shape Estimation Near Region Borders by Thickening the Borders

The borders between texture regions are areas where spatial-frequency content changes significantly in the image. Spatial-frequency changes are, after all, the criterion used to segment regions of texture in the visual input. These changes do not reflect variations in surface shape; they are a byproduct of differences between surfaces, or between textures on the same surface. However, the spatial-frequency changes near region borders can extend a number of pixels past the borders themselves, into the region bodies. The shape estimation process can then be misled by these changes, because it is designed around the principle that texture inhomogeneities reflect only changes in surface shape (see Section 4.2.1). The texture inhomogeneities near borders are due to other factors. To reduce the degree to which the shape estimation process is misled in these areas, it is useful to inhibit shape processing in a small radius around the region borders.

A simple way of simulating this inhibitory process is to *thicken* the borders reported by the segmentation process. This reduces the sizes of the region bodies by trimming them in those locations where spatial frequencies in the input are most likely to exhibit abrupt changes due to transitions between surfaces and textures. In other words, making the borders thicker acts to whittle away the region bodies at their extremities, at precisely the locations where spatial-frequency changes cannot be trusted to inform the system about surface shape.

There is an additional benefit to performing a border-thickening step. Sometimes, small gaps can remain in the borders between regions in the segmentation output. Figure 6.1 on the following page shows an example of this. If left alone, such gaps make it difficult to label the regions: the labeling process spills over from one region to the next. A uniform thickening process tends to close small gaps in the boundaries by growing the ends of the borders and causing them to bridge the gaps.

I implemented this inhibitory, border-thickening process by applying a binary *dilation* operation to the segmentation output produced by the MCM model. Dilation is a morphological operation commonly used in computer vision (Haralick, Sternberg, & Zhuang, 1987; Jain et al., 1995; Pratt, 1991). Intuitively, dilation corresponds to swelling or expanding a line in all directions. For instance, a line having a width of one unit would become three units wide after being subjected to the operation. Dilation is performed using a filter called a structuring element; in this model, I use a structuring element composed of a 3×3 array of one's, and perform several dilation operations in succession to gradually thicken the region borders. The number of dilation operations applied in the model is controlled by a parameter, n_d.



(a) (b)

Figure 6.1: Example of small gaps that can be left in segmentation borders. (a) The image input, previously used (with different parameter settings) as test case H in Experiment Three. (b) The output of the MCM segmentation model using parameter values $\alpha = 149$, $\lambda = 18$, and $\gamma = 0.6$. For this combination of parameters, the segmentation process leaves a small gap in one of the upper right-hand region borders. Such gaps complicate the process of finding the extents of the region bodies, because the region-labeling process (Section 6.1.2) spills across the gaps.

An additional, practical issue in the implementation must be considered at this point. The output from the MCM segmentation model (as implemented here) is represented using a supergrid having size $(2n + 1) \times (2n + 1)$, where n is the width of the APF maps. (See the discussion in Section 5.4.1.) This format must eventually be converted into a region-based representation having the same size as the APF maps, but for the border-thickening process, it is more convenient to maintain the supergrid representation. Because the eventual supergrid conversion will reduce the size of the grid by a factor of two in each direction, the number of dilation operations applied to the segmentation output at this stage must be increased to compensate for the size reduction. For all simulations described in this dissertation, $n_d = 3$.

Figure 6.2 on the next page shows an example of applying the thickening process to the border representation produced by the MCM segmentation component. The figure shows that the effect is simply a uniform expansion of the size of the borders in the MCM output. Note that stray border lines, such as the one visible in the upper left portion of the semi-circular region, are expanded along with everything else. These stray borders must be eliminated in a subsequent step; see Section 6.1.3 below.

6.1.2 Labeling Regions

Following the border-thickening process, the next step is to segregate image locations into regions based on the segmentation borders. Functionally, this is the process of marking or tagging each location in the input with a label indicating to which region it belongs.

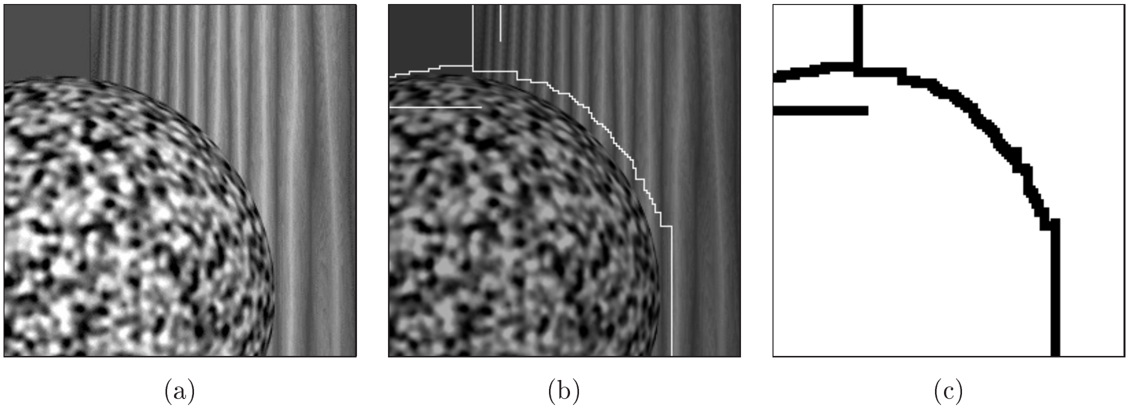


Figure 6.2: Example of applying repeated dilation operations on the output produced by the MCM segmentation method. (a) The input image, previously used as test case G in Experiment Three. (b) The segmentation result. (c) The result of successively applying a dilation operation a total of n_d times to the segmentation result shown in the middle. Note that the stray segment in the left side of the round region has also been thickened. It is removed in the filling-in process described in Section 6.1.3.

For the present simulation, the region labeling process can be implemented in a simple way, using a connected-components labeling procedure of the sort commonly used in computer vision (Jain et al., 1995; Pratt, 1991). The border-based representation produced by the MCM process is essentially a binary image: it is a topographical map in which each location is marked as either being on a border or not. Applying a basic, connected-components labeling algorithm to this representation transforms it into a region-based form in which each location is marked with its region membership. The labeling procedure I use in this model assumes 8-connected neighbors. [The term 8-*connected* refers to the definition of neighbors on a square grid; under 8-connectedness, two locations are considered to be neighbors if they share at least one corner (Jain et al., 1995).]

Figure 6.3 on the following page presents an example of the results of this labeling step. It shows how a border-based representation from the MCM segmentation process is converted into a region-based representation. Note that the regions exclude the segmentation borders; in other words, locations that were on the borders are left empty, not part of any labeled region. The border-thickening step (and parameter n_d) described above controls the width of these empty areas between segmentation regions.

Using a connected-components labeling method is a simple approach for identifying regions based on the borders output by the segmentation component. However, it does have one limitation: it requires that the border representation contain closed regions. Otherwise, if there are gaps in the borders between regions, then, as mentioned above, the labeling process will spill over from one region to the next, merging regions that

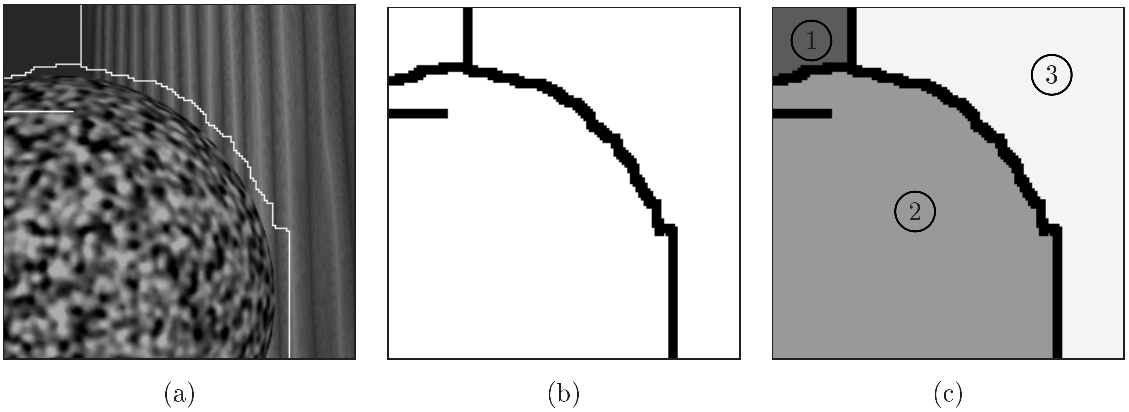


Figure 6.3: Example of applying connected-components labeling to the thickened segmentation result. (a) A segmentation result from the MCM model. (b) The segmentation result after performing the border-thickening process described in Section 6.1.1. (c) The result of performing the labeling process. (The ordering of the regions is arbitrary.)

should be left separate. The thickening process described in Section 6.1.1 helps avoid this problem, but if the gaps in a border are too large, the thickening process cannot bridge them. For this reason, it is important that the segmentation process be tuned to produce closed regions.

As noted above, I implemented the process of converting borders to regions using a labeling procedure for expediency; it is not intended as a literal model of how the brain may identify regions in the visual input. A more effective approach would be to modify the coupled-membrane formalism to produce a region-based representation directly.

6.1.3 Filling In Holes Due to Stray Border Lines

The segmentation process sometimes leaves stray, disconnected border lines within regions. This problem was already encountered with some test cases in Experiment Three; another example can be seen in Figures 6.2 and 6.3, in which a short horizontal line is present in one of the segmentation regions. Such stray border lines introduce a minor problem: the connected-components labeling step (Section 6.1.2) excludes them—the labeling process simply flows around them. This leaves empty spaces where the stray border lines were located within the bodies of regions. Although this is not a critical problem, the presence of empty spaces within an image region complicates the (later) process of estimating shape from texture.

The simplest solution is to fill in the spaces introduced by the stray border lines. For the purposes of this simulation, I implemented a filling-in process using a combination of dilation and *erosion*, another morphological image-processing operation (Haralick et al.,

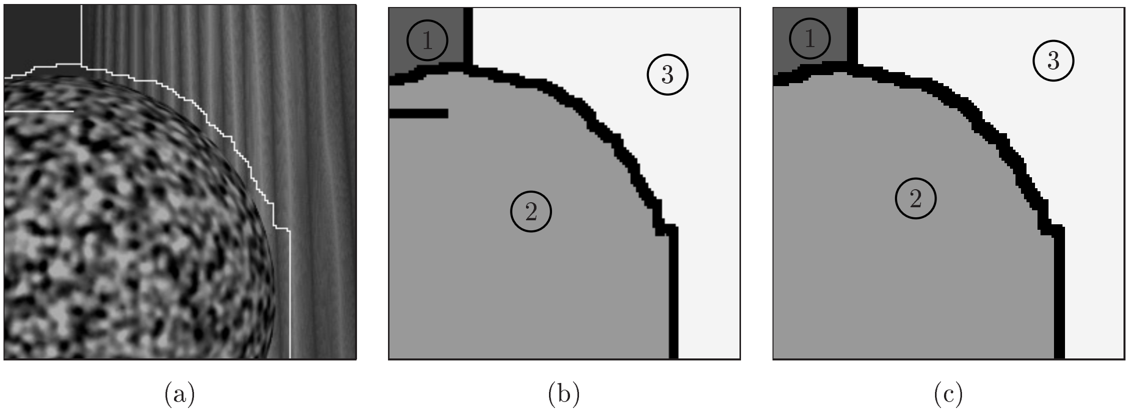


Figure 6.4: Example of performing the filling-in process. (a) A segmentation result. (b) The segmentation result after performing the labeling and small-region-removal processes. Note that borders are actually empty spaces; that is, they are considered to be untextured and lying in-between regions. The border segment in the upper left-hand portion of the semi-circular region is therefore an empty space. To avoid complicating the shape estimation process with special mechanisms for handling empty spaces within a region, it is best to fill in the spaces. (c) The result of performing the filling-in process on the representation shown in the middle image. The empty space caused by the stray border line has now been filled in.

1987; Jain et al., 1995; Pratt, 1991). Erosion is the opposite of dilation; when performed using a structuring element consisting of a 3×3 array of one's, it has the effect of shrinking a shape uniformly. The combination of dilation followed by erosion, when applied to the representation of an image region produced by the labeling step, acts to fill in small empty spaces without altering the region's overall shape. The dilation operation fills holes inside a region, but it also grows the region boundaries outward; hence the need for the erosion operation, to bring back the region boundaries.

The dilation and erosion operations must be applied to each image region separately, after excessively small regions have been removed. I implemented the process as follows. For each region, the model performs the dilation operation $n_d + 1$ times, followed by the erosion operation $n_d + 2$ times. The effect of this swelling followed by shrinking is to fill in the holes left by unclosed border lines that fall within an image region produced by segmentation. Figure 6.4 provides an example of the results of this process.

6.1.4 Detecting and Ignoring Excessively Small Image Regions

If the radius of an image region is much smaller than the values of the shape estimation model parameters r_a, r_i, and r_s, then the shape estimation process cannot produce a reasonable result within the region. For this reason, it is useful to eliminate excessively small, closed regions from the segmentation results. These areas sometimes arise near corners between larger regions; examples are visible in some of the segmentation results from Experiment Three, such as Figure 5.19 on page 233.

One approach to eliminating small regions is to count the number of locations in each closed image region, and then remove those that have fewer than a threshold number of locations. I used a fixed threshold parameter τ_r, with a value of $\tau_r = 1000$ for all simulations presented in this dissertation. (Again, this is measured on the supergrid form of the segmentation result.) If a region has fewer than τ_r locations after the labeling step described above in Section 6.1.2, it is flagged as excessively small and ignored in subsequent processing steps.

6.1.5 Summary of the Overall Segmentation Component

The addition of the four processes described above completes the segmentation aspect of the overall model. After these processes are performed on each of the image regions, the results can be subsampled by two in each direction to reduce the supergrid representation to the same size as the APF maps. This then serves as a map of region labels, indicating the region membership of each point in the input.

Figure 6.5 on the next page summarizes the segmentation component. The figure shows that the output of the MCM model from Chapter 5 is passed to the additional processes described in this section. The final output is a representation of the major regions in the input. The figure shows MCM, border inhibition and region labeling elements colored in gray, to indicate that these three processes could be collapsed into one by reformulating the segmentation function. (See the discussion at the start of this section.)

6.2 Adapting the Shape Estimation Component

As described in Chapter 4, the shape estimation portion of the overall model operates by tracking changes in spatial frequencies across a surface to infer changes in the shape of the surface. There are four stages in the model described in Chapter 4: computing complex cell responses, computing the APF value at each location across the input, estimating local slant and tilt at each location by normalizing the APF values and performing lateral inhibition, and finally, integrating the estimates of local surface slant and tilt. In the combined processing model, the first two stages (computing the complex cell responses and unnormalized APF values) are common to both the shape estimation and segmentation components. Once a segmentation of an input image is achieved, the system must finish the shape estimation process within each image region separately.

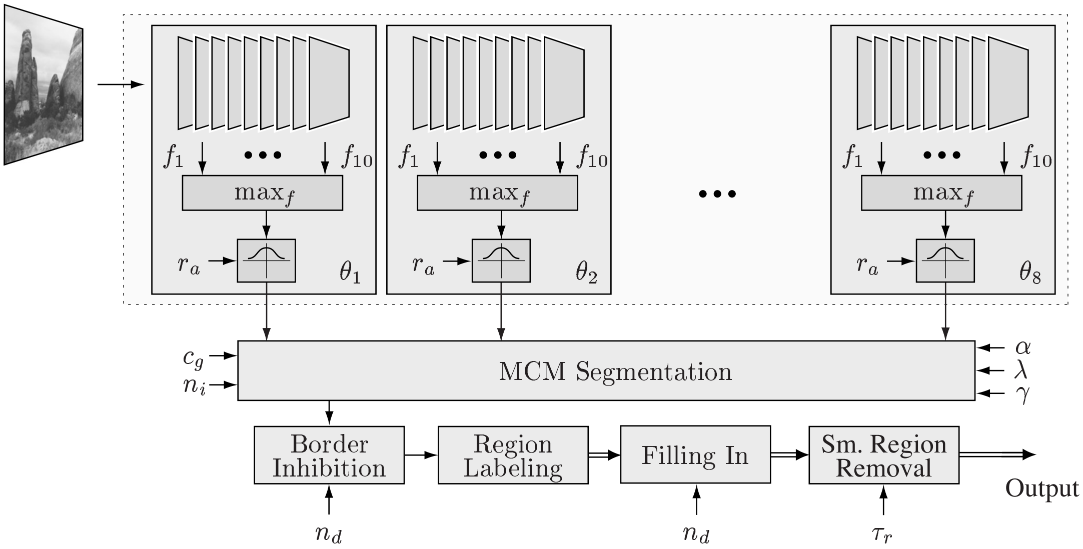


Figure 6.5: Illustration of the complete segmentation component. The segmentation process starts with the unnormalized APF maps created as part of the shape estimation process described in Chapter 4. The set of APF maps (one for each orientation implemented in the simulation) are used as inputs by the modified coupled-membrane segmentation model of Chapter 5. The output of the MCM model is further processed using the steps described in this section: inhibition near region borders using border thickening; region labeling; filling in holes introduced by stray border lines; and exclusion of excessively small regions.

The primary change required to the shape estimation model is to make the APF normalization process operate within separate regions. In addition, a few other mechanisms are also necessary. First, before even attempting the APF normalization step, the shape estimation model should detect and ignore image regions that appear to be devoid of texture. Second, the model should detect regions that appear to contain flat surfaces facing the viewer, and short-circuit the shape estimation process in those regions. And finally, the process of estimating the local direction for surface integration needs to be adjusted in order to better handle region boundaries. Each of these changes is discussed in the following paragraphs.

6.2.1 Detecting Image Regions That Lack Texture

Most natural scenes contain a mixture of textured and untextured areas. In those regions of an image that are devoid of texture, the shape estimation component cannot function properly. However, the shape estimation model described in Chapter 4, by itself, cannot *detect* the lack of texture and stop operation. A separate mechanism is required to test for this condition and prevent the shape estimation process from attempting to operate in image regions devoid of texture.

The complex cells that serve as input to this visual processing model do not respond in image regions that do not contain contrast variations. Areas that lack texture in an image are characterized by a large proportion of zero responses in the complex cell outputs, and consequently also in the APF maps created as the first step in the shape estimation process. A test for absence of texture can therefore be formulated as follows: if fewer than a certain proportion of cell responses are zero over a region, then label that region as devoid of texture and remove it from further consideration.

I implemented this idea by modifying the shape estimation model to perform the following steps within each segmentation region prior to APF normalization:

1. Count the total number of spatial locations in the region;
2. In each orientation of the APF maps:
    (a) Count the number of locations across the current region having a nonzero value in the APF map for the current orientation;
    (b) If the number of nonzero responses is greater than a factor c_t times the total number of spatial locations in the region, treat the region as containing texture and continue with shape estimation.
3. If none of the orientation components in the APF maps reach the threshold of activity, consider the region to be devoid of texture.

The parameter c_t represents the fraction of region locations that must contain active cell responses in order for the region to be considered to contain texture; it is expressed as a fraction of the total number of spatial locations in a region. For all of the simulations reported here, $c_t = 0.5$.

6.2.2 Performing APF Normalization Within Separate Regions

In Sakai and Finkel's (1994, 1995, 1996) original shape estimation model, the input is assumed to contain a single surface. After computing the average peak frequencies across the input image, the model then performs APF normalization. The goal of that process, described by Equation 4.14 on page 139, is to normalize the APF at each location with respect to the lowest average peak frequency across the entire input. By contrast, the integrated model in this chapter must handle inputs containing multiple textured surfaces. Each texture region in the input will generally lead to a separate region in the segmentation output. The normalization process in the shape estimation model therefore needs to be performed separately within each region, rather than across the entire input.

This must be performed after the segmentation component has segregated the input into regions based on the unnormalized APF values.

It is straightforward to change the APF normalization step to work within restricted regions of the input. Recall from Sections 4.2.2 and 4.3.2 that the normalization process involves applying Equation 4.14,

$$\widetilde{O}_2(x, y\,; \theta) = \frac{O_2(x, y\,; \theta)}{\min\limits_{x', y' \in I(x,y)} O_2(x', y'\,; \theta)} - 1,$$

to each point in the set of APF maps. To limit this process to separate spatial regions, all that is required is to treat points outside of the current region as being zero. Then Equation 4.14 can be written

$$\widetilde{O}_2(x, y\,; \theta) = \frac{O_2(x, y\,; \theta)}{\min\limits_{x', y' \in I(x,y\,; r)} O_2(x', y'\,; \theta)} - 1, \tag{6.1}$$

where $I(x, y\,; r)$ signifies the portion of the input image encompassed by region r. This normalization process must be repeated for each region.

In theory, normalization could be performed in parallel within each image region, given appropriate hardware or a neural network. However, as with other aspects of the simulation described in this dissertation, the normalization in my implementation is performed serially.

6.2.3 Detecting Flat Surface Areas

Textures on flat surfaces viewed from the front produce regions in the image that should be interpreted as being flat. Ideally, the shape estimation system should automatically produce a flat surface rendition for this kind of input; i.e., the proper interpretation should arise naturally in the normal course of estimating surface shape. However, in practice, real image textures are never completely homogeneous, and this complicates shape estimation in flat regions: the minor deviations from homogeneity may give the impression that there are variations in surface shape. An additional mechanism for detecting flat textures is thus required.

A simple modification to the shape estimation process can achieve this goal. Recall that the shape estimation model of Chapter 4 includes a step, in Stage Four, which finds the area of lowest normalized APF in the current image region. This is done using a thresholding step in Equation 4.25 on page 151. It happens that if an image region consists of a flat surface, then a large percentage of the region's area will fall under the threshold and end up being considered as part of the area of lowest normalized APF. This suggests a simple test for flat textures: if a large percentage of an image region's area is thresholded out during this step, tag the region as being flat.

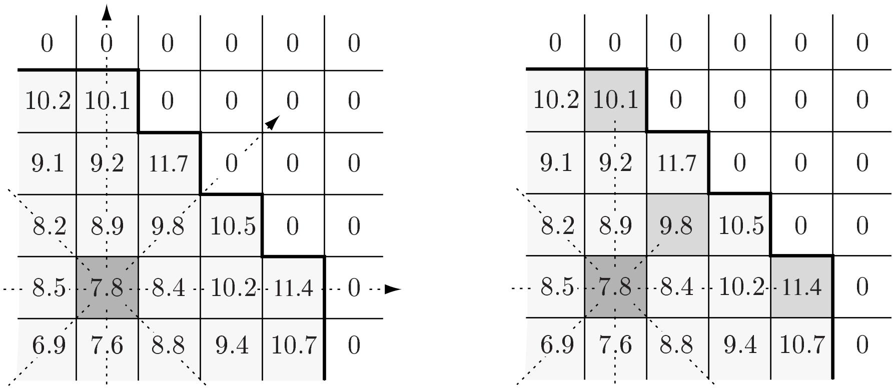
I implemented this idea using a parameter, c_f, and an additional step added to Stage Four of the shape estimation component. The additional step precedes the slant integration process. After computing the result of $T_s[\bar{O}_{3s}(x, y)]$ in the current region, the fraction of spatial locations included in the resulting area of "lowest normalized APF" is computed. If the fraction of locations is greater than c_f, the region is considered to be flat and omitted from further processing. I set c_f to the value $c_f = 0.75$ in all of the simulations presented in this dissertation.

6.2.4 Estimating Integration Directions Near Region Boundaries

The process of integrating surface slant, described in Section 4.3.4, requires computing a map of directions along which the integration should proceed. This involves a step in which APF values are summed along paths leading away from each spatial location, according to Equations 4.29 and 4.28. But this leads to a problem at region borders.

At locations near a region's borders, the distance l_p used for looking ahead in the different candidate integration directions may extend past the end of the region. This is illustrated in Figure 6.6(a) on the following page. There are no valid data available past the border of a region, and yet these border locations must nevertheless be assigned *some* value in order to allow the integration process to work. What values should be assigned to these locations?

Since there are no actual data available past a region's borders, the best one can do is to make an estimate of what the value of $\text{wssum}(x, y, r, s)$ might be if the trend in the region were to continue past the border. There are different ways of extrapolating the data; I chose a simple approach based on the last available value along a path before the border, illustrated in Figure 6.6(b) on the next page. The approach is to multiply the last value of $\bar{O}_{3s}(x, y)$ by the remainder of the distance l_p that extends past the region border. More specifically, the process computes $\text{wssum}(x, y, r, s)$ along a path as far as it can go up to the border (let us say this is a total of ${l_p}'$ locations), then uses the last data value next to the border and multiplies that value by $l_p - l'_p$.



0	0	0	0	0	0
10.2	10.1	0	0	0	0
9.1	9.2	11.7	0	0	0
8.2	8.9	9.8	10.5	0	0
8.5	7.8	8.4	10.2	11.4	0
6.9	7.6	8.8	9.4	10.7	0

(a)

0	0	0	0	0	0
10.2	10.1	0	0	0	0
9.1	9.2	11.7	0	0	0
8.2	8.9	9.8	10.5	0	0
8.5	7.8	8.4	10.2	11.4	0
6.9	7.6	8.8	9.4	10.7	0

(b)

Figure 6.6: Illustration of a problem in computing the integration direction map near region borders. Each grid represents a portion of the map of $\bar{O}_{3s}(x, y)$ for an input image, in the areas near a region border. The border is indicated by the dark solid line. (a) At locations less than distance l_p from the border, the path of l_p consecutive neighbors used for I_p in Equations 4.17 and 4.27 will extend past the border of the region. Beyond this border, all APF values are treated as being zero (signifying no complex cell activity). Unless these special, zero values are treated carefully in computing the integration directions, the directions heading towards the border will appear to have low sums in Equation 4.27. This will in turn misled the process of estimating integration directions. (b) Since there are no actual data to work with beyond the borders, the best one can do is estimate what the sum-of-neighbors might be, given the available data points along the path up to the border. The simplest approach is to estimate the sum in Equation 4.29 using the last real value along the path. Here, the three locations shaded in medium gray indicate the last usable values before the region border along three of the eight possible paths away from the darkly-shaded point. These three shaded locations correspond to (r'_l, s'_l) in Equation 6.2.

This changes Equation 4.29 on page 154 to be

$$\text{wssum}(x, y, r, s) = \left\{ \begin{array}{ll} \displaystyle\sum_{\substack{r', s' \in I_p(r,s) \\ x+r', y+s' \in I_r}} \left[W(r', s')\, \bar{O}_{3s}(x + r', y + s') \right] & \text{if } x = r \text{ or } y = s, \\ \displaystyle\sum_{\substack{r', s' \in I_p(r,s) \\ x+r', y+s' \in I_r}} \left[W(r', s')\, \text{diagval}(x, y, r', s') \right] & \text{otherwise} \end{array} \right\}$$

$$+ \left\{ \begin{array}{ll} 0 & \text{if } l'_p = l_p, \\ c_b\, (l_p - l'_p)\, \bar{O}_{3s}(x + r'_l, y + s'_l) & \text{if } l'_p \neq l_p \text{ and } \{\ x = r \text{ or } y = s\ \}, \\ c_b\, (l_p - l'_p)\, \text{diagval}(x, y, r'_l, s'_l) & \text{if } l'_p \neq l_p \text{ and } x \neq r \text{ and } y \neq s. \end{array} \right\} \tag{6.2}$$

where I_r is the set of points in the current region; (r'_l, s'_l) is the last location within the boundary of I_r along the path of $I_p(r, s)$ from (x, y); l'_p is the distance from (x, y) to (r'_l, s'_l) along the path of $I_p(r, s)$; operator *diagval* is as described in Section 4.3.4 (i.e., bilinear interpolation adapted to compute diagonal values specifically in the map of

$\bar{O}_{3s}(x, y)$ values); and c_b is a factor used to adjust the weight given to the estimation. Note that the weighting terms $W(r', s')$ are purposefully omitted when using the second case. The reason is that, for short path distances close to a border, the terms in the inverted Gaussian $W(r, s)$ have low magnitudes; if they are left in the equation, they counteract the purpose of estimating the path sum using $l_p - l'_p$.

The factor c_b is useful for fine-tuning the behavior of the mechanisms for estimating wssum(x, y, r, s) near region borders. Higher values tend to create a repulsive force near the boundaries, such that for locations near a region border, the system is more likely to choose an integration direction that does not cross the border. However, if the value of c_b is too high, the repulsive force is too strong and all directions end up being forced away from the boundaries, leading to inaccurate shape estimates. Values of c_b greater than one and less than two are most appropriate for the combination of parameters used in the implementation of the model in this dissertation. For all the simulations described in this chapter, I used a setting of $c_b = 1.1$.

6.2.5 Summary

The addition of the four modifications described above completes the shape estimation aspect of the overall model developed in this dissertation. Figure 6.7 on the next page summarizes the final design of this component. The figure shows the data paths and operations for one image region; each region delivered by the segmentation component (Section 6.1) is processed in an identical fashion using the processing sequence illustrated in the diagram.

6.3 The Architecture of the Combined Model

The different pieces discussed above and in previous chapters can finally be brought together to compose a model of combined texture-based segmentation and shape estimation. This section summarizes the complete model, beginning with a general outline and then moving on to a more detailed summary.

6.3.1 Overview of the Combined Model

The operational goal of this model is to accept images of scenes containing multiple textured surfaces, segment the inputs into regions representing the major areas of texture, and estimate the shapes of the textured surfaces. In outline, the model achieves this as follows:

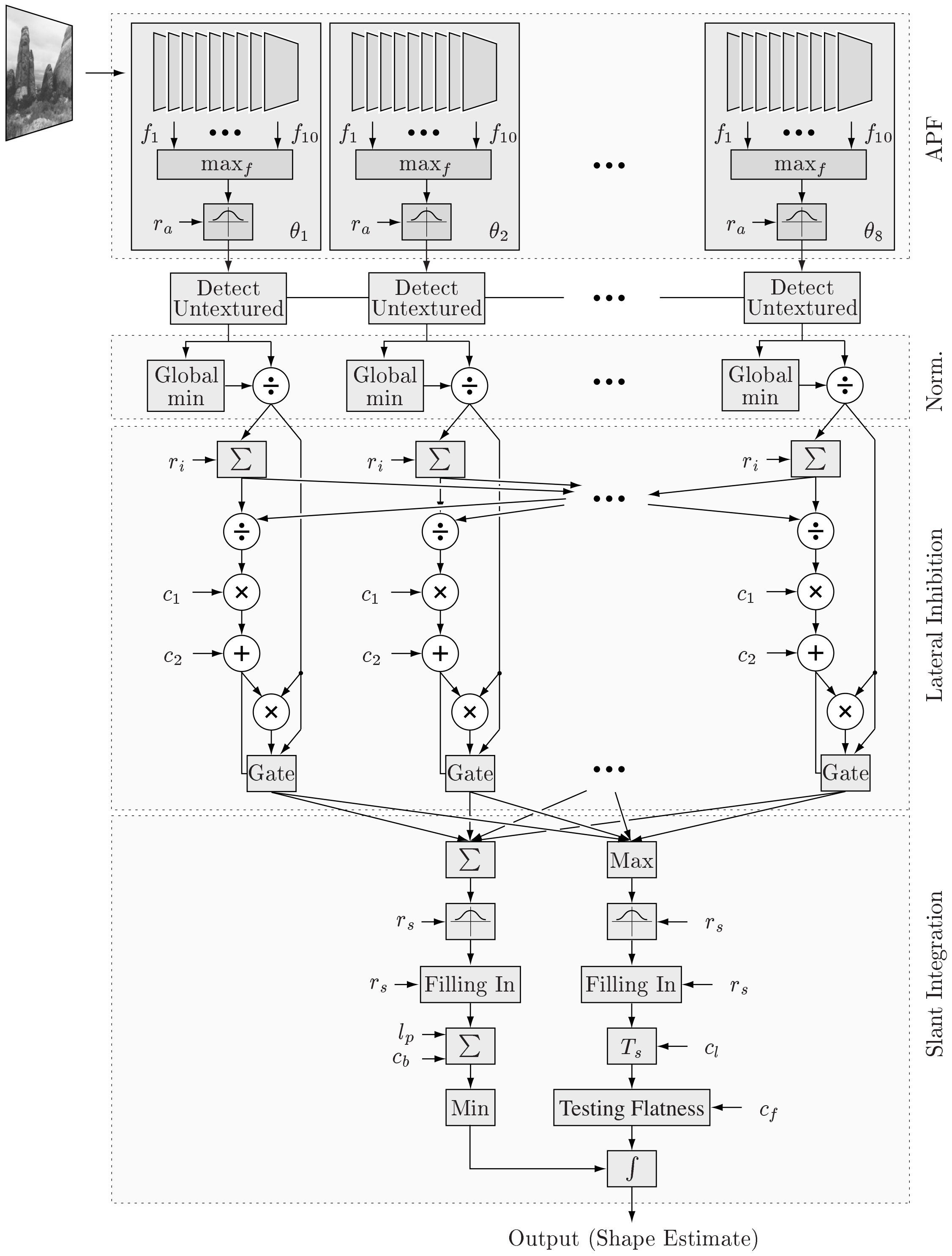


Figure 6.7: Illustration of the complete shape estimation component, based on the model previously illustrated in Figure 4.11 on page 156. Shown here is the processing sequence for a single image region; each region found by the segmentation component is processed using the same sequence of computations depicted above.

1. A collection of visual detectors, modeled after cortical complex cells, analyzes the input image and produces a set of signals. These signals indicate how well the visual input matches the stimulus qualities preferred by the complex cells at each visual location. The collection of cell responses is used to estimate the spatial-frequency properties of the input. A further computational step is used to estimate the average of the dominant spatial frequencies in small neighborhoods everywhere across the visual image. This set of APF measurements then serves as the input to both the shape estimation component and the segmentation component.

2. The segmentation component takes the set of APF measurements and analyzes it for patterns that indicate abrupt changes in the spatial-frequency properties of the visual input. These abrupt changes are assumed to represent changes in the characteristics of the surfaces in the visual image. The segmentation component uses this to infer where there are regions of similar-looking textures. It segregates these regions and labels them, producing an output that is usable in itself (one of the two main outputs produced by the overall model), and that also serves as one of the inputs to the shape estimation component.

3. The segmentation output and the APF values are used as the inputs to the texture-based shape estimation component. The shape estimation model uses the pattern of APF values within each segmented region to infer the three-dimensional shape of the textured surface that gave rise to the region. The output of this component (and the final output from the overall model) is a representation of the slant and tilt at each location within each region of the input image.

The next section summarizes the model in more detail, focusing on the computational processes behind each of the three main steps listed above.

6.3.2 Summary of the Computational Steps in the Combined Model

Figure 6.8 on page 264 provides a diagram of the overall architecture and the sequence of processing. The analysis of a visual input begins with the set of simulated complex cells. The simulation is designed to approximate the general stimulus-response properties of complex cells in cortical area V1 of the brain using the model presented in Chapter 3. Each complex cell responds best to stimuli containing a specific range of spatial frequencies and orientations. Its response can therefore be taken as an estimate of whether the stimulus falling within its receptive field contains those frequencies and orientations. The output of this computational stage is a set of topographically-organized response maps. Each

(x, y) location in a map corresponds to a spatial location in the input image. There is one map for each of the 80 possible combinations of preferred frequencies and orientation angles implemented in the model.

The complex cell response maps are used to estimate the average of the dominant spatial frequency at each spatial location in the visual input. This computation involves finding the spatial frequency of the strongest-responding complex cell at each location, then averaging the frequency values in each small neighborhood of radius r_a. Thus, strengths of complex cell responses are translated into averaged spatial frequencies in this step. The computations can be performed in parallel at every location. The dimensionality of the data is reduced from four dimensions to three, and the resultant values are represented in terms of one APF map for each orientation angle θ implemented in the model. The outputs from this APF calculation step are then diverted to both the segmentation and the shape estimation components.

The segmentation component uses a modified coupled-membrane model which simultaneous examines the entire set of APF values, locating abrupt changes in the landscape of values while ignoring gradual changes. The segmentation model operates by iteratively minimizing a function that represents the competing goals in segmentation: growing larger regions versus creating boundaries. When the function reaches a best-fit configuration, its state can be "read out" to produce a labeling of the abrupt transitions in the set of APF values—in other words, region borders. The entire process is under the control of five parameters: α, λ, γ, c_g, and n_i. The first three parameters control the likelihood that the system will put a segmentation border at any given location, and the last two parameters control how long the iterative settling process runs before stopping.

The boundaries between texture regions are areas where the shape estimation process can be misled by changes in spatial frequencies arising from region transitions. To help avoid errors at the region boundaries, and to help close small gaps that may remain in the borders, the segmentation process uniformly thickens the region borders. This has the effect of inhibiting the shape estimation process in a small radius everywhere near the borders. The width of inhibition is controlled by parameter n_d.

The border-based representation is then converted into a region-based representation using a labeling step. This step tags each spatial location in the input image with a label indicating to which segmentation region the location belongs. Following the conversion to a region-based representation, the segmentation system flags excessively small regions so that they can be ignored from further processing. The threshold size of regions that are considered "too small" is controlled by parameter τ_r. After this, the final step in the segmentation component is the process of filling in any holes inside the bodies of

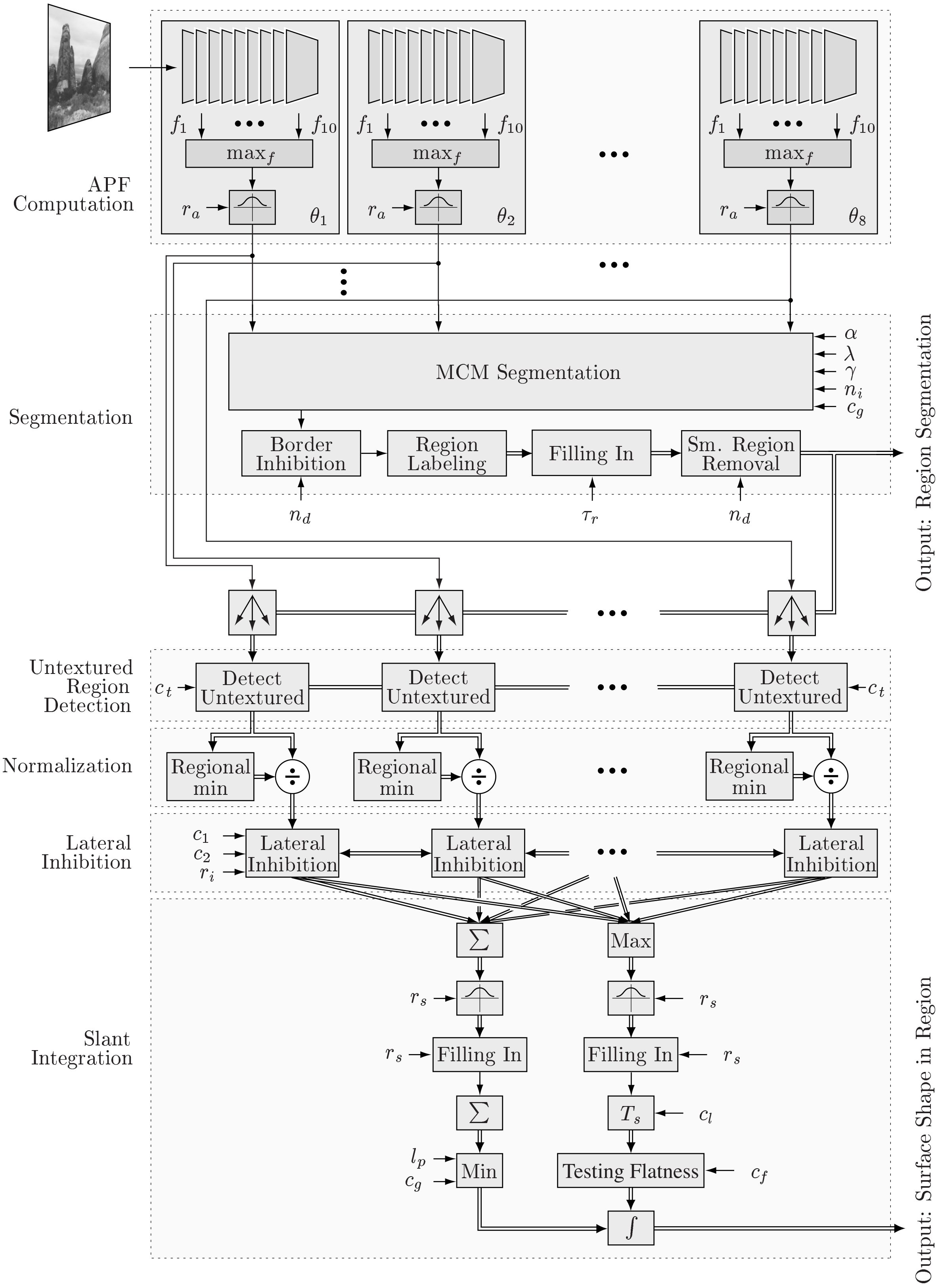


Figure 6.8: Illustration of the overall model.

the remaining regions. The amount of filling-in is controlled by parameter n_d. The final output of the segmentation component is a representation of the major regions in the visual input.

The output of the segmentation process is used to guide the third major component in the overall model: texture-based shape estimation. The estimation of surface shape from texture proceeds separately within each segmentation region. In Figure 6.8, this separate, parallel processing is indicated by the doubled lines. The shape estimation component first tests a region to evaluate whether it appears to contain texture, based on the proportion of locations having nonzero APF values (and thus, nonzero complex cell outputs). The threshold of activity needed for a region to be considered textured is controlled by parameter c_t. If a region appears to contain texture, the next step in the shape estimation process is APF normalization within the region. The result of normalization is an estimate, within each orientation, of the surface slant at each location. Following this, a lateral inhibition process is used to estimate the tilt direction at each location, using the principles laid out in Chapter 4. This part of the shape estimation model is controlled by parameters r_i, c_1, and c_2.

Next, the results of these computations are evaluated to infer whether the surface shape in the region appears to be flat. If it is, there is no point in continuing with shape estimation, so the process is short-circuited and the region is labeled as being flat. Parameter c_f determines the percentage of a region that must appear flat before the entire region is considered unsuitable for continued analysis. If a region is not flat, the final step is performed: integrating the local slant and tilt values to produce an estimate of the overall surface shape within the region. The behavior of this final process is controlled by parameters r_s, c_l, and n_s.

The output of the shape estimation component is in the form of a three-dimensional rendition that represents the model's estimate of the shape of the surface in each region.

6.3.3 Parameters in the Model

Table 6.1 on the next page provides a summary of the different parameters in the model, grouped by the component to which each belongs. The table lists all parameters that are not fixed by a design constraint or other consideration; in other words, all the listed parameters are in some sense free to be varied to tune the behavior of the model on a set of input images.

The number of parameters in Table 6.1 is certainly daunting, and may even seem excessive—after all, given enough parameters, one can fit any set of data. In defense of this large number of parameters, I offer the following observations.

Table 6.1: Parameters in the combined model of segmentation and shape estimation

	Symbol	Description	Section
C.C.	τ_c	Absolute threshold for minimum value of complex cell responses; lower response values are set to zero	3.3.3
Shape Estimation Component	c_1	Scaling factor used in computing lateral inhibition between orientations in Stage Three of the shape estimation process	4.3.3
	c_2	Offset constant used in computing lateral inhibition between orientations in Stage Three of the shape estimation process	4.3.3
	c_b	Factor used to adjust the weight given to estimates of integration directions that cross region boundaries	6.2.4
	c_l	Dimensionless factor used to set threshold for finding the region of lowest normalized APF in $\bar{O}_{3m}(x, y)$	4.3.4
	c_t	Minimum fraction of active cell responses in a region needed for the region to be considered to contain texture	6.2.1
	c_f	Maximum fraction of a region that can be thresholded by $T_s[\bar{O}_{3s}(x, y)]$ before the region is considered flat	6.2.3
	l_p	Length of path used for lookahead in computing the integration direction map in Stage Four of the shape estimation process	4.3.4
	n_s	Number of iterations for which the surface shape integration computation is performed	4.2.2
	r_a	Radius over which complex cell responses are averaged to create APF maps	4.3.2
	r_i	Radius over which normalized APF values are averaged during lateral inhibition in Stage Three of the shape estimation process	4.3.3
	r_s	Radius over which $\bar{O}_{3s}(x, y)$ and $\bar{O}_{3m}(x, y)$ is averaged prior to slant integration in Stage Four of the shape estimation process	4.3.4
Segmentation Component	α	Cost of creating a segmentation break between any two given points	5.2.2
	λ	Characteristic distance affecting interactions between discontinuities across space in the set of APF values used in segmentation	5.2.2
	γ	Strength of coupling of the segmentation process across orientations in the APF maps	5.2.2
	c_g	Constant used to calculate the stopping threshold on successive nodal values in the inner loop of GNC/SOR minimization	5.3.2
	n_i	Maximum number of inner loop iterations in GNC/SOR minimization	5.3.2
	n_d	Number of times a dilation operation is applied during thickening of segmentation borders	6.1.3
	τ_r	Absolute threshold on the minimum size of a texture region	6.1.4

C.C.: complex cell component.

- Although there are few *a priori* constraints on the possible values for the parameters, not all values are valid or useful in practice—there are practical constraints. For example, the complex cell output threshold, τ_c, has a useful range from zero to a small fraction of the mean value of the complex cell responses. It is not useful to set τ_c to a higher value, because then too many complex cell outputs fail to reach the threshold and are suppressed. A similar situation holds for most of the other parameters in the table: there is not an infinite amount of freedom in setting their values.

- It is almost certainly the case that biological systems have even *more* parameters acting on each processing mechanism. Consider, for example, that every neuron and neural network in the brain is subject to a large number of influences such as hormones and modulatory neurotransmitters (Shepherd, 1994). These are not modeled here; if they were, they would surely add several more parameters to each subprocess. Thus, compared to a neural system, the list of parameters in Table 6.1 is likely to be short.

These points notwithstanding, the large number of free parameters in the model does reflect a need for further research on how to set their values automatically. Many brain mechanisms, such as that of contrast adaptation in simple and complex cells of cortical area V1, act to set parameters dynamically based on the current visual input. This kind of feedback mechanism would be ideal for controlling the values of the parameters in Table 6.1. However, the addition of such feedback mechanisms will have to await further development of this model.

Section 6.4.1 discusses the specific values given to these parameters during Experiment Four, and the method I used to arrive at the values.

6.3.4 Constraints on the Types of Input Images

One of the original goals for this dissertation was to develop a model that could perform both segmentation and shape estimation on general images of natural scenes, such as the image in Figure 1.1 in the introductory chapter. However, this goal ultimately proved too difficult, and I settled on a more modest goal of handling restricted types of inputs. The main restrictions have already been discussed at different points in previous chapters and in Experiments Two and Three, but it is worth reiterating them here.

The combined segmentation and shape estimation model can accept images containing a mixture of textured and untextured surfaces, but the images must satisfy the following constraints:

1. *Orthographic image projection.* The shape estimation component assumes input images that are created by orthographic image projection. It cannot make use of perspective cues, and in fact, will be misled by the presence of perspective in an image. The assumption of orthographic projection limits the information that the approach can extract from an image. The model can only infer changes in shape from texture; it cannot judge relative distances in a scene, nor the relative positions of multiple surfaces. It is also subject to the tilt ambiguity: the question of whether a surface is concave or convex cannot be resolved based solely on the information in the image. As explained in Section 4.1.1, for this research I built into the model the assumption that all viewed surfaces are convex.

   The segmentation component does not require orthographic projection, and can function with images containing perspective effects. As already mentioned, recent work by Sakai and Finkel (1997) shows that the type of shape estimation model used here can be extended to work with perspective projection. The use of orthographic projection in the present work is therefore only a convenient simplification, and not a fundamental limitation.

2. *Texture homogeneity.* The separate textured surfaces in a scene must each be covered with homogeneous or nearly homogeneous surface textures. This is especial important for the shape estimation component, because it relies on directly measuring the deviations from homogeneity in an image that are caused by changes in surface shape and orientation with respect to the viewer. The type of texture homogeneity that I assume in this work is that textures are statistically free from systematic trends. Strictly speaking, only regular textures (a grid, or an array of dots) are truly homogeneous, but even textures created by a noise-like process can be sufficiently homogeneous if the variations in the pattern are on a small scale. For example, the filtered random noise textures used in images in the previous chapters satisfy these criteria.

   Texture homogeneity is also important for the segmentation process, although less so than for the shape estimation component. For segmentation, it is enough that surface textures be devoid of significant inhomogeneities, such as whorls or knots. Such features can be misinterpreted by the segmentation process as representing region boundaries; therefore, for the system to work properly they must be absent from the input.

3. *Texture visibility.* The texture on a surface must be visible everywhere on the surface to permit the shape estimation component to measure changes in texture

compression. No surprisingly, if the texture on a surface cannot be seen for any reason, such as due to loss of focus or being too fine for the image sampling rate, then the model cannot measure the changes in the image texture and will fail to operate properly. This issue was brought out in Experiment Two, in which the shape estimation process failed for one of the test cases because the surface appeared smooth at the borders.

4. *Sufficient region size.* Regions which are small in the image are difficult or impossible to serve as a basis for shape estimation by the shape estimation component. The radii of image regions need to be large compared to parameters such as r_a, r_i, r_s, and l_p.

5. *Moderate range of spatial frequencies.* The segmentation component can sometimes fail to produce an adequate result if the textures in an image have large disparities in spatial frequencies. This was made clear by one of the test cases in Experiment Three: the test image had a region of texture that produced responses at spatial frequencies several times higher than those over other regions in the image. This made it impossible to find a set of parameter settings that both avoided producing too many segmentation breaks across the fine texture and allowed the other texture boundaries to be detected.

   As mentioned in Section 5.5, there is published research on the topic of adjusting the segmentation parameters adaptively across an image. I have not implemented this kind of mechanism in the present model, but it may offer a solution to this problem in the future, thereby removing a limitation on the kinds of inputs that this model can handle.

6.4 Experiment Four: Combined Texture-Based Segmentation and Shape Estimation

This section presents the results of experimental testing of the model of combined segmentation and shape estimation. Since the main components have already been tested separately, the present experiment mainly focuses on the additional mechanisms developed in this chapter, as well as some of the general properties of the overall model.

For the purposes of this experiment, I formulated three predictions about the expected behavior of the simulation:

1. The segmentation component, augmented with the additional mechanisms described in Section 6.1, should be able to produce a qualitatively correct representation of

the major regions in each input image. This representation should be free of small extraneous regions and stray border lines.

2. The shape estimation component, augmented with the additional mechanisms described in Section 6.2, should ignore regions devoid of texture and regions that appear to contain flat surfaces in each input image.

3. The shape estimation component should produce qualitatively correct shape estimates for the regions of non-flat texture in each input image.

The goals for the quality of the results remain the same as in previous chapters; specifically, the segmentations and the estimated surface shapes should be at least *approximately* correct. There should be clear similarities between the simulation results and the interpretations of the images that a human observer would make, but the results need not be quantitatively exact.

6.4.1 Methods

I implemented the simulation of the model as an addition to the models of shape estimation and segmentation described in the previous two chapters. As before, the implementation consisted of a mixture of code written in the language C (Kernighan & Ritchie, 1988) and the Matlab environment (MathWorks, 1998a).

All inputs were 256-level gray-scale images 512 × 512 pixels in size. The images used in this experiment are shown for reference in Figure 4.12 on page 158. Each image depicts multiple textured surfaces viewed under orthographic image projection. Image A is a photograph of a portion of a cantaloupe; the photograph was taken using the digital camera setup described in Section 4.4.1. Images B, C, D and F were generated in software using a ray-tracing rendering package (POV-Team™, 1997). The textures in image B are combinations of sine waves; the textures on the surfaces in the other images are filtered random noise, artificial wood, and (in the case of the left-most spherical shape in D) random tessellations. Image E is another photograph, this time of racquet balls covered with stick-on dots; it was acquired using the digital camera setup mentioned above. Some of these images were previously used in Experiment Three.

Running the Simulation

Running the simulation involved a three-step process for each input image: first, generating APF maps; second, running the segmentation component; and third, running the shape estimation component.

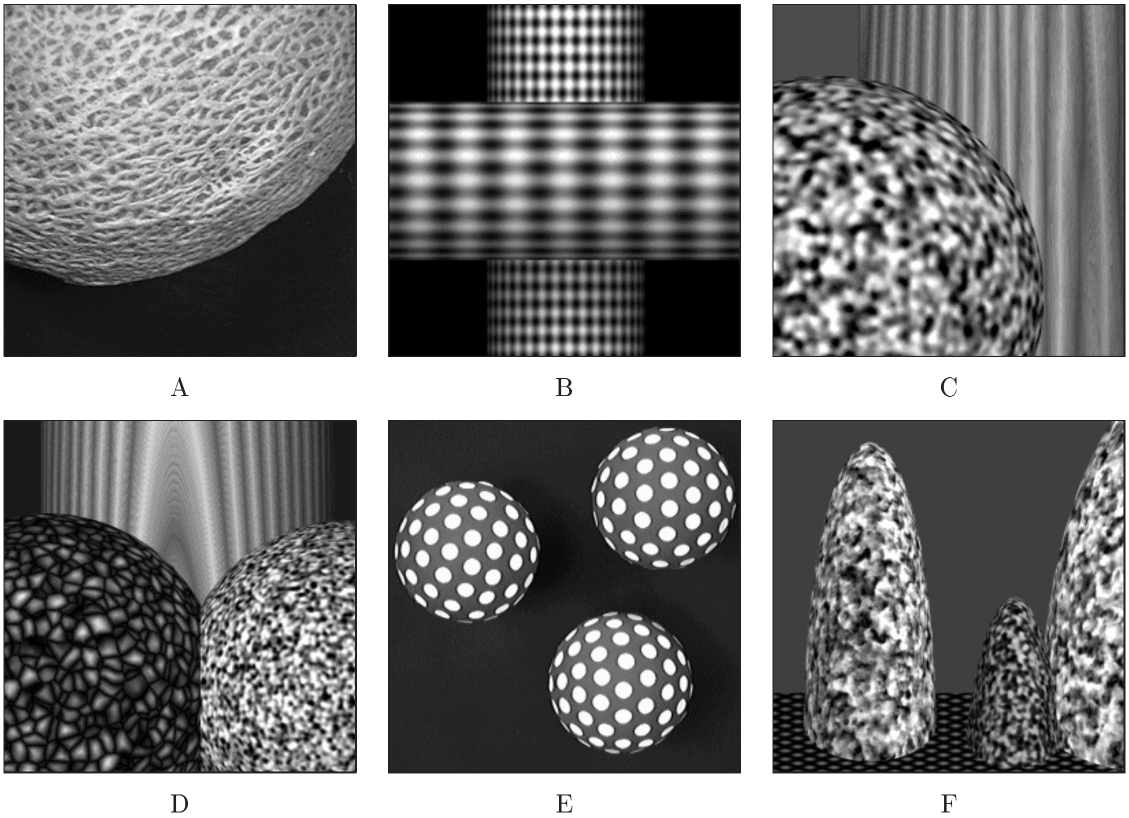


Figure 6.9: Images used as inputs for Experiment Four. More information about the images and how they were created is provided in the Methods section.

In the first step, I generated unnormalized APF maps for each image using the approach described in Chapter 4 (i.e., using Equation 4.12 on page 137). This produced a total of eight unnormalized APF maps for each input image, one map per complex cell orientation. I stored these data in hard disk files in preparation for the next phase of the simulation.

In the second step of the simulation, I ran the segmentation component described in Chapter 5 to produce a segmentation for each input image. As explained in that chapter, in its current form this component is supervised—even though most parameters can be fixed for all inputs, the threshold parameter α must be adjusted manually for each input image. I ran the simulation system for each image beginning with a general value of α, examined the results, then repeatedly adjusted α and ran the simulation again, searching for a parameter value that produced an acceptable segmentation. I chose the output that contained a minimum of extraneous regions and stray border lines, and in which the segmentation boundaries were closest to the true borders of the regions in the input image. Once a result was obtained this way, I ran the additional processing steps described in

Section 6.1 to produce a region-based segmentation. For convenience, I stored the final segmentation results in hard disk files in preparation for the third step.

In the third step of the simulation, I ran the shape estimation component with the additional processing mechanisms described in Section 6.2 above. The output from this part comprised the final output of the simulation for each input image.

The complete simulation produces four types of outputs: (1) the basic segmentation output from the MCM model; (2) a representation of the regions in the input produced by the segmentation post-processing mechanisms described in Section 6.1; (3) a tilt direction map that also includes information about which regions lack texture and which are flat; and (4) a three-dimensional representation of the estimated surface shape in each region.

Setting Parameter Values

One of the lessons from the previous two experiments has been the following. The relatively large parameter values (e.g., for the various averaging radii) previously used in the different aspects of the model work well when an input contains large surfaces, but can lead the system to produce poor results when the input contains smaller surfaces. In an effort to improve the model's performance on images containing smaller surfaces, I sought to reduce some of the parameter values, specifically the averaging radii r_a, r_i and r_s, and the value of λ. (See Table 6.1 on page 266 for a summary of all the parameters.)

I have already noted the difficulty of setting the parameters controlling the individual components in this model. Combining the major components and then attempting to reset their parameters in unison is an even more difficult chore. Since many of the parameters in the model interact, one cannot in general adjust a single value in isolation. It is necessary to use a process of trial-and-error, setting one parameter at a time to a new value, then attempting to produce the best results possible by varying the other parameters while keeping the first one fixed.

Three characteristics of the model simplify the task slightly. First, the only parameters that affect both the shape estimation and segmentation components are r_a and τ_c. I left τ_c unchanged, which meant that only changes in r_a required testing the effects on both components. Second, the segmentation and shape estimation processes operate somewhat independently, so that when changing r_a, it is possible to test the effects on each component separately. Finally, some guidance is available from the previous parameter settings: once workable parameter values have been found for the individual components (from the previous three experiments), making adjustments in the neighborhood of the values is less likely to produce radical changes in the overall model's performance than if values were chosen without any guidance at all.

Table 6.2: Default parameter values used for simulations reported in this section

Parameter	Value	Parameter	Value
τ_c	0.001	r_i	9
c_1	3.7	r_s	15
c_2	-0.1	α	(variable)
c_b	1.1	λ	16
c_l	1.37	γ	0.55
c_t	0.5	c_g	0.00005
c_f	0.75	n_i	800
l_p	37	n_d	3
n_s	140	τ_r	1000
r_a	35		

I began by reducing the APF averaging radius, r_a. The original value used in Chapter 4 was $r_a = 40$; I examined new values of 37, 35 and 32. Changing this parameter required generating new APF maps for all inputs, then testing both the segmentation and shape estimation components while varying their other parameters, all in order to study the impact of the change in r_a. After considerable experimentation, I settled on $r_a = 35$ as producing the best compromise in output quality in images containing medium-to-small surfaces as well as images containing large surfaces.

With the new value of r_a settled, I proceed to search for a new, lower value for parameter λ in the segmentation component. Lowering λ tends to produce segmentations that track region borders more closely, at the expense of increasing the likelihood of introducing extraneous borders. I tested the results of performing segmentations on a number of input images, including all six of the test cases in this experiment, and settled on $\lambda = 16$ as a value that led to acceptable results on most inputs. Parameter γ also required a small change (from 18 to 16) in order to produce the best results.

Once this was accomplished, I searched for new, lower values of r_i and r_s for the shape estimation model, again using as inputs all of the six test images in this experiment and a few additional images not shown. Changing these two parameter and that of r_a also required adjusting several other parameters in the shape estimation model.

The final values for the parameters are shown in Table 6.2. Unless otherwise noted, these were the settings used for all cases in this experiment.

Evaluating the Results

In the experiment described here (and indeed, in all of the experiments described in this dissertation), my approach to evaluating the results is subjective and qualitative. A more objective and quantitative approach would be preferable; ideally, there should be a metric for measuring the quality of the segmentation and shape estimation results.

Developing a metric for the quality of the segmentation outputs is difficult. Published research on weak/coupled-membrane approaches generally has not used precise metrics for evaluating the quality of the results, relying instead on the type of subjective, qualitative approach used here (e.g., Blake & Zisserman, 1987; Lee et al., 1992; Lee, 1995; Lumsdaine et al., 1991). I have followed the standard practice of the field in this respect.

One possible reason for the lack of widely accepted evaluation methods is that it is difficult to automatically generate "correct" segmentations for real images. Instead, it is often necessary to compare simulation results to segmentations produced by humans. Unfortunately, the humans involved are often the experimenters themselves, which is problematic because of the bias inherent in having researchers evaluate the outcomes of their own simulations. One approach to solving this problem may be to borrow techniques from psychophysics. For example, one could present a set of trained human judges with a list of criteria for a good approximate segmentation, then ask them to rate the outputs from a simulation, and finally take the averages of their ratings. This would provide a quantitative measure of goodness obtained in a manner that is more objective than relying on an experimenter's own ratings, and it would allow the outputs from different models to be compared. This approach would also be better than using a rating derived from, say, the differences in the number of correctly classified pixels between a model's outputs and templates of the desired results. The reason is that a measure based on pixel differences tends to obscure other factors that contribute to the quality of a segmentation, such as the smoothness of the region borders.

In the case of shape estimation, many of the input images used here are artificial, and the exact shapes of the surfaces in the images are known in advance. Therefore, it should be possible to measure the deviations of the model's estimates from the true shapes of the surfaces and thus generate a quality-of-match rating, perhaps in terms of least-squares differences between the model output and the actual surface. This could also be used to compare the results of this model to the results of other models.

Both of these approaches for more objective and quantitative evaluation of the results are viable. I did not pursue them here, but ideally, they should be adopted for future research in this subject area.

6.4.2 Results

For all cases, I examined the outputs of the simulation visually and evaluated how well they met the three predictions listed above. The results are presented and discussed in the following paragraphs. For each case, I provide a figure showing the outputs from different portions of the simulation. The format of the figure is as follows:

- *MCM model segmentation output.* The plot in the right-hand column in the first row of each figure presents the output from the modified coupled-membrane model. This is the output produced by the component described in Chapter 5. The format is the same as that used in Experiment Three, namely, white lines superimposed on a dimmed version of the original image.

- *Region labels.* The output produced by the additional segmentation processes described in Section 6.1.5 is shown in the left-hand column in the middle row. This plot shows the regions corresponding to the segmentations produced by the MCM component; each region is shown in a different shade of gray, similar to the example in Figure 6.4 on page 253.

- *Tilt direction map.* The right-hand column in the middle row presents the tilt direction map created by the shape estimation component described in Chapter 4. The format of the plot is an augmented version of that used in Experiment Two. It consists of an arrow diagram representing the estimated direction of tilt at each location of the input image, and in addition, includes symbols indicating regions devoid of texture (marked with ×) and regions with flat surfaces (marked with □). The orientation granularity of the tilt direction map is 45°, and to make the maps legible in the figures below, they have been subsampled from size 128×128 to 32×32 prior to plotting.

- *Surface plots of the results of surface integration.* The bottom row of each figure presents two views of the three-dimensional rendition of the surface computed by the shape estimation process. This is the same format as was used to present the shape estimates in Experiment Two in Chapter 4. In these plots, depth is relative across the surface, and the viewpoint is above the surface looking down at it.

As discussed in Chapter 4, the use of a three-dimensional reconstruction is simply a device for expressing the results of the shape estimation process. The actual representation used by the brain for surface shape estimates is unknown, and probably does not involve explicit reconstruction of this sort. It is more likely that the brain uses a distributed representation of slant and tilt.

Note that because the parameters in the model have been changed from their values in previous chapters, the segmentation results on some of the images previously used in Experiment Three are slightly different here.

Results for Test Image A

Figure 6.10 on the next page presents the results of the simulation for the first test input. The input image is a photograph of a portion of a cantaloupe. I chose this test case in part because it contains a single large surface. As mentioned above, in this experiment I lowered some of the parameter values in order to improve the simulation's ability to handle smaller surfaces; therefore, it is a good idea to test whether the model can still correctly handle large surfaces.

The results for this case generally agree with expectations. First consider the segmentation results. The output from the MCM model, shown in the upper right-hand corner, tracks the border of the surface quite well. In fact, the estimated border is surprisingly close to the true border in the image—better than in some of the results of the previous chapter. The model also correctly located the area that lacks texture in the scene.

The tilt direction map in the second row is fairly good, although it is rather noisy: the direction arrows do not radiate outward smoothly from one area, but switch back and forth across the surface. To some extent, this can be expected from the roughness of the surface texture; these inhomogeneities are bound to confuse the shape estimation process to some degree.

The shape reconstruction shown in the bottom of the figure is roughly correct, though it is far from perfect. The reconstructed surface slants away from the upper part of the image in a semi-spherical manner, so in this respect, it matches expectations. However, the shape is peaked in a way that does not agree well with human perception of the surface in the input image. The pointed quality stems from the fact that the area of lowest normalized APF in the input (and thus the area interpreted as being closest to the viewer) was estimated to be a small area in the upper left-hand corner. It is actually difficult for a human observer to judge where the closest point to the observer really is in the input image, and one cannot rule out the possibility that it is in fact the upper left-hand corner. So in that respect, the simulation output may be a valid interpretation of the surface shape. Nevertheless, the result could stand improvement. Had the area of lowest APF encompassed more of the upper portion of the image, the results might have been more agreeable. This problem may have been caused in part by edge effects at the image border.

On balance, then, this test case confirms the three predictions: the segmentation component produced a qualitatively correct output, the shape estimation component found and ignored areas devoid of texture in the image, and the shape estimation output is a valid (though not ideal) interpretation of the input.

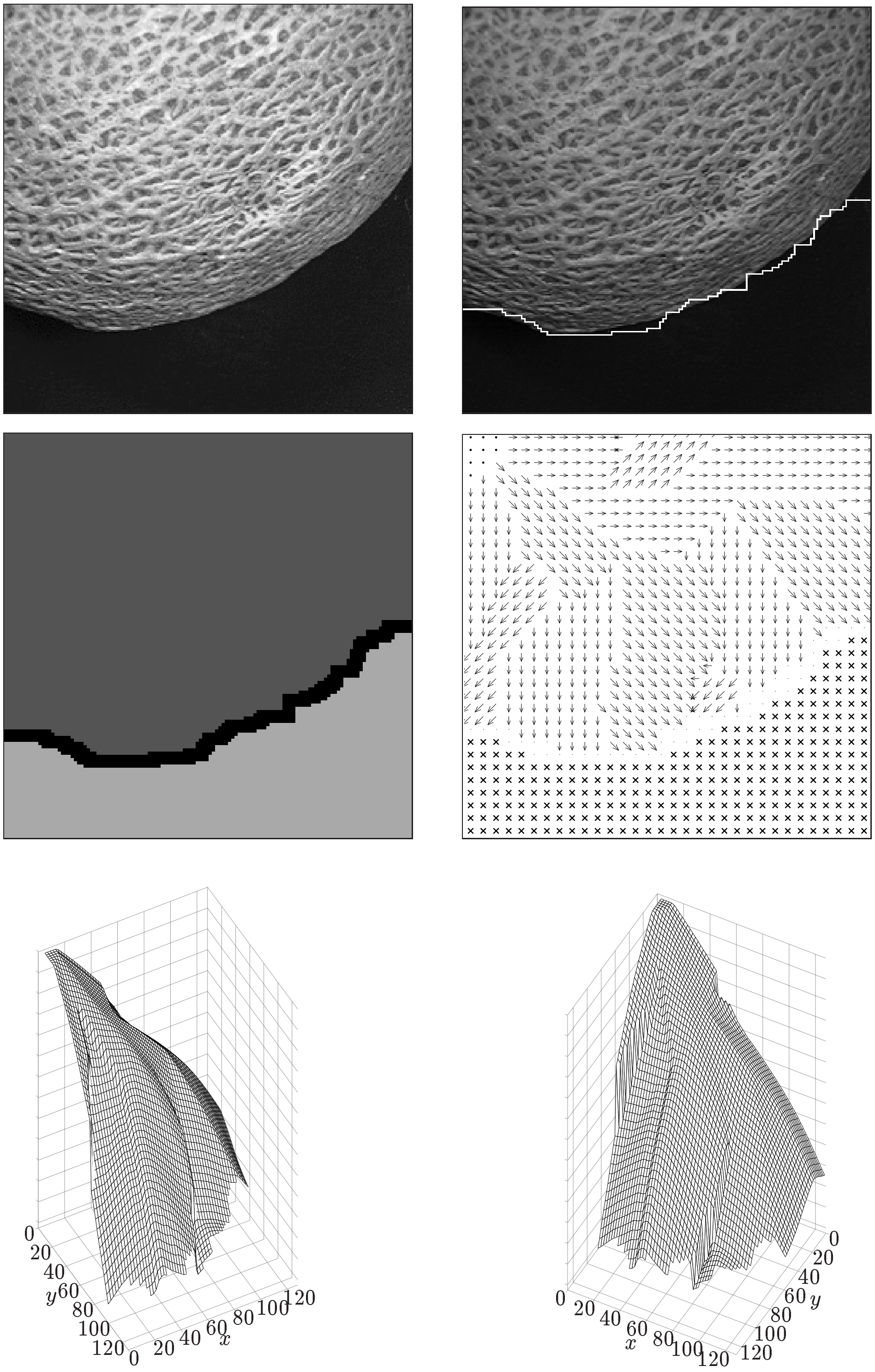


Figure 6.10: Results for test image A. The segmentation output is shown in the upper right ($\alpha = 242$); the corresponding region representation is shown in the left column of the middle row. The tilt direction map is shown in the right column of the middle row; it has been reduced to size 32×32 in order to make it legible in this figure. Areas containing symbol × in the map indicate a region interpreted as lacking texture. A plot of the surface shape estimated by the model is shown in the bottom row. The surface is shown from two vantage points; in each plot, the observer's viewpoint is an imaginary location above the surface and looking down at it.

Results for Test Image B

Figure 6.11 on the following page presents the results for the second test case. The input is an artificial image of two textured cylinders oriented orthogonally to each other, with one cylinder in front of the first. The main purpose of this test case is to illustrate the effects of one of the limitations in the current formulation of the model.

The results for this case are good within the limits of the model's design. The segmentation output shown in the upper right-hand corner captures the main texture regions in the image, although it suffers from three minor problems. One is that the region borders fall some distance away from the true surface borders. This is a problem with the segmentation method already noted in Experiment Three, and is due to the fact that complex cell responses tend to spill across textured/untextured region borders. A second problem is that the borders for the top and bottom cylinder portions cut into the middle cylinder. This is the source of some roughness in the shape estimates discussed below. Finally, there are some stray border lines in the segmentation output, but these have been removed in the final region representation (shown in the left-hand column of the middle row in the figure). This stray-line removal is the goal of the filling-in process described in Section 6.1.3.

The tilt direction map produced by the model for this case is reasonably good. The tilt arrows show some confusion near the areas of lowest normalized APF, but they generally point in the expected directions. The areas of lowest APF are located roughly where they should be. In addition, the four corner regions in the image have all correctly been found by the simulation to be devoid of texture.

Except for the positioning of the three surfaces, the shape rendition in the bottom row of Figure 6.11 is also good. Each surface region clearly shows a cylindrical quality, although the region at the bottom of the image has a much rougher form due to an abrupt transition from flat top to steeply-angled sides. The problem with flat areas in the surface shape estimates is discussed in more detail in Experiment Two. The notches in the horizontal cylindrical surface are due to the way that the segmentation borders cut into the middle cylinder from the top and bottom halves of the vertical cylinder.

The shape output demonstrates in a vivid way one of the most important limitations of the model: it must analyze each texture region independently, and is unable to determine the relative distances between the surfaces in the input. This caused the top and bottom surface renditions to be placed at the same relative distance from the viewer as the middle cylindrical surface, something that disagrees with human perception of the surfaces in the input image. The inability of the model to judge relative distances was discussed in Chapter 4, and stems from the fact that the model only uses orthographic image cues. As

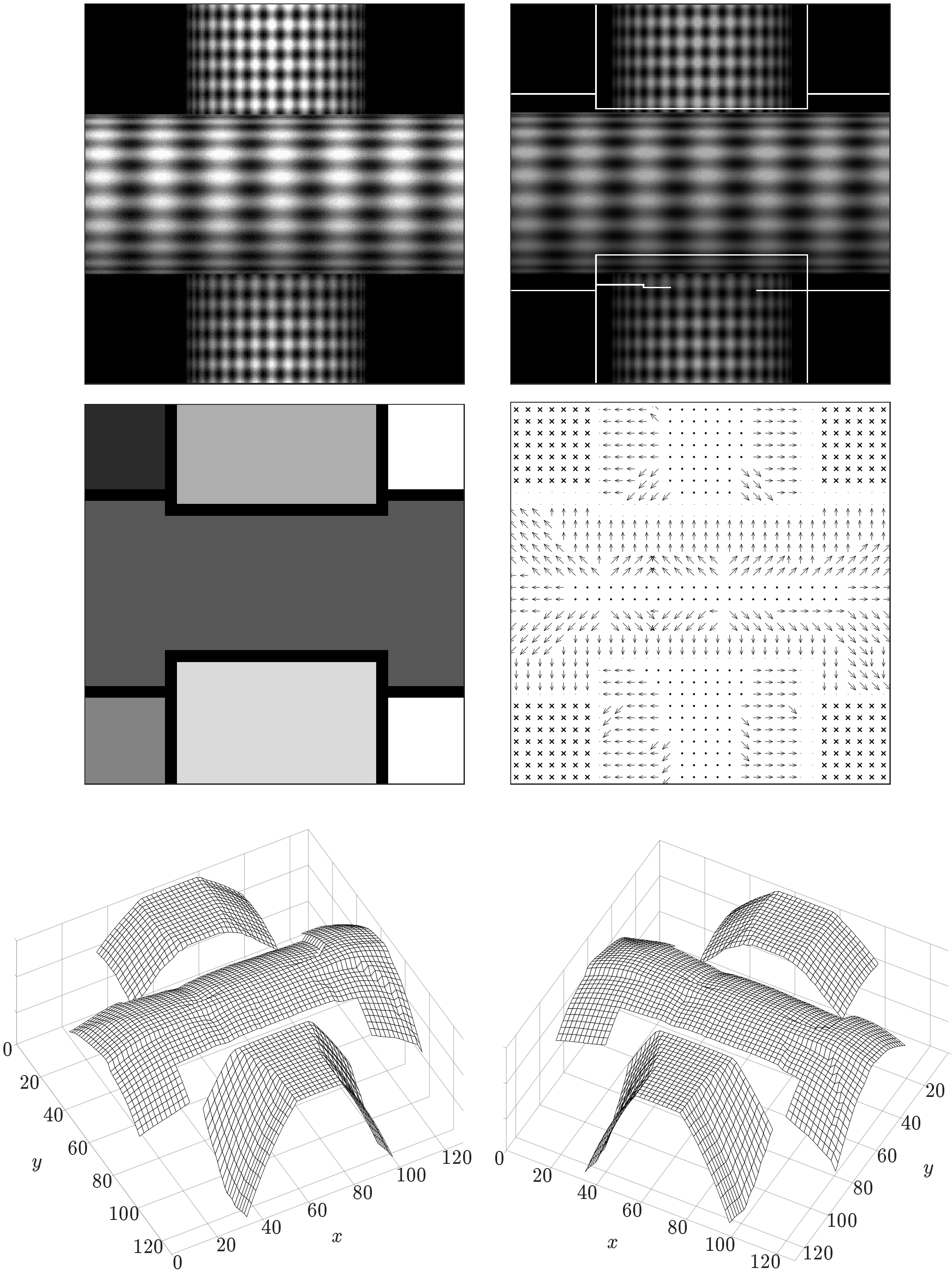


Figure 6.11: Results for test image B. The segmentation output shown in the upper right was produced with $\alpha = 336$. The format of this figure is the same as that of the previous figure.

mentioned before, it would be necessary to use some other source of information such as perspective cues (and perspective image projection) in order to overcome this limitation.

It is important to note that humans bring additional information to bear on this problem that allow us to perceive the surface shapes more accurately. For example, we interpret the cylindrical surface in the back to be continuous, but in fact, there is no *direct* evidence in the image itself that the upper and lower cylindrical surfaces are joined behind the horizontal cylinder. Humans *infer* this, but doing so is going beyond the information given by the image. It is impossible for a system to make this interpretation without using an inference of this sort. Thus, treating the surface patches as being separate (as this model has done in Figure 6.11) is valid given the available information.

In summary, the simulation outputs satisfy the predictions for this experiment. The segmentation mechanisms located the major regions in the input and removed stray border lines; the shape estimation mechanisms ignored the regions lacking texture in the input; and the renditions produced by the shape estimation process describe the overall surface shapes in a qualitatively accurate manner, within the limits imposed by the model's inability to place separate surfaces at their correct relative distances.

Results for Test Image C

Figure 6.12 on the next page presents the outputs of the simulation for the third test input. The image contains two texture regions resembling, in a vague way, a rock next to a tree trunk, plus one blank region.

The outputs from the simulation for this third case are excellent. The segmentation output tracks the region borders well, although once again there is some notable localization error. The stray border lines are removed by the segmentation post-processing steps, resulting in a region representation that is quite clean. The tilt direction map shows that the model achieved good overall estimates of the tilt directions over both the spherical and the cylindrical textured surfaces. There is some error in the tilt estimates over the cylindrical region on the right: the region of lowest normalized APF in the region (corresponding to the area closest to the viewer) should ideally cover the entire right-hand edge of the region, but instead in the model's output it covers only a short, oblong area. This led to inaccuracies in the tilt direction estimates; some of the locations point vertically up and down instead of pointing towards the left as expected for this surface shape. It is likely that this error was caused or at least exacerbated by image edge effects.

The surface rendition shown in the bottom row of Figure 6.12 is excellent. The three-dimensional shapes are smooth and match the shapes in the image quite well. There is a small error in the rendition of the cylindrical portion; the lower section was interpreted

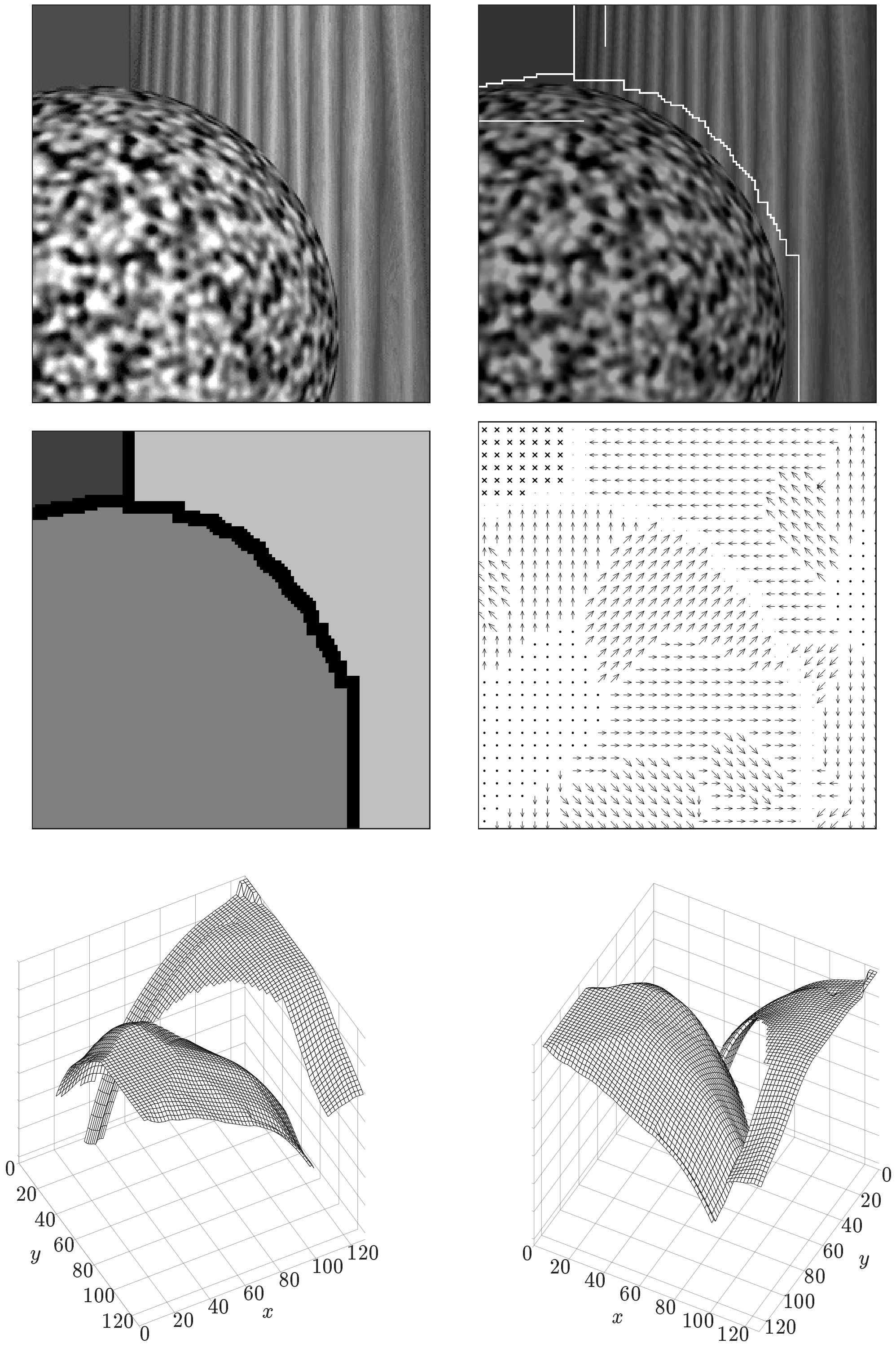


Figure 6.12: Results for test image C. The segmentation output shown in the upper right was produced with $\alpha = 240$. The format of this figure is the same as that of Figure 6.10 on page 277.

by the model as falling away from the viewer, which does not reflect the true shape of the surface. This was caused by the failure of the model to correctly locate the area of lowest normalized APF as mentioned above. The output also displays the problem noted with the previous test case: the model is unable to judge the relative distances between the surfaces in the image. The result is that the spherical and cylindrical surfaces in the surface estimates are placed at the same distance. By contrast, it is more likely that a human observer would judge one of the surfaces as being closer.

Notwithstanding the model's inability to distinguish relative positions of surfaces, the model performed quite successfully on this third test case. Once again, the three predictions for the experiment have been confirmed.

Results for Test Image D

Figure 6.13 on the following page presents the results for the fourth test case. The input is another artificial image of two semi-spherical shapes next to a cylindrical tree-like texture. This image is similar to, but more complex than, the previous case.

For the most part, the simulation results for this case are good. The segmentation output is of decent quality, though it is somewhat rough near the middle juncture of the three textures. The region labeling output (in the left column of the middle row) shows that the post-processing steps have cleaned up the segmentation results to some extent.

The tilt direction map and shape rendition do contain one notable error: the small rectangular region in the upper right-hand corner should have been found to be devoid of texture, but it was not. This resulted is a small surface patch in the upper right-hand corner of the tilt direction map and the surface rendition in the bottom row. In order to understand why this occurred, I examined the set of eight APF maps in that region. This revealed that there was a large proportion of complex cell activity in the area, even though this region is free of texture in the image. The source of the activity turned out to be the spill-over of complex cell responses first noted in Experiment Two: there were responses from cells tuned to low spatial frequencies some distance away from the actual border of the cylindrical region in the image. These responses covered a proportion larger than c_t times the total number of spatial locations in the corner, leading the simulation to label the region as containing texture.

In other respects, the tilt direction map and shape rendition are fairly good, albeit rough. The estimates for the spherical region in the lower left are especially rough, with rather sharp creases in the three-dimensional surface reconstruction and a noisy tilt direction map. The poor quality of the result in this region may be due to inhomogeneities in the tessellated texture. This texture is somewhat less homogeneous than other textures

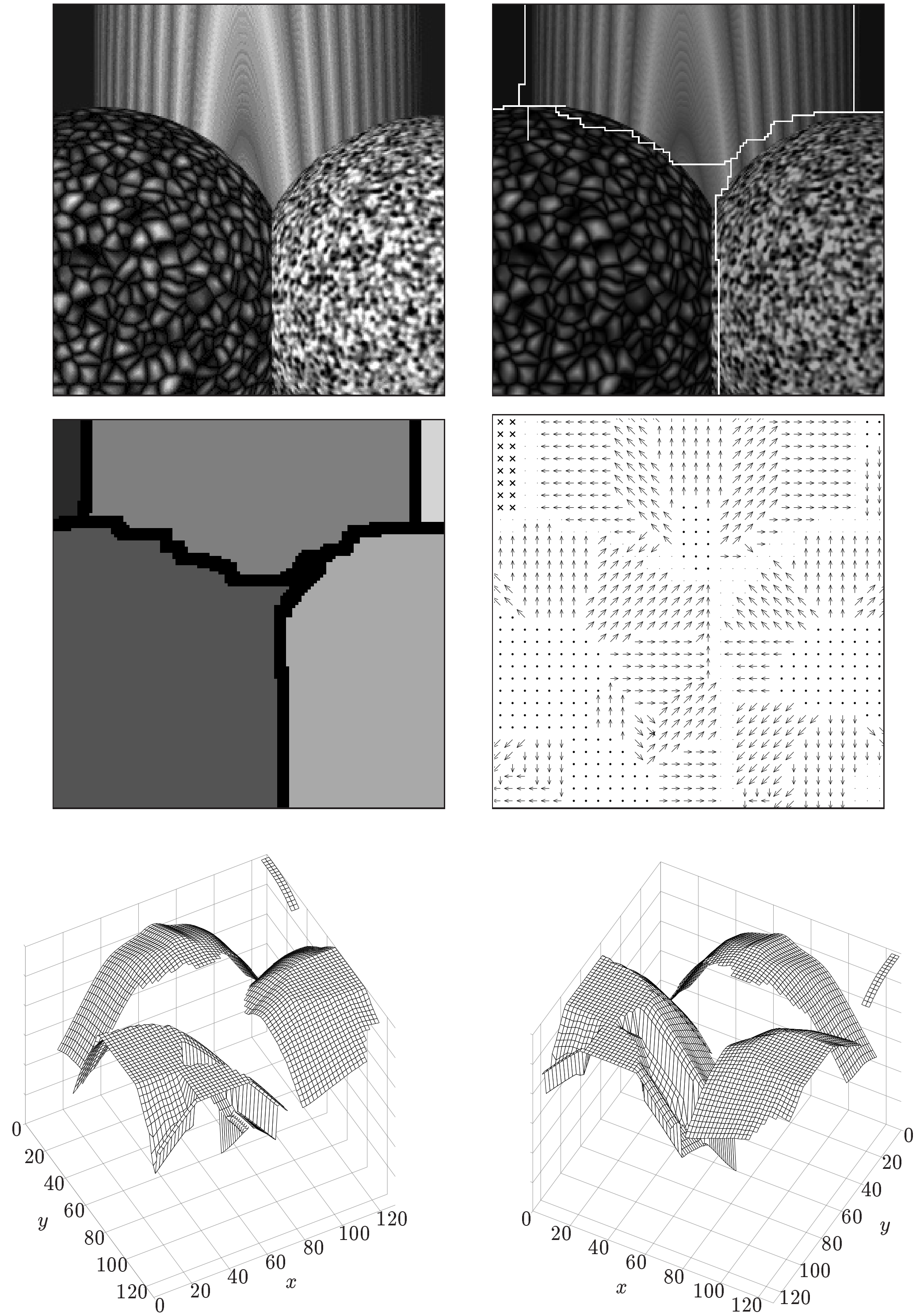


Figure 6.13: Results for test image D. The segmentation output shown in the upper right was produced with $\alpha = 155$. The format of this figure is the same as that of Figure 6.10 on page 277.

Table 6.3: Alternate parameter values explored for test image E

Parameter	Original Value	Alternate Value
c_l	1.37	1.17
l_p	37	17

used in the images in this experiment, and this may have confused the shape estimation process. Aside from this, the model produced reasonable interpretations of the remaining two texture regions. The cylindrical region at the top, and the spherical region in the lower right, were both given fairly good shape estimates.

In summary, the results for this case are mixed, and only partly confirm the predictions for the experiment. The segmentation mechanisms performed well, but the shape estimation component was only partly successful: it ignored one area devoid of texture but not a second one. I discuss this result further in Section 6.4.3 below.

Results for Test Image E

Figure 6.14 on the next page presents the outputs of the simulation for the fifth test input, a photograph of three balls covered with dots.

The results for this input at the current parameter settings are disappointing. Although the segmentation and region representations are acceptable, the tilt direction map shows that the leftmost surface was estimated to be completely flat, and one of the surface rendition for one of the other two surfaces contains a severe break. Thus, although the segmentation component succeeded on this test case, the shape estimation component failed to produce an adequate result. Based on this, only the first prediction for the experiment is satisfied by these results.

Experiment Two (involving the shape estimation model) has already revealed that parameter settings that work for images containing large surface regions can sometimes be inappropriate for ones containing smaller regions. The present test image includes three relatively small surfaces. I decided to search for new parameter settings that might allow the shape estimation model to produce an acceptable output for this test case. I began this search by reducing the values of the threshold c_l and the path length l_p, and soon found a new range of parameter values for which the shape estimation process produced satisfactory outputs. Table 6.3 summarizes the values which gave the best subjective results. The output produced by the model using these parameter values is shown in Figure 6.15 on page 286.

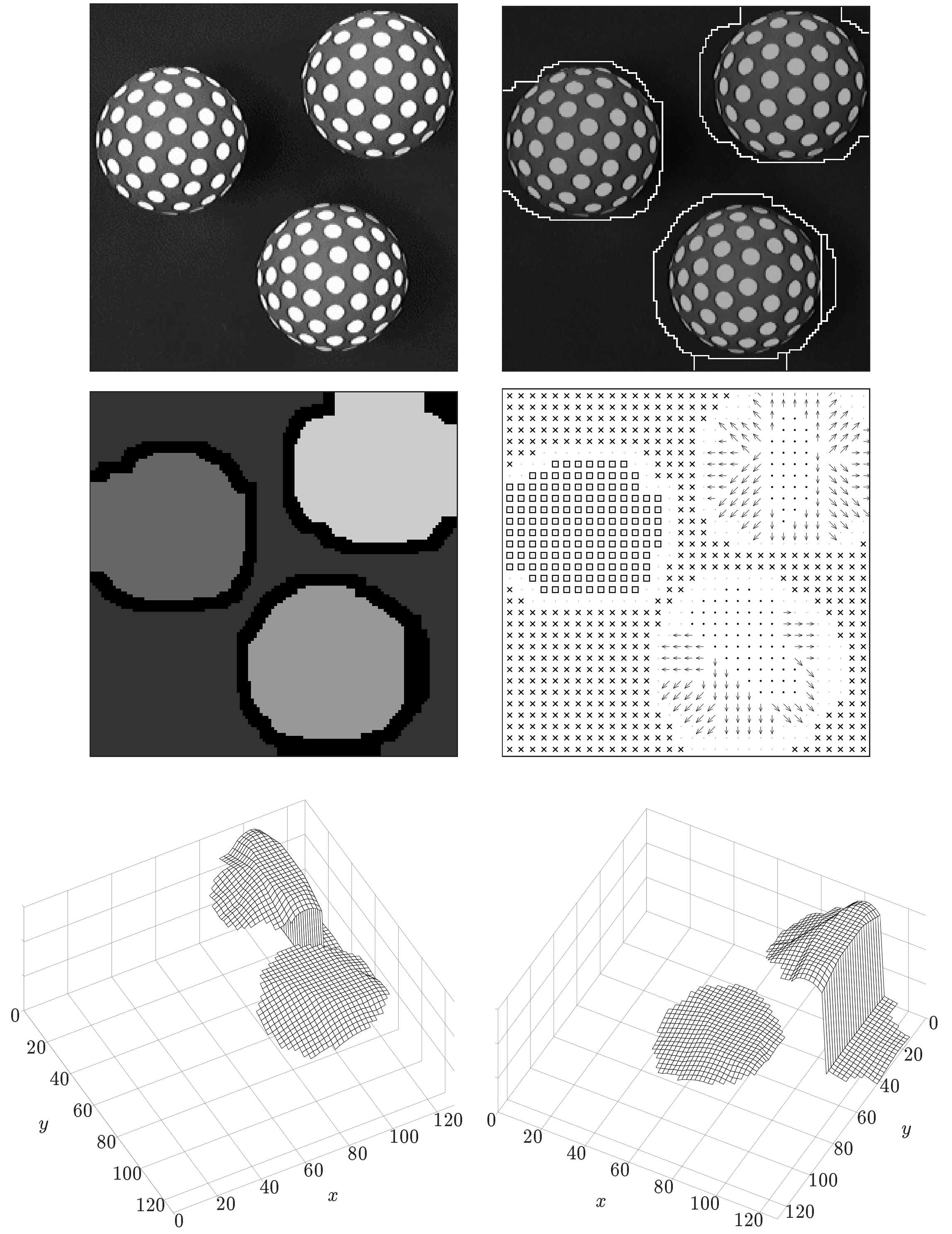


Figure 6.14: Results for test image E. The segmentation output shown in the upper right was produced with $\alpha = 82$. The format of this figure is the same as that of Figure 6.10 on page 277. Areas containing symbol □ in the map indicate a region interpreted as containing texture on a flat surface.

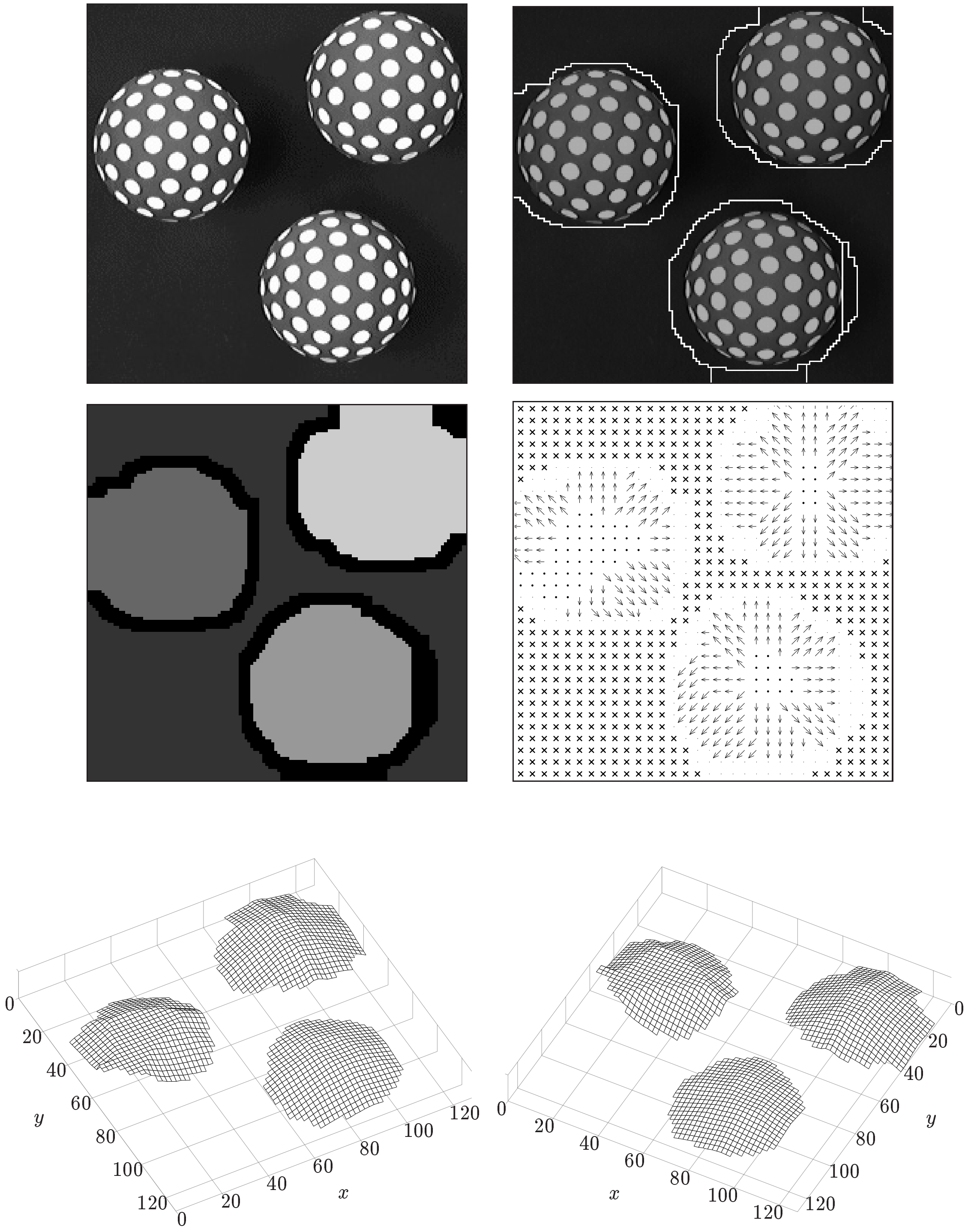


Figure 6.15: Results for test image E using alternate parameter settings. The segmentation output shown in the upper right was produced with $\alpha = 82$. The format of this figure is the same as that of Figure 6.10 on page 277.

The results using these new parameter settings are much improved over the original outputs shown in Figure 6.14. This time, the simulation did not interpret the leftmost surface as being flat, and the break in the upper right surface has disappeared. In addition, the surface renditions in the bottom row of the figure are considerably improved, demonstrating clear similarities to the shapes in the input image.

The estimated surface shapes are still less than ideal; all three surfaces are too flat to agree with human perception of the input image. The probable cause of the flatness is the fact that the surface textures are fairly coarse. Even towards the edges of the surfaces, the texture elements remain rather large and cover the surfaces sparsely. This may mean that the range of scales across each region is small. Since the shape estimation method relies on detecting changes in spatial frequency, which is correlated with scale of texture across the surface, a limited range of texture scales implies a limited range of changes in APF values. This can lead to flat surface renditions.

Does this mean that the original parameter values shown in Table 6.2 on page 273 were in fact unsuitable, even for the other test cases? To answer this question, I retested the other input images using the alternative parameter values shown in the right column of Table 6.3. Some of the resulting outputs were still reasonable, but for most input images, the results were substantially worse and would not be considered acceptable. Thus, it seems that this particular test case needs different parameter settings in order to produce an acceptable result. Some of the implications of this are discussed in Section 6.4.3 below.

In summary, the three experimental predictions are only satisfied for this case if two of the parameter values are readjusted. Thus, the results on this case are mixed.

Results for Test Image F

Figure 6.16 on the following page shows the outputs of the simulation on the final test case. The input is an artificial image intended to resemble the scene pictured in Figure 1.1 on page 2.

The results for this final test case are once again mixed. First consider the segmentation results. The output shown in the upper right corner of the figure shows that the segmentation model capture the major regions fairly well. There are several small, extraneous regions in the segmentation result, but these were eliminated by the post-processing mechanisms, leaving the six major regions depicted in the left column of the middle row. This portion of the model therefore succeeded on this test case.

The shape estimation component was only partly successful. The dark, spotted texture covering the simulated ground in the image appears to be flat, which is expected due to the use of orthographic image projection. The shape estimation mechanisms correctly

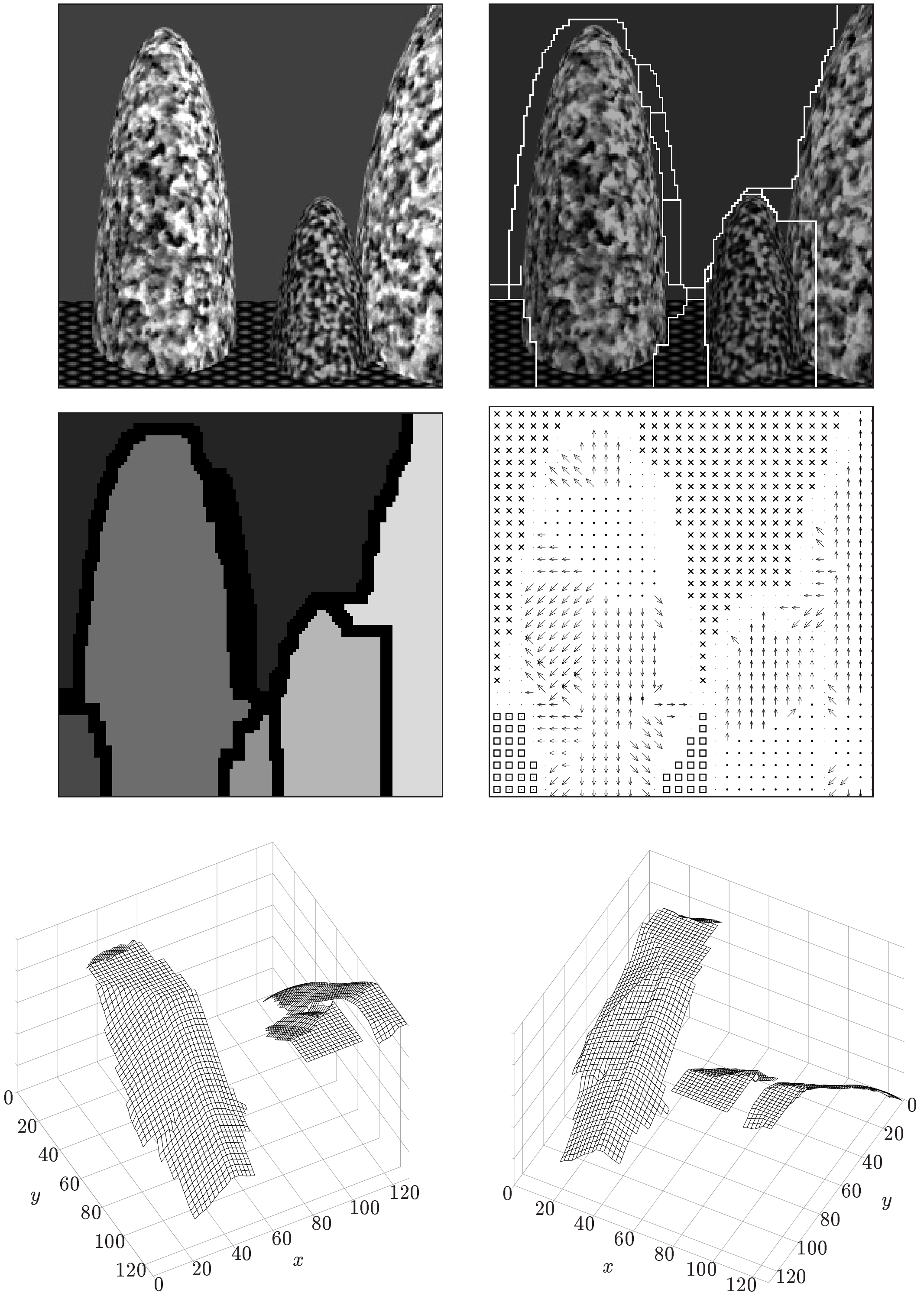


Figure 6.16: Results for test image F. The segmentation output shown in the upper right was produced with $\alpha = 87$. The format of this figure is the same as that of Figure 6.10 on page 277.

labeled this area as flat in the tilt direction map. Likewise, the system correctly labeled the upper portion of the image (the background) as being devoid of texture.

The rest of the tilt direction map for this image is partly acceptable. The notable failure involves the tall, oblong region in the left half of the image. The oblong region suffers from an incorrectly placed area of lowest normalized APF. The low APF area was placed near the top of the region, whereas it would be more appropriate if the region cover a strip in the middle lower portion of the region. Because of this problem, the tilt directions in the oblong region are misdirected. In the other two regions of texture, however, the tilt directions were estimated more appropriately. The low APF regions in both of those cases is near the bottom of the region, roughly as it should be.

The failure in estimating the correct tilt directions in the oblong area led to a surface rendition that does not match expectations. The representation shows the surface to be closest to the viewer at the top and angled away from the viewer near the bottom, exactly opposite of how a human would likely interpret the shape in the image. The shape estimates for the two smaller regions at the right are better; the surfaces were estimated to fall away from the viewer at the top, and slightly less so at the sides, but it would be more appropriate if the surface renditions were curved away from the viewer on both sides.

In summary, the experimental predictions are only partly satisfied for this test case. The segmentation mechanisms once again succeeded, but the shape estimation portion produced only moderately acceptable results on some of the textured surfaces and unacceptable results on another.

6.4.3 Discussion of Experiment Four

Overall, the results of this experiment demonstrate that the combined model of textured-based segmentation and shape estimation generally works as intended. The simulation can produce approximately correct segmentations as well as perceptually plausible representations of the general shapes of textured surfaces in the input.

With respect to the three specific predictions made at the beginning of the experiment, the results of this experiment have been mixed:

1. The segmentation mechanisms produced qualitatively correct representations of the major regions in every input image. The additional mechanisms described in Section 6.1 worked as intended; they delineated the regions bounded by the segmentation boundaries, eliminated excessively small regions, and removed stray border lines. Thus, the results completely support the first prediction for this experiment.

2. The shape estimation component almost always succeeded in detecting and ignoring regions devoid of texture and regions containing flat surfaces in the input. There was one exception for each type of condition. First, the simulation erroneously considered one untextured region in test image D to be textured. The cause of this was revealed to be in the complex cell responses: the spill-over of complex cell responses first noted in Experiment One lead to nonzero APF values in an untextured area immediately adjacent to a textured area. This is therefore not so much a failure of the mechanism described in Section 6.2.1 as it is a general problem with the complex cell responses. Second, when using the parameter settings used for most of the experiment, the simulation erroneously labeled one curved surface region in test image E as being flat. This was caused by parameter c_l being somewhat too high for that particular image. Parameter c_l has a direct impact on the detection of flat areas, but it is not actually part of the mechanism for detecting flatness described in Section 6.2.3. One of the other default parameter values was also not suitable for test image E; it therefore seems more appropriate to call this a failure of setting the model parameters rather than a failure of the flat-area detection mechanism. Thus, the results support the second prediction for this experiment.

3. The shape estimation component generally produced qualitatively correct shape estimates for each texture region. However, there was one clear failure in test image F: the model failed to make an acceptable interpretation for one of the texture regions in the image. In addition, some of the shape estimates for the other images were relatively rough. Thus, the experimental results do not entirely support the third prediction.

In summary, the first two predictions are supported by the results of the experiment, but the third prediction was not supported.

On the Failure to Produce Qualitatively Correct Shape Estimates

The surface renditions produced by the shape estimation model were sometimes of poor quality, and in one case (image F), there was an outright failure to produce an acceptable interpretation. Why did this occur?

Given that the model succeeded in most instances, the problem seems more likely to be rooted in the characteristics of certain input images. The rock-like textures in image F have visible inhomogeneities, and could easily have led the model to misplace the location of the area of lowest normalized APF for the oblong region. Similarly, in the case of image D, the texture on the lower left surface that produced a rough surface shape estimate is somewhat less homogeneous than the other textures in that image. There are areas where the tessellations in the texture seem to be pinched together, and other areas where they are so dark that they almost disappear; these changes do not correspond to texture compression in the image projection due to surface orientation changes. It is possible that the inhomogeneities happened to change the spatial-frequency spectrum across the region in such a way as to mislead the shape estimation process. The failure in test case F and the roughness in some of the other results are therefore likely to be due to failures of satisfying the assumption of texture homogeneity.

This begs the question of whether the shape estimation component is too sensitive to the lack of texture homogeneity. The results of this experiment suggest that it is. Some of the parameters in the model have a significant impact on this sensitivity, in particular the various averaging radii (r_a, r_i, and r_s). The larger these averaging radii are, the more widely the input data are smoothed, and therefore the less sensitive the process becomes to inhomogeneities in the textures. Unfortunately, by the same token, the larger the radii are, the less sensitive the process is to changes in surface shape across a region. Thus, there is a tradeoff: decreasing sensitivity to texture inhomogeneities necessarily involves simultaneously decreasing sensitivity to surface shape.

Rather than attempting to maintain a single value of these parameters for all inputs, it may be more appropriate to include mechanisms that rescale the parameter values based on each input image. Other evidence also points to the need for this, as discussed next.

On the Difficulty of Finding Parameter Values Suitable to All Inputs

The problem of finding appropriate values for the parameters in the model has become a recurrent theme in this research. Once again, this experiment has shown that although it is possible to find fixed parameter values that will work for most input images, it is difficult to do so, and it is more likely that the settings will fail to be appropriate for some kinds of inputs. In the present experiment, this was made clear by the failure on test image E: the shape estimation component initially failed to produce an acceptable result, but after adjusting just two of the parameters, the mechanism was able to produce reasonable shape estimates.

This makes a strong argument for introducing additional mechanisms to rescale parameter values automatically based on characteristics of each input. The use of fixed parameter values is only acceptable as a temporary measure, until further research can uncover what kinds of mechanisms would be suitable. The results on test image E suggest that at a minimum, the path length parameter l_p and the threshold c_l in the shape estimation component should be scaled dynamically. Chapter 5 discusses parameters in the segmentation component that should similarly be rescaled based on qualities of each input image.

On the Limitations Imposed by Orthographic Projection

Test image B makes clear one of the practical implications of using orthographic image projection: the model is unable to estimate the relative distances between different surfaces in a scene. All texture regions are analyzed independently and placed by the model at the same arbitrary distance from the viewer. This can lead to surface reconstructions that do not entirely correspond to a human observer's interpretation of the same scene. This limitation could be removed by augmenting the shape estimation model with the ability to use additional sources of information such as perspective (Sakai & Finkel, 1997).

It is worth noting that neither the present model nor an extended model that exploits perspective projection can estimate *absolute* distances to surfaces in a scene. In order to do so, it is necessary for a model either to know the actual sizes of texture elements on surfaces, or else to use other (non-texture) sources of information. If the model knows the sizes of texture elements on surfaces in a scene, then it can estimate the distances from the viewer to the surfaces based upon the reduction in size of the texture elements as seen in the image. Unfortunately, it is almost never the case that one can know the true sizes of texture elements on a surface, especially in a natural environment. Thus, for practical purposes, a system must rely on non-texture sources of information (e.g., motion parallax, stereo disparity, lens accommodation, eye convergence, etc.) to judge the actual distances from the viewer to surfaces in a scene.

6.5 Summary

The previous chapters have described the individual components of the overall model; in this chapter, the components have been combined into a complete system. The combined model takes the outputs of complex-cell-like mechanisms and performs both segmentation and shape estimation on images containing multiple regions of curved textures.

Some of the modifications to the individual components that are required prior to combining them were described in chapters 4 and 5. In the present chapter, I have introduced additional changes and new mechanisms to complete the overall model of combined textured-based segmentation and shape estimation. The changes to the segmentation component (described in Section 6.1) consist of adding mechanisms that implement the following processes:

- Inhibiting shape estimation near region borders, simulated here by "thickening" the borders;

- Labeling the region bodies implied by the border-based region representation produced by the MCM segmentation model;

- Filling in holes within regions due to stray border lines; and

- Detecting and ignoring excessively small regions.

The changes to the shape estimation component (described in Section 6.2) consist of adding mechanisms that implement the following:

- Detecting and tagging regions devoid of texture in the input;

- Performing APF normalization within each separate segmentation region rather than over the entire input;

- Detecting and tagging regions containing textures on flat surfaces; and

- Improving the method of estimating integration directions near region boundaries.

The model described in this chapter can accept images containing a mixture of textured and untextured surfaces, but in order for the model to operate properly, the input images must satisfy certain constraints. These constraints are a combination of the requirements on input images imposed by the individual components of the overall model. They are described in more detail in Section 6.3.4; briefly, the constraints are:

1. Images must be created by orthographic image projection;

2. The textures on physical surfaces must be homogeneous or nearly homogeneous;

3. The textures on surfaces must be visible—shading effects must not hide the textures;

4. Regions of texture in the image must be sufficiently large compared to the various averaging radii in the model; and

5. The range of spatial frequencies in a scene must not be too great.

Experiment Four shows that if an input image satisfies these requirements, the model of combined segmentation and shape estimation can produce reasonable results. It can generate rough segmentations that capture the major regions of texture and nontexture in an input, and estimate the general surface shapes in regions containing texture. It can also detect and flag regions that are devoid of texture and regions that appear to contain flat surfaces. Nearly all the test inputs resulted in outputs that agree qualitatively with the interpretations that a human might make given the same input image. Most of the predictions for the experiment were satisfied, with the exception of a failure to produce an acceptable shape estimate for a portion of one of the input images.

Experiment Four also revealed some problem areas that would benefit from further research in the future:

- The complex cell responses tend to spill over the boundaries between regions of texture and nontexture. This can cause an untextured area to be erroneously labeled as textured, because the presence of complex cell responses can fool the mechanism that detects untextured regions.

- The use of fixed parameter values for all input images means that some inputs are not interpreted correctly, because it is difficult to find a single set of parameter values that will allow the model to work adequately for all inputs.

- The shape estimation model is fairly sensitive to inhomogeneities in surface textures. This occasionally causes the simulation to fail to produce a correct interpretation of an input.

- The assumption of orthographic image projection imposes limitations that can cause the model to produce overall scene renditions that do not agree with human interpretation. The reason is that the relative distances between different surfaces in a scene cannot be estimated, and instead all surfaces are placed at the same distance from the viewer.

These problems, and ideas for overcoming them, are discussed further in the next chapter.

Chapter 7

Summary and Discussion

The motivation for this research has been to study how the visual system could use texture to perform two basic tasks in spatial layout analysis: locating possible surfaces in the visual input (segmentation), and estimating their approximate shapes. The primary goal of this dissertation has been achieved: I have presented a biologically-motivated model of combined, texture-based segmentation and shape estimation. The model demonstrates that by integrating segmentation and shape estimation, a system can share information between these processes, allowing them to constrain and inform each other as well as save on computations. The software simulation of the model exhibits good performance and clearly demonstrates the feasibility of the approach. As desired, the simulation can perform texture-based segmentation and shape estimation on images containing multiple, curved, textured surfaces.

In this chapter, I take a step back and review the work. This research has covered several topics in vision research, making contributions to each of them, and it is important to summarize the accomplishments. Further, although the simulation of the model operates reasonably well for simple types of inputs, it has some shortcomings and limitations. Of course, this is intrinsic to the modeling process—models and simulations are never complete and without limitations. What is crucial is understanding the nature of the limitations and whether the important ones can *ever* be overcome. Therefore, in this chapter, I also examine the problems remaining with the model and present ideas for solving them.

This chapter is organized as follows. I begin in Section 7.1 with a summary of the accomplishments of this research. In Section 7.2, I discuss the plausibility of the overall model from biological and psychological perspectives. Section 7.3 contains an analysis of the remaining problems in the model, and Section 7.4 contains ideas for overcoming some of these problems. In Section 7.5, I discuss some predictions that are implied by the model, and in Section 7.6, I discuss ideas for future directions in this line of research. Section 7.7 then concludes this chapter.

7.1 Accomplishments

In Chapter 2, I pointed out that nearly all existing models of texture-based segmentation have been designed and tested on images of flat patches of texture being viewed frontally. But realistic scenes contain variations within regions of texture in the image because of surface curvature and other effects. On the other hand, nearly all existing models of texture-based estimation of surface shape have been designed and tested on images containing single regions of texture. But again, realistic scenes contain multiple regions of texture, each with different shapes and orientations.

The primary accomplishment of this dissertation is developing a system that can perform both texture-based segmentation and estimation of shape on images containing multiple, curved surfaces. The only other examples of combined processing published to date are either limited to obtaining the orientations of flat surfaces (Krumm, 1993; Krumm & Shafer, 1994), or rely on feature-based texture analysis methods that only work for limited classes of textures (Moerdler, 1989; Moerdler & Kender, 1987). By contrast, the model developed here can handle more general types of surfaces and textures. This is important because analyzing the spatial layouts of realistic scenes requires the ability to segment inputs containing slanted and curved surfaces, as well as the ability to estimate spatial properties of multiple regions in the input.

Achieving this combination of capabilities required bringing together a collection of evidence, ideas and techniques from three topic areas in vision research: cortical complex cells, texture-based segmentation, and texture-based shape estimation. These subjects have almost invariably been treated separately in the literature. As a result, most existing research on texture-based vision has not considered the possible interactions between the different processes. The model developed in this dissertation shows that there can be benefits to considering these interactions: at the very least, it may be possible to share information between different computational components.

In addition to the primary accomplishment of this work, this dissertation offers contributions in each of the three separate subject areas mentioned above. Each contribution is discussed in the following paragraphs. Table 7.1 on the following page summarizes all of the accomplishments.

7.1.1 Models of Texture-Based Segmentation

The goal of segmenting more general types of surfaces is met by using an approach to texture-based segmentation that can cope with texture distortions created by image projection effects. This is in contrast to most models of segmentation, which are designed

Table 7.1: Accomplishments of this research

Description
1. Development of a system that can perform both texture-based segmentation and estimation of shape on images containing multiple, curved surfaces
2. Adaptation of the coupled-membrane model of segmentation to use an intermediate result computed as part of shape estimation
3. Development of segment labeling and cleanup mechanisms that follow the basic segmentation process
4. Adaptation of the average peak frequency model of shape estimation to work within individual regions found by the segmentation process
5. Development of mechanisms that allow the shape estimation component to discriminate between regions containing texture and regions devoid of texture, and between regions containing curved surfaces and regions containing flat surfaces oriented frontoparallel to the viewer
6. Combination of the contrast-normalization model of complex cell behavior with the Gaussian derivative model of receptive field profiles, for a more realistic model of complex cells
7. Distilling of experimental data about properties of the receptive fields of cortical cells that need to be considered when developing a simulation of complex cells

for the limited case of flat textures viewed from the front. As I have already mentioned, this is a situation rarely encountered in natural settings. The present model's ability to handle curved and slanted textures comes from using a parallel, distributed segmentation approach that explicitly allows gradual shifts in texture qualities. The approach is based on an existing model, the coupled-membrane model of Lee (1993, 1995; Lee et al., 1991, 1992).

One of the specific contributions of the present research is the adaptation of the coupled-membrane model to start with an intermediate result computed by the shape estimation process. This intermediate result is an average measure of the dominant spatial frequencies in the neighborhood of each location in the visual input. This collapses the four-dimensional set of complex cell responses to a three-dimensional set of measures, significantly reducing the number of computations (and thus the time) required by the process to segment a visual input. The resulting system loses some precision in its ability to locate boundaries between texture regions, but it is entirely adequate for performing coarse segmentation of the major regions of texture in an input image.

An additional contribution is the development of processes that clean up and label the outputs produced by the modified coupled-membrane model of segmentation. This transforms the results into a format that is more effective for guiding the process of shape estimation.

7.1.2 Models of Texture-Based Shape Estimation

The present research embodies two accomplishments in the area of texture-based shape estimation. First, I offer a way of adapting the average peak frequency model of Sakai and Finkel (1994, 1995, 1996) to work within individual regions found by the segmentation process. Sakai and Finkel's original model (and most other models of shape from texture) was designed around the assumption of single surfaces being presented one at a time. The simple but important adaptation to remove this restriction (described in Section 6.2.2) allows the shape estimation approach to be an integral component in a more general model. Second, I extend the shape estimation process (in Chapter 6) to make it able to detect regions devoid of texture and regions containing flat textured surfaces oriented facing the viewer. These are important mechanisms in a model of texture-based shape estimation. Without them, the shape estimation process will not function properly for untextured and flat regions without external supervision. The need for these additional processes is usually not encountered in most research on texture-based shape estimation, because the (human) experimenters typically test the models on regions containing texture and manually prevent their operation over regions devoid of texture and regions containing flat surfaces.

7.1.3 Models of Cortical Complex Cells

This model of texture-based segmentation and shape estimation is based on a biologically-plausible model of cortical complex cells. Two accomplishments contribute to this.

First, the simulated complex cells used here have many qualities that mimic those of actual cortical cells, in particular with respect to correlations between tuning properties. For example, quantitative studies of cortical simple and complex cells (e.g., De Valois et al., 1982, 1985; Foster et al., 1985) show that there are correlations between quantities such as preferred spatial frequency, spatial-frequency bandwidth, and orientation bandwidth. The simulated complex cells used in this project are designed so as to embody these correlations. The collection of data used to design the cells represents a distilling of many constraints and can serve for developing other models.

Second, the complex cells are based on a model of complex cell function combined with a model of simple cell receptive fields at many spatial scales. Both models are supported by experimental evidence. The functional characterization of complex cells is based on the contrast-normalization model (Heeger, 1991, 1992a, 1992b); the receptive fields are based on Gaussian derivatives (Young, 1985, 1986, 1991). Contrast-normalized cells are often implemented with receptive fields based on a cosine function (Heeger, 1991; Nestares & Heeger, 1997). However, this cosine function was chosen by its authors on the basis of its convenient properties for the contrast-normalization model, rather than on the basis of detailed comparisons to actual cortical receptive fields. By contrast, there is substantial empirical and theoretical research on Gaussian derivatives as models of receptive fields, and the research suggests that the Gaussian derivative is a reasonable idealization of actual simple cell receptive fields. By using this model of receptive fields, the complex cell simulation is better grounded in experimental evidence.

At the time of this writing, there is at least one other published example of combining the contrast-normalization model with Gaussian derivatives (Simoncelli & Heeger, 1997). However, that example uses only one derivative order. The present work uses Gaussian derivatives having many orders, in order to more closely approximate the apparent properties of real cortical cells. There are two costs to this approach. First, a complicated procedure is needed for normalizing, in an approximate fashion, the outputs of the resulting receptive field operators (see Section 3.3.2). Second, the resulting collection of simulated cells can never be normalized perfectly, even using the normalization procedure discussed in Section 3.3.2. Nevertheless, the collection of receptive field operators can be normalized well enough for simulation purposes.

7.2 Plausibility of the Model

The model of combined processing developed in Chapter 6 contains a large number of elements, something that is especially evident from the diagram of Figure 6.8 on page 264. How plausible is the model and the elements composing it? This question has already been touched upon in other parts of this dissertation; here I address the question in the larger context of the whole model.

7.2.1 Mechanisms Used in Each Component of the Model

The present state of knowledge about neural mechanisms in the visual cortex is still in its infancy, and it is difficult to state unequivocally whether the brain can or cannot implement a specific operation. This makes evaluation of a model's biological plausibility difficult, if not ultimately impossible. I believe the best one can do at this time is to evaluate whether specific mechanisms are *reasonable* given available evidence.

For example, one of the mechanisms used in the shape estimation component is lateral inhibition between neurons tuned to orthogonal orientations. (See Figure 6.8 and Section 4.2.1.) The notion that neurons can have inhibitory influences on each other is no longer disputed (though it was at one time; Hebb, 1980), and the idea of inhibition between visually-sensitive neurons in particular is supported by a variety of experiments. Some of these experiments (e.g., Bonds, 1989) specifically support the idea that neurons with different orientation preferences are influenced by inhibitory interactions. It is therefore reasonable to propose the kind of lateral inhibition used in the shape estimation component of this model, even though there is presently no direct evidence for such a process in the brain. This is an example of a biologically-*reasonable* mechanism, as opposed to a biologically-*plausible* one; the burden of supporting a biologically-plausible mechanism is much higher because it demands more direct evidence.

In general, there exists evidence for many *local* operations in the brain; that is, operations that take their inputs from limited areas of the visual field. Thus, an averaging operation such as that used in the computation of average peak frequency (APF) is a type of operation that can reasonably be assumed to be implementable in neural circuits. Likewise, winner-take-all operations (the *max* operations shown in Figure 6.8) also take inputs from a local neighborhood and perform a function that can, at least in theory, be implemented in known neural structures (Winder, 1998). The border inhibition process included as part of the segmentation component is again a local operation.

There also exists evidence that the visual system, beginning even as early as area V1, can implement operations that are regional in scope, meaning operations that integrate information from a wide area of the visual field. The evidence is both direct, in terms of structures for medium-range interactions between neurons in the visual cortex (Gilbert & Wiesel, 1989; Gilbert, Hirsch, & Wiesel, 1990), and indirect, in terms of demonstrations of the influence of regional context upon individual neurons' behaviors (Gilbert, 1993, 1994). Thus, it seems reasonable to propose a mechanism such as the small-region-removal operation in the segmentation component (Section 6.1.4), because the operation only requires integrating information over a limited area of the visual field. It could perhaps be implemented by circuits, located at each point of the visual field, that take their inputs from a certain region of the segmentation output and detect when a segment is smaller than some threshold. A similar argument applies to the process of detecting regions devoid of texture (Section 6.2.1) and regions corresponding to flat surfaces (Section 6.2.3), both of which have the flavor of regional operations. The summation operations in the slant integration stage of the model (see Figure 6.8) are similarly regional in scope.

There is direct evidence in the visual system for filling-in processes in which neural receptive fields expand to fill in an artificially-induced hole in the visual field (Gilbert & Wiesel, 1992; Gilbert, 1993). This is another example of a regional operation. Thus, the neural filling-in processes shown in Figure 6.8 are well-supported and plausible.

The places where one strains to support a theoretical mechanism with biological evidence are where the mechanism requires global information from the entire visual field. However, none of the mechanisms in the present model require global information—at most, the mechanisms require regional information. Thus, at first approximation, the model developed in this dissertation is biologically reasonable, given the current state of knowledge about neural structures and operations in biological visual systems.

7.2.2 Time-Course of Processing Implied by the Model

The overall diagram of the model in Figure 6.8 implies that a series of processes must take place in a particular sequence before a final result is produced. Each of these processes take some amount of time to execute. Is the time-course implied by the model plausible with respect to how long the human visual system actually takes to perform segmentation and shape estimation?

Two major components of the model are clearly not plausible as they are currently implemented. The first is the heavily iterative method used to minimize the segmentation function. As already mentioned in Chapter 5, the current segmentation component requires several thousand iterations per input image. This process takes place in two nested loops, an outer loop that is executed approximately six times for the parameter settings used in simulations here, and an inner loop that executes several hundred times at each turn. However, it is important to note that the approach used here is a software simulation that performs discrete minimization on the segmentation network. As discussed in Section 5.5, it is possible that exploiting parallelism and an analog implementation (such as what has been attempted using VLSI hardware; Harris et al., 1990; Lumsdaine et al., 1991) would allow replacing the inner loop minimization steps with a continuous diffusion-like process (Lee, 1995). This would leave only the outer loop (approximately six iterations) to be executed discretely. Depending on how long the resulting iterations take to execute, the result may be more reasonable as a model of human visual segmentation.

The second unreasonable aspect is the iterative process in the shape estimation component that is used to combine local slant and tilt estimates into a three-dimensional surface estimate. In the simulations shown in this dissertation, this process takes over 100 iterations to produce a result. Again, such a long, iterative process is unrealistic. However, as explained in Chapter 4, the visual system may not need to perform this integration process. The output of the lateral inhibition stage in Figure 6.8 consists of two values at each location of the input: an estimate of the slant and direction of tilt at that location. The collection of these values implicitly defines surface shape. Such a distributed representation may be enough for the visual system to perform certain tasks such discriminating between different surfaces or quickly recognizing common shapes. The implication of this is that the integration stage in the model could be eliminated except for the operation testing for flat regions.

Thus, although these two aspects of the model are not realistic in the face of evidence about the behavior of the human visual system, it seems that they *could* be reimplemented in forms that are more consistent with the time-course of processing in vision. For the sake of argument, let us suppose that these two components could be modified as described above; i.e., the segmentation process reduced to a six-step iteration, and the integration stage skipped in favor of directly using the output from the lateral inhibition stage. Further assume that each computational step (such as a *max* operation) takes approximately 10 ms to execute, and that the complex cells in area V1 of the brain are activated 50 ms after the retinas are stimulated with an input. Then the total time requirements would be:

- 50 ms for complex cells in V1 to produce an output;
- 10 ms for the max_f operation in the APF computation stage;
- 10 ms for the averaging operation in the APF computation stage;
- 60 ms for the modified coupled-membrane segmentation process, and an additional 40 ms for subsequent clean-up steps;
- 10 ms for the step of testing for the presence of texture;
- 10 ms for the normalization step;
- 10 ms for the lateral inhibition stage.

The total is 200 ms, which is not implausible, but still somewhat long compared to the estimates of 100–150 msec for stimulus-evoked recognition activity measured in the brain (Victor & Conte, 1991). However, it is gratifying that the resulting time estimate is not an order of magnitude beyond expectations. Perhaps some additional sharing of information could reduce this total execution time into the range of 150 ms, at which point the time-course would be both reasonable and plausible.

7.2.3 Relationships to Other Processes in Vision

One of the basic premises in this work has been that the texture-based processes being investigated here are not the only mechanisms used by the visual system in most situations. But given the plethora of other mechanisms currently believed to underly visual perception, does the model developed here make sense in a larger framework containing other processes? In this section, I examine three specific issues: the possible relationships of this model to other mechanisms for segmentation and shape estimation, the relationships to memory and stored representations, and the interactions with visual attention.

Mechanisms for Segmentation and Shape Estimation Using Other Sources of Information

In Chapter 2, I mentioned that it is well-known that the human visual system uses many sources of information besides texture as a basis for segmenting visual scenes. These sources include contours, color, stereo disparity, motion, and many others. Likewise, the visual system uses many sources of information for estimating surface shape. These include the many depth cues (e.g., stereo disparity, accommodation, convergence, linear perspective, and others), as well as more direct sources of shape information such as contours and shading. In light of this, is it worth proposing a model of segmentation and shape estimation that uses only texture? And can the model proposed here be extended into a framework that uses these other sources of information?

There are at least two arguments for focusing exclusively on texture in the present research. First, understanding the processes of texture-based segmentation and shape estimation is an important issue in the broader context of understanding spatial layout processing. Texture is a pervasive phenomenon in the natural world, and the presence of texture is a powerful and reliable cue to the presence and location of a surface. It is more reliable than contours because the latter do not always indicate where a surface lies—consider, for example, the contours created by an object's shadow, which can trick a viewer into seeing a false surface boundary (Cavanagh, 1989). Further, texture is a source of information available when some of the other cues mentioned above are not. For example, stereo disparity is not available to all animals, nor to the more than 10% of the humans in Western countries (and higher proportions in Third World countries) that suffer from amblyopia, a partial or complete loss of sight in one eye (Watt, 1988).

A second argument is that any research must begin with *some* starting point. As expressed by the *small experiments* approach described in Chapter 1, it is easier to start small and build up than to start big and attempt to manage and understand the massive result. The present research can hopefully serve as a starting point for further work on spatial layout processing and integration of visual mechanisms.

Can the model proposed here be extended into a framework that incorporates other sources of information? In theory, the answer is yes. There is no intrinsic limitation that would prevent the integration of other visual mechanisms into the framework developed in this dissertation. Some types of visual processes could be added simply as extra modules that draw upon the results of the initial processing component described in Chapter 3. The complex cell model is generic and has no features specifically tailored towards texture—the simulated complex cells respond to texture, but do not have special enhancements *designed for* texture. Other visual mechanisms that begin with complex cell outputs could presumably start with the model of complex cells developed here. Similarly, the average peak frequency measures that serve as the starting point for both segmentation and shape estimation could potentially also serve as the starting point for other visual process.

The key problem in incorporating other mechanisms of segmentation and shape estimation is how to merge the results from these separate mechanisms. For example, adding a second mechanism for segmentation means that the results of *two* segmentation processes must somehow be combined (assuming that the human visual system does in fact combine such results). How should the results be placed into register? How should they be combined if the separate results disagree? Achieving this kind of integration is a difficult and open problem in vision research, and one that is not resolved in this dissertation.

Memory and Stored Representations

The segmentation and shape estimation processes modeled in this research all operate bottom-up: they make no use of stored representations or prototypes of common object shapes or scene elements to aid their operation. But humans do make use of internal representations. Such prototypes are essential to help fill in information that is not available in the raw visual input (see the results for test image B in Experiment Four, Chapter 6). Does this mean that the model is inappropriate?

The central issue is *when* do stored prototypes come into use when interpreting a visual input. In order for a stored prototype to be used, it seems logical that the visual system must make some initial interpretation of a visual input in order to activate representations in memory. Otherwise, it is not clear how the proper representation could be brought to bear on interpreting a given input. From this standpoint, the research in this dissertation is focused on the processes that take place before the activation of prototypes. That is, an initial estimate of certain aspects of spatial layout is *exactly the kind of information that could activate stored prototypes of common shapes.*

Thus, the model presented here is entirely compatible with the idea that stored prototypes of objects or scene elements can aid in segmenting a visual input and estimating the shapes of surfaces. The goal here has been to study preattentive processes that take place before initial memory activation; the kind of rough layout information produced by these processes is exactly the kind of trigger that would serve well for allowing past experience to influence more detailed spatial layout analysis.

Visual Attention

One of the roles of visual attention is limiting the amount of processing that the visual system must do. An attentional mechanism can guide processing to those aspects of the input that are especially salient, allowing the visual system to use its resources more effectively than if it had to perform all operations all the time over the entire input. Could a focus-of-attention component be added to the present model? Would it be beneficial?

There are actually two ways in which attentional mechanisms could interact with this model: (1) visual attention could simplify the work of the segmentation and shape estimation processes, and (2) the segmentation and shape estimation processes could help guide visual attention. In the first case, one simple way of incorporating a focus-of-attention mechanism into the present model would be the following. The attentional system could take the output of the segmentation component, evaluate the segmentation regions on the basis of qualities such as overall size, and then select one of the regions as the target for continuing shape estimation. The effect would be to focus processing on a particular region or surface of the visual input, reducing the amount of overall processing required on the part of the shape estimation component.

In the second case, an attentional mechanism could use the results produced by the present model as a source of information. Visual attention could guide other mechanisms (e.g., more detailed shape estimation) based on the approximate segmentation results, or based on the approximate shape estimates, or a combination of the two.

The model developed in this dissertation does not take advantage of the first kind of interaction, but a focus-of-attention mechanism could be added and it would offer benefits in terms of reducing overall computations. The second kind of interaction would be more appropriate in a larger model that incorporates other visual mechanisms besides texture-based segmentation and shape estimation. Both of these possibilities remain areas for future work.

7.2.4 Summary: Plausibility of the Model

In summary, the present model appears to be reasonable from the standpoint of the types of mechanisms required, although the time-course of processing is somewhat longer than it should be. The model could also interact with visual attention, and some form of attentional focusing could be added to the current framework. Overall, I believe the model provides a promising beginning and can serve as a stepping stone to further research in this area.

7.3 Remaining Problems

A presentation of a model would not be complete without an evaluation of its problems and a discussion of how these problems might be overcome. Table 7.2 on the following page lists five specific problems with the current formulation. These problems lead to symptoms that have manifested themselves throughout the experiments performed in this research.

Problem P1 (*spatial averaging performed as part of APF computation blurs borders*) refers to the trade-off inherent in using averaging as part of the computation of average peak frequencies for an input image. The averaging process blurs the responses of complex cells, including the responses near region borders. This smears the boundaries between different regions of texture in the image, making it more difficult for the segmentation process to accurately localize the region borders. The symptom of this problem is summarized as item S1 in Table 7.2: segmentation borders sometimes are not placed exactly at the locations of the true region boundaries. There is no way to avoid the problem: the averaging performed during APF computation *necessarily* blurs the complex cell responses. One can only reduce the degree of blurring by reducing the averaging radius, but this affects the smoothness of the shape estimates. This problem is undesirable, but on the other hand, the goal in this research has been to develop a model of *approximate* processing. The segmentation results do not need to be highly accurate, as long as they capture the general regions of texture in an input.

Problem P2 (*complex cell responses spill across textured/untextured region boundaries*) refers to a problem noted in Experiments One and Two: complex cells tuned to low spatial frequencies tend to exhibit responses relatively far from the true boundary between a textured and untextured region. This causes symptom S2 listed in Table 7.2. The problem is rooted in the way that complex cells respond to contrast borders. Section 7.4.1 below discusses an idea for overcoming this difficulty.

Table 7.2: Specific problems remaining with the components of the model

	Symptom		Underlying Problem
Segmentation Component	S1. Borders between textured regions are often not localized accurately	←	P1. Spatial averaging performed as part of APF computation blurs borders
	S2. Borders between regions of texture and nontexture in an image are often not localized accurately	←	P2. Complex cell responses spill across textured/untextured region boundaries
	S3. May fail for inputs having large range of spatial frequencies	←	P3. There are no mechanisms for dynamically rescaling parameter values
	S4. Particular set of fixed parameter values is not always best for all inputs to the segmentation process	←	
Shape Estimation Component	S5. Particular set of fixed parameter values is not always best for all inputs to the shape estimation process	←	
	S6. Estimates of surface shape are subject to the tilt ambiguity	←	P4. Shape estimation method relies on orthographic image projection only
	S7. Method cannot determine the relative depths between separate surfaces	←	
	S8. Method is sensitive to inhomogeneities in surface textures	←	P5. Shape estimation method is not sufficiently powerful
	S9. Surface shape renditions have flat areas	←	

Problem P3 (*there are no mechanisms for dynamically rescaling parameter values*) refers to an issue noted in Experiments Two, Three and Four. The problem is that it is sometimes impossible to find fixed parameter values that will work for all input images. This results in symptoms S3, S4 and S5. Section 7.4.2 discusses ideas for overcoming this difficulty.

Problem P4 (*shape estimation method relies on orthographic image projection only*) refers to the fact that the shape estimation model used in this research assumes that input images are generated using orthographic image projection. The model cannot make use of perspective cues. This reliance on a limited type of image projection causes symptoms S6 and S7. Section 7.4.3 below discusses ideas for overcoming this problem.

Finally, problem P5 (*shape estimation method is not sufficiently powerful*) concerns the difficulties exhibited by the shape estimation process on some test cases in Experiments Two and Four. The problem manifests itself in several ways, but two especially notable symptoms are S8 (the method is sensitive to surface texture inhomogeneities) and S9 (surface renditions have flat areas). The fundamental issue is that the shape estimation method used here is straightforward and works acceptably well for carefully controlled textures, but it may be *too* simple to produce good results on realistic textures. Natural textures are often not truly homogeneous, and this causes problems for the shape estimation approach. In addition, the flat areas visible in shape renditions in Experiments Two and Four further show that the method for integrating slant and tilt estimates could stand improvement.

7.4 Possible Approaches to Solving the Remaining Problems

From among the five problems discussed above and listed in Table 7.2, solving P1 and P5 would require significantly changing the framework of this model. However, the remaining problems could be solved in the existing framework. In this section, I discuss possible approaches to resolving these three problems, following ideas that have been alluded to elsewhere in this dissertation. The three problems are P2 (the spill-over of complex cell responses across textured/untextured region boundaries), P3 (the lack of mechanisms for automatically rescaling parameter values), and P4 (limitations in shape estimation due to using orthographic image projection).

7.4.1 Spill-Over of Complex Cell Responses

In Chapter 3, I first pointed out a problem with the behavior of complex cells at the borders between textured and smooth areas in an input image. The problem is that cells may respond far away from the true boundary of an area of texture, causing difficulties for the segmentation component discussed in Chapter 5.

Analysis of the Problem

The reason for the problem is illustrated in Figure 7.1 on the next page. Suppose an input consists of a single textured sphere centered on a blank background. The spatial frequency content in a small window at each point over the object will (correctly) shift towards higher frequencies out towards the edges of the object, because there the surface curves away from the viewer. But as shown in Figure 7.1(a), it turns out that complex cells tuned to low spatial frequencies also produce responses some distance away from the edges of the object, beyond the points where the high-frequency cells respond. The reason is

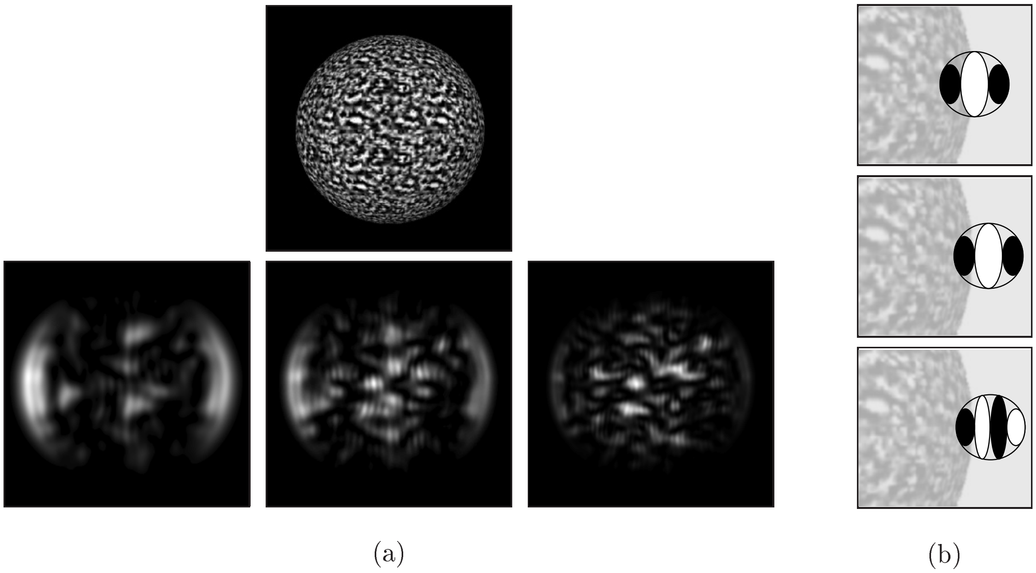


Figure 7.1: (a) When a textured surface is bordered by an untextured region, the responses of complex cells can extend past the edges of the surface into the blank space. This example shows, along the bottom row, the maps of complex cell responses for the three lowest peak spatial frequencies implemented in the present simulation. Note that all three extend past the edge of the surface shown in the top image. (b) Closeup of simple cell receptive fields at the edge of a textured sphere. The configuration of the excitatory and inhibitory regions of the receptive fields means that the cells may respond not only when their centers are at the actual borders of the surface (top), but also when they are farther away (middle and bottom).

simply that complex cells (and simple cells) respond to edge stimuli. In fact, some simple cells tuned to low spatial frequencies have receptive fields with only two lobes, modeled as a first-order Gaussian derivative, and these are optimally stimulated by a step edge in the image. But even cells modeled as higher-order Gaussian derivatives can be activated by edges to some degree. The problem caused by this behavior is most evident for cells tuned to very low spatial frequencies, because those cells have large receptive fields. The change in contrast caused by the border between a textured and untextured region is enough to trigger activity at the edges of these large receptive fields, even when the centers of the cells are relatively far away from the border, as illustrated in Figure 7.1(b).

The problem is not limited to images of single surfaces on a black background. It can occur even if the region surrounding a texture is carefully adjusted to have an intensity equal to the average intensity in the textured region. The surrounding region does not even have to be smooth; a finer texture surrounding a coarser texture will effectively act as a smooth region to cells tuned to low (coarse) spatial frequencies.

The problem *could* be alleviated by eliminating or ignoring the responses of the neurons tuned to the lowest spatial frequencies. However, this would merely shift the problem to a different band of frequencies. The reason for this is twofold: scenes can have texture/nontexture transitions at a variety of spatial scales, and complex cells at all spatial scales respond to edge stimuli. For instance, suppose that the two lowest spatial-frequency bands were removed from the model. Then, on a surface containing a fine texture that triggered activity in cells tuned only to high spatial frequencies, cells in the third-to-lowest frequency band would still respond past the borders of the textured surface in the image. This would again introduce confusing frequency components into later visual processing stages.

The problem is not that the low-frequency complex cells are responding incorrectly; they are functioning exactly as designed. It is simply that their responses, in the situations illustrated in Figure 7.1 on the preceding page, are undesirable for the functions to which they are being applied. Simple and complex cells are in fact not ideal for texture-based visual processing (Petkov & Kruizinga, 1997). They respond to contrast variations within their receptive fields, a necessary starting point for texture analysis. But they respond to too many types of variations, including step edges, single lines, and other nonrepetitive patterns that would not ordinarily be called "texture". A visual system that uses simple and complex cells must be able to make a further discrimination beyond the basic cell responses: it must decide, is it texture, or is it not?

Possible Solution: Combining the Responses of Multiple Complex Cells

The complex cells as currently designed have numerous desirable properties, notably their tuning properties for spatial frequency and orientation and their insensitivity to phase within the receptive field. Is it possible to keep those properties but modify or combine their responses somehow so that they are less likely to respond over areas devoid of texture in an image?

An inspiration can be found in a recently-identified type of complex cell nicknamed *grating cells* (von der Heydt, Peterhans, & Dürsteler, 1991; von der Heydt et al., 1992; von der Heydt, 1995; see also Glezer, Ivanoff, & Tscherbach, 1973). The defining property of grating cells is the absence or near-absence of responses for single light or dark bars and edges, and preference for many repeating bars, such as in a grating pattern. These have been modeled as being composed from many simple cells whose receptive fields are regularly spaced along an imaginary line (Kruizinga & Petkov, 1995; Petkov & Kruizinga, 1997; von der Heydt et al., 1991). The idea behind this type of model is that several simple cells must be activated in order for the overall unit, the grating cell, to produce a response. Grating cells may be important for some types of texture-based visual processing (Hine, Cook, & Rogers, 1997), but their propensity for highly periodic stimuli would seem to make them too narrowly focused on only certain types of textures. However, the idea of combining multiple, regularly spaced cells is worth investigating.

The simplest modification to the model developed in this dissertation would be to combine complex cell outputs in some fashion, and then use the result as the input to the later texture-based processing stages. Such a multicell unit should have the following desirable characteristics:

- It should retain as much as possible the spatial-frequency tuning properties of the underlying cells. This is because the responses of this new type of combined cell will need to serve as the input to the processes that estimate surface shape from texture distortion, and those processes rely on having a systematic correspondence between the cells responding to an input and the range of spatial frequencies for which those cells are tuned.

- It should, as much as possible, not respond over image areas that are devoid of texture. This implies that its responses to single light-dark edges should be minimal. Instead, the new unit should prefer stimuli that have several light-dark transitions.

- It should have a simply design that can clearly be implemented in neural circuits. This argues for straightforward mechanisms such as thresholds and logical AND/OR gates (Koch & Poggio, 1987; Shepherd & Koch, 1990), rather than elaborate circuits.

I will call this kind of combinational cell a *texture cell*, to emphasize its use in texture-based processing and to distinguish it from true grating cells. The requirements above, and the existing grating cell models of Petkov and Kruizinga (1997) and von der Heydt et al. (1991), suggest that texture cells could be constructed by combining the responses of multiple complex cells whose receptive fields are aligned, but separated by some distance, as illustrated in Figure 7.2 on the next page.

7.4.2 Fixed Parameter Values

In experiments Two and Three, one of the biggest problems was the fact that both the shape estimation and segmentation components use parameter values that remain fixed for all inputs. Although it is possible to find parameter settings for each component that work fairly well for most inputs, it appears to be impossible to find a set of fixed values that work for all inputs. What the model needs is a mechanism or set of mechanisms to rescale parameter values dynamically based on each input image.

There is ample precedent for dynamic rescaling of parameter values in biological visual systems. For example, in the retina there are mechanisms of light adaptation that

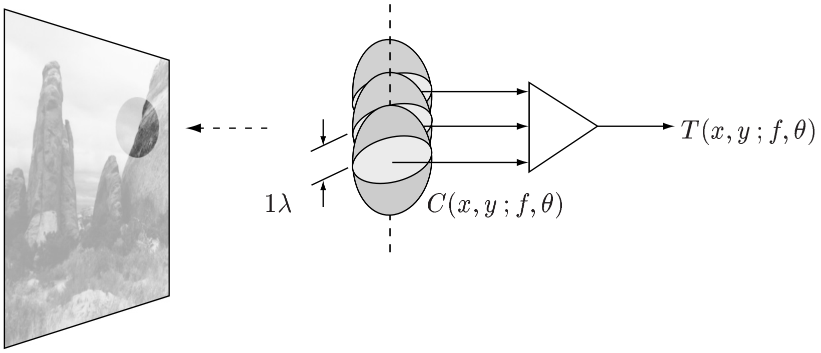


Figure 7.2: The proposed basic structure of a texture cell. Such a unit $T(x, y\,; f, \theta)$ with a preferred spatial frequency f and orientation θ takes its inputs from multiple complex cells having identical spatial-frequency and orientation tunings. The receptive fields of the complex cells (represented here as two-tone gray circles) are aligned in the direction perpendicular to the preferred orientation, and offset by one period of the preferred spatial frequency to which the complex cells are tuned.

modulate the sensitivity of the eye to light. As the level of illumination in a scene changes, the adaptation mechanisms adjust many parameters in the retina so that the range of retinal response values remains approximately the same (Kandel et al., 1991). In theory, this allows the brain to process information about the content of an input without being distracted by irrelevant details about the ambient light level. Another example of a dynamic scaling mechanism comes from the contrast-normalization model of complex cells used in this dissertation. The contrast-normalization process normalizes the outputs of complex cells based on the overall contrast in an input image (Chapter 3).

Thus, it seems entirely appropriate to extend the segmentation and shape estimation components with mechanisms that would rescale certain parameter values dynamically, based on their inputs. This would likely resolve problem P3 listed in Table 7.2 on page 307, and improve the performance of both the segmentation and the shape estimation components.

7.4.3 Orthographic Image Projection

The model of texture-based shape estimation described in Chapter 4 is based on an assumption of orthographic image projection. This simplifies the mathematics of the shape estimation method, but at the same time limits the power of the approach. While it is a convenient simplification for the purposes of developing a working model in this dissertation, ultimately the limitation needs to be addressed and the model extended to use full perspective projection.

Sakai and Finkel (1994, 1997) have developed extensions to their model of texture-based shape estimation that allows it to interpret perspective cues. Their approach is based on the following principle. If texture elements on a surface are compressed equally in all orientations, humans tend to perceive a surface receding in depth. Therefore, one way to detect the presence of perspective effects is to look for isotropic compression of texture. In areas where there is isotropic texture compression, the lateral inhibition mechanism for shape-from-texture (see Section 4.2.2) is disabled; instead, the surface shape is computed using a process based on calculating the variance of the normalized APF across all orientations (Sakai & Finkel, 1994, 1997).

Thus, there is good reason to believe that problem P4 of Table 7.2 can be solved. As a first step towards solving the problem, it would be appropriate to explore the addition of Sakai and Finkel's (1994, 1997) perspective cue extensions to the present model.

7.5 Predictions Arising From the Model

If the model described in this dissertation reflects the true nature of texture-based visual processing in the brain, certain predictions arising from the model should be experimentally verifiable. The following are five candidate predictions:

1. *Segmentations are available before shape estimates.* Although the processes related to shape estimation are begun first, the process of segmentation is finished before the final results of shape estimation are obtained. Thus, if the model is broadly consistent with how texture-based segmentation and shape estimation are done in the brain, experimental testing of human visual processing should confirm this.

2. *Segmentation involves a serial time-course of processing.* The parallel, cooperative processing used in the segmentation model goes from a messy segmentation to a final result over a certain period of time. It should be possible to observe this improvement in the segmentation results in the brain, perhaps by recording the activity of neurons, or using psychophysical techniques to observe the intermediate results in human visual processing.

3. *Shape estimation involves a serial time-course of processing.* Similar to segmentation, the shape estimation process also follows a time-course in which different mechanisms are invoked in succession. It should be possible to obtain evidence of such a succession of processes using psychophysical or neurophysiological experiments.

4. *The use of average peak frequency as the input to segmentation implies that the segmentation boundaries are inaccurate.* The process of computing APF values necessarily blurs the responses of complex cells across region boundaries. If texture-based segmentation in the human visual system actually relied on a measure such as APF, then it should be possible to demonstrate that without careful scrutiny, human observers cannot accurately localize the borders between regions of texture.

5. *The use of average peak frequency implies that the segmentation mechanism will fail to distinguish some textures that are too similar in their APF characteristics.* This prediction means that, if two textures have similar APF characteristics, the segmentation method will not be able to segregate them. Thus, if texture-based segmentation in the human visual system actually relied on a measure such as APF, then the visual system should fail to segregate certain textures that are similar in APF values. This should be straightforward to test using psychophysical techniques.

7.6 Future Directions

The model developed in this dissertation performs just two basic tasks in spatial layout analysis. The human visual system performs additional processing automatically and preattentively on each visual input. One of the future directions for the present line of research is to extend the model to account for some of these other capabilities.

Two such additional capabilities are estimating the locations of image regions in terms of their central tendency, and estimating the sizes of image regions. In this section, I offer evidence for the idea that the visual system does in fact compute these basic quantities. Although I do not implement these capabilities in this dissertation, I offer the evidence as background research in support of the overall theory of spatial layout processing sketched in Chapter 1.

7.6.1 Locations of Regions in Terms of Central Tendencies

Neurons in the early cortical areas such as V1 and V2 are retinotopically mapped; that is, if two cortical neurons have adjacent receptive fields, they will be found physically adjacent to each other in those areas. This implies that the early neural representations of regions and their locations are in retinal coordinates. In area V1, which has the most precise retinotopic map (Zeki, 1993), the visual system has direct information about the locations of stimulus components in retinal coordinates simply on the basis of the location of neural activity. That is, because a stimulus component will produce activity

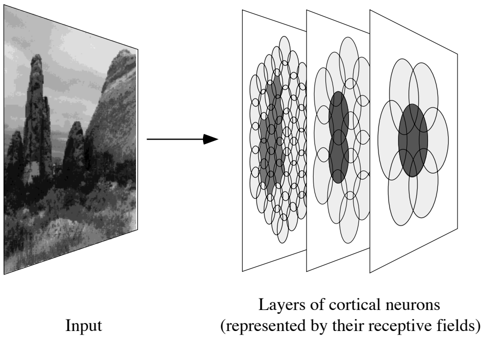

Input
Layers of cortical neurons
(represented by their receptive fields)

Figure 7.3: In some areas of the visual cortex, particularly the early areas such as V1 and V2, neurons are arranged retinotopically, with many neurons for each location in the visual field. The spatial scales to which the cells are sensitive vary from small to large. We can conceptualize these neurons as being arranged in overlapping layers, where a given layer consists of all the neurons responding to a particular spatial frequency and orientation. Each region of the visual field will stimulate a group of neurons. The ensemble of neurons responding to a particular visual region implicitly also acts as a representation of the location of that region in retinotopic coordinates. In this illustration, the neurons responding to one of the rocks in the image are shown highlighted in dark gray.

in a localized group of neurons in V1, the location of the component on the retina (and thus, indirectly, in the field of view) can be determined from the location of the activity in V1. This idea is illustrated in Figure 7.3.

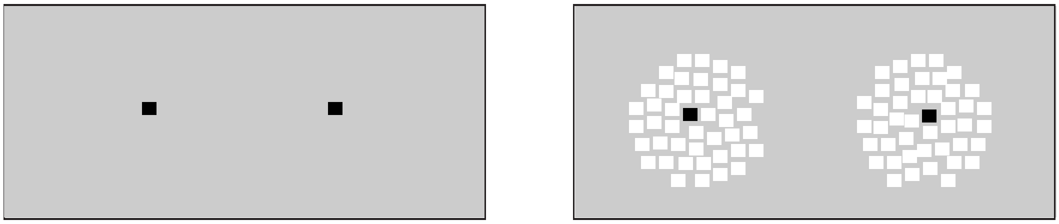
Although the representation of location information in V1 must involve something like this direct activity-based scheme, it has two difficulties. The first is that a great many neurons, typically numbering in the millions in area V1 of a higher primate, will respond to each region of the input. Somehow the activities of these many neurons must be related together, so that the system can determine *which* sets of neurons are responding to different regions of the input. This is known as the *binding problem* (or more accurately, *one* of the binding problems, since in cognitive science there are several) and although some theories have been proposed to explain the mechanisms at work (Milner, 1974; Singer, 1993; Eckhorn, Dicke, Kruse, & Reitboeck, 1991; Grannan, Kleinfeld, & Sompolinsky, 1993), no one knows how the brain actually does it. It remains an unresolved question in this research project.

The second problem is that the activity of a large group of neurons is not a useful representation for further processing. It would make sense if the visual system computed a more abstract representation of region locations. As it turns out, there *is* empirical evidence supporting the idea that the system computes the locations of simple figures automatically and preattentively (Baylis & Driver, 1993; Heathcote & Mewhort, 1993; Cohen & Ivry, 1991), and there is evidence that the location information is in the form of a measure of *central tendency* (Burbeck, 1991; Hess & Holliday, 1992; Hess, Dakin, &

Badcock, 1994; Hirsch & Mjolsness, 1992; Morgan et al., 1990; Vos, Bocheva, Yakimoff, & Helsper, 1993).

One line of evidence comes from experiments by Morgan et al. (1990). They presented human subjects with sequential displays of two stimuli, depicted in Figure 7.4. The first pair (the standard) consisted of two small reference squares, colored green and separated horizontally by about two degrees of visual angle. The second stimulus (the comparison) consisted of two clusters of squares, each made up from a large number of small squares. The actual centers of the clusters in the comparison stimulus were separated by the same amount as the standard stimulus on a given trial. Within each cluster, one of the small squares was colored green, and the exact position of this target square within the cluster was varied from trial to trial, so that the colored square could be either at the geometrical center of the cluster or to the left or right of it. The standard stimulus and the comparison stimulus were presented sequentially at the same location on a screen. The subjects were asked to compare the distance separating the two green target squares in the second stimulus of each trial with the green squares in the first stimulus of each trial. Morgan et al. then plotted the differences between the subjects' judgements of the distance and the actual distance against the displacement of the target squares from the cluster centers of the comparison stimulus. From this data, it was clear that their subjects were not comparing the distance between the target squares, but rather the distance between the *cluster centers*, despite that the subjects were explicitly trying to compare the distances between the target squares themselves. In other words, the subjects' judgements were strongly influenced by the centers of the clusters—the extraction of the centers seemed to be "automatic, effortless and unavoidable" (Morgan et al., 1990). In a second experiment, Morgan et al. used only single clusters of small squares, and tested their subjects' threshold for detecting the displacement of a colored square from the center of the cluster. The subjects were very accurate at inferring the centers. The authors concluded that this accuracy "is further evidence that mechanisms must exist for determining the centroid of textured patches" (Morgan et al., 1990, p. 1804).

Strictly speaking, a centroid is the geometric center of an area of homogeneous density. Because it stimuli used by Morgan et al. (1990) were closed and roughly homogeneous, it is not possible to determine whether their subjects really were using centroids or whether they were using some other form of central tendency. Other measures of central tendency are possible: the "center" of a region could be its geometric center, or the location of the mean of the intensity distribution, or the location of the peak of the intensity distribution composing the region, or some other measure (Burbeck, 1991; Hess et al., 1994). This issue has not been clearly settled; however, a number of researchers favor a *center-of-mass*



(a) Standard stimulus.

(b) Comparison stimulus.

Figure 7.4: Stimuli similar to those used by Morgan et al. (1990) in their experiments. (a) The standard stimulus, consisting of two green squares (shown here as black). (b) The comparison stimulus, consisting of two green squares within clusters of squares forming texture patches. Subjects were first shown the standard, followed by a blank screen, followed by the comparison stimulus, and were asked to compare the distance between the first two squares with the distance between the two green squares in the comparison stimulus. The positions of the green squares *within* the clusters in the comparison stimulus was varied to examine the effects upon their perceived separation.

measure of location. A center-of-mass definition of location is similar to a centroid, but takes account of possible uneven distribution of light intensity in a region. Experiments carried out by Vos et al. (1993) illustrate this. They presented subjects with figures shown one at a time for 200 ms on a screen, and after each stimulus presentation, asked the subjects to move a cursor to the perceived location of the figure that had appeared. The stimuli were irregularly shaped dot clusters, outlined polygons, and filled polygons. In all cases the positions that the subjects indicated were, with reasonable accuracy, located at the centers-of-mass of the figures and not at the centers of the areal shapes. Vos et al. also tested figures with irregular interior densities; in these situations, subjects' position judgements were biased toward the higher density regions, again consistent with a center-of-mass interpretation.

The idea that the visual system can extract the locations of figures in terms of a center-of-mass measure also makes sense from a computational standpoint. Watt (1988) has argued on computational grounds that a center-of-mass representation of stimulus locations is an effective way for the early visual system to represent the locations of image structures. It is extremely unlikely that the visual system represents the location of every individual intensity point in an input; it is computationally much more efficient to group intensity patterns into clusters or regions and choose reference points for the regions as wholes. Watt's model, MIRAGE (Watt, 1988; also see Watt, 1987; Watt & Morgan, 1983), codes position information hierarchically. MIRAGE begins by partitioning a visual input into groups of elements according to a clustering algorithm based largely on the proximity of elements. It derives the positions of the groups in a scene-based frame of reference, and then determines the positions of the elements in each group relative to the group's location. Watt's proposed hierarchical position coding model has recently received experimental support from psychophysical studies by Baylis and Driver (1993).

In sum, there is experimental evidence and theoretical grounding for the view that the visual system automatically determines the locations of stimulus regions. This location information appears to take the form of the centers-of-mass of the regions. Of course, more elaborate location information is also extracted by higher levels of the visual system; the idea here is simply that a basic aspect of early visual processing involves a general estimation of the center-of-mass of each visual region.

The model developed in this dissertation could be extended to extract this kind of location information. Once the segmentation component produces a labeling of the regions in the input image, an additional mechanism could estimate the approximate center-of-mass of each region. Such a hypothetical mechanism might consist of a network of detector units that take, as input, the output from the segmentation component. The role of the detector units would be to find the centers-of-mass of each segmentation region.

7.6.2 Sizes of Visual Regions

As already discussed, the earliest stages of visual analysis in the cortex display sensitivity to the spatial frequency of a stimulus. For a simple stimulus such as a unitary blob, this is effectively the same as being sensitive to its size. But there is another level at which size information plays a role in visual processing. For an organism attempting to survive in a rapidly changing environment, it is useful to be able to estimate quickly "how big" things are. Big things may be potential predators, small things potential prey. Moreover, navigation and reaching activities often exploit size information, and indeed, even object identification often depends on size. More generally, a component of perceiving the spatial arrangement of the environment involves assessing the general sizes of overall figures and spaces (Kaplan & Kaplan, 1982; Kosslyn et al., 1990; Lesperance, 1990). Since a given stimulus will lead to activity in many visually-sensitive neurons in cortical area V1 and beyond, there must be additional circuitry in the brain to derive overall figural size information from the collection of responses of individual, size-tuned mechanisms in the early visual cortex.

Behavioral evidence for the idea that the visual system automatically and preattentively computes the sizes of visual stimuli comes from many different sources. Foltz et al. (1984) performed experiments in which they presented human subjects with pairs of written object names, digit names, and numerals, and measured their reaction times to judging which item of each pair was the larger of the two. They told their subjects that the pictorial sizes of the stimuli would vary, but that they should ignore the difference and report which item was "conceptually larger." As shown in Figure 7.5, the sizes of the written names and numerals were varied: in the congruent condition, the sizes of the names were consistent with the relative sizes between the named objects or numerals (e.g., a smaller figure "1" and a larger "6"); in the incongruent condition, the sizes of the names were inconsistent with the relative sizes (e.g., a larger figure "1" and a smaller "6"); and in the same-size conditions, the figures were either both large or both small. Foltz et al. found that, in the congruent condition, subjects were faster and more often accurate at expressing which item was larger than in the same-size condition, whereas in the incongruent condition, they were slower than in the same-size condition and made more errors. Evidently, a perceptual comparison of the sizes of the figures was fast and obligatory for the subjects.

Another line of behavioral evidence comes from studies of the salience of different visual features to infants. A number of psychological experiments have examined how four to ten month old infants react to various transformations of simple visual stimuli such as a larger circle containing two smaller, filled circles inside of it (Kagan, Linn, Mount, Reznick, & Hiatt, 1979; Linn, Reznick, Kagan, & Hans, 1982), or a simple block shape (Linn, Hans, & Kagan, 1978). The results show that young infants are more sensitive to changes in the sizes of the stimuli than to a number of other transformations such as changes in shape. Again, this supports the idea that size assessment is a basic process.

At a lower level of visual processing, stimulus size affects the targetting of visual eye movements in humans. A person presented with a novel scene will automatically begin

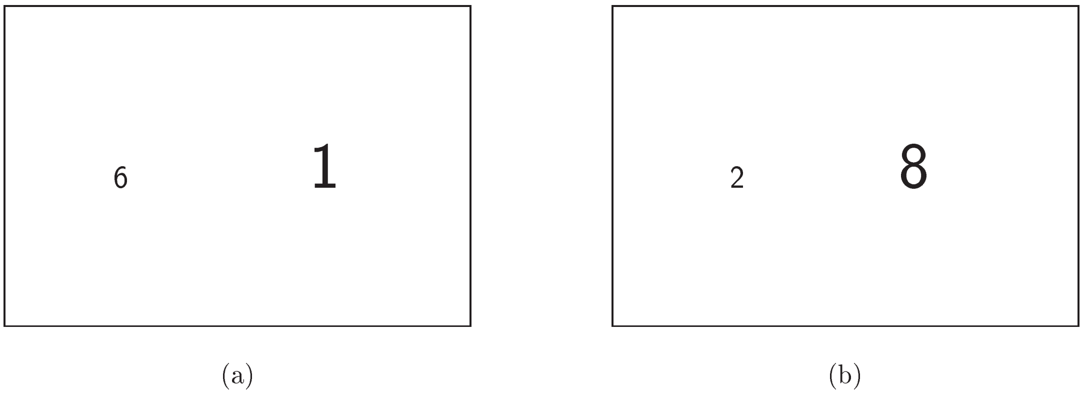


Figure 7.5: Example of stimuli used by Foltz et al. (1984). (a) An example of the incongruent condition: the difference in sizes of the numerals does not correspond to their relative difference in value. (b) An example of the congruent condition: the sizes of the numerals do correspond.

making eye movements to different locations. These *saccades* are a way for the visual system to focus processing on specific locations, to explore them in more detail using the high acuity regions of the eyes. Purely physical qualities of the visual input are not the only determinants of eye movements; what normally controls saccades is a combination of the raw visual features and cognitive factors exerting a top-down influence. However, clever experimental techniques have revealed some systematic relationships between physical characteristics of visual stimuli and the behavior of the saccade control system (Becker, 1991; Findlay, 1980; Findlay, Brogan, & Wenban-Smith, 1993; Menz & Groner, 1987). Work by Findlay (1980) and Findlay et al. (1993) has shown clear effects of target size on the eye movements of human subjects.

Thus it seems reasonable to assume that the visual system contains neural circuits that automatically and preattentively extract the sizes of different figural elements in the visual input. Kosslyn et al. (1990) have proposed that the computation of size is one of the functions implemented by the areas of the brain concerned with processing spatial information. They also argue that size and location are intimately related: “size can be conceived of as the number of small locations an object occupies”, and indeed, “local aspects of shape are no more than the distribution of locations occupied by small portions of an object ... ” (Kosslyn et al., 1990, p. 219).

The model developed in this dissertation could be extended along these lines by adding a mechanism that takes, as input, the regions delivered by the segmentation component and summarizes the size of each region in terms of the number of locations it covers. “Size” in this case would mean the extent of a region in a retinotopic frame of reference. Small regions would consist of relatively fewer locations than larger regions. Comparing the sizes of two region would be a simple matter of judging which one covers the greater number of locations.

7.7 Conclusions

This research offers a model of combined, texture-based segmentation and shape estimation, two components of approximate spatial layout processing in the visual system. The model begins with the responses of complex cells of the primary visual cortex, and combines a weak membrane/functional minimization approach to segmentation with a shape estimation method based on tracking changes in the average dominant spatial frequencies across a surface. Separately, the two tasks of segmentation and shape estimation have been studied extensively in vision research, but they have not previously been examined together in the context of a model grounded in neurophysiology and psychophysics.

I have shown that by integrating segmentation and shape estimation, a system can share information between these processes, allowing the processes to constrain and inform each other as well as save on computations. In support of the model, I have presented a software simulation that can perform texture-based segmentation and shape estimation on images containing multiple, curved, textured surfaces.

The research has drawn upon evidence and results from three subjects: the study of the behaviors of neurons in the visual cortex, the study of texture-based segmentation, and the study of texture-based shape estimation. Examining all three subjects together forces one to face not only the challenge of integrating different visual processes into one model, but also the challenge of simply making sense of the different languages and approaches used by researchers in the three subject areas. But it seems clear that the field of vision research must make a greater effort to explore the interactions between such separate visual mechanisms. After all, the human visual system can perform both texture-based segmentation and shape estimation in an integrated architecture, and if this capability is ever to be explained, the integration issue must be studied more closely.

Appendices

Appendix A

Background Review of Basic Concepts in the Study of Vision

This appendix presents a selective review of background material about visual perception, and in particular about the primate visual system, in order to summarize concepts that are central to the rest of this dissertation. The topics covered here include: spatial scale, spatial frequency, neural pathways in the early visual system, simple and complex cells, and spatiotemporal tuning. More extensive treatments are available in a number of excellent books on vision (e.g., Bruce et al., 1996; De Valois & De Valois, 1990; Jain et al., 1995; Zeki, 1993).

A.1 Spatial Scale

Spatial scale refers to size, and is intimately connected to the physical structure of the world and the problem of making observations about that structure. The concept of spatial scale can be illustrated using images of natural scenes such as those shown in Figure A.1. Different elements in an image give rise to patterns occurring at a variety of sizes, simply because objects in the world exist as meaningful entities at different sizes (Lindeberg & ter Haar Romeny, 1994). Pebbles and flowers, for example, are smaller than boulders and trees. The fact that the world is organized at many different spatial scales imposes certain constraints on a visual processing system.

In both artificial and natural vision systems, observations are made as measurements using devices that have finite-sized apertures—openings through which visual observations are made—and finite numbers of receptors that transform light into electrical or electrochemical signals. In both types of systems, the results of the measurements are analyzed by mechanisms that perform operations on the signals (Perona, 1995; Watt, 1991b). An example of an operation might be calculating the average value of the signals coming from a subset of the receptors. But before an operator can be applied to physical signals, the extent of the spatial neighborhood over which the operator samples the data must be

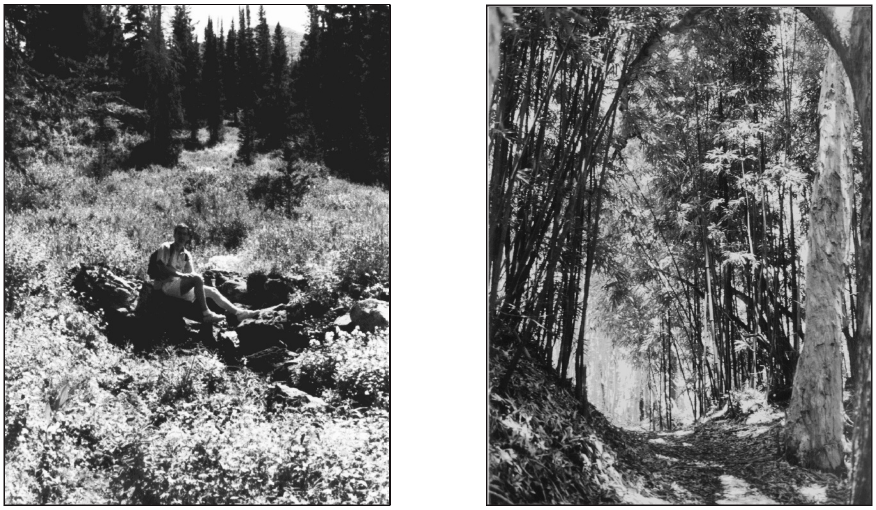


Figure A.1: Examples of natural scenes. Note the range of spatial scales at which visual structures exist in these examples. On the left, a very large collection of very small elements (flowers) composes a ground plane, and the slope of that ground plane leads to a gradient in the visible texture of the ground. On the right, closely spaced, vertically-oriented trees almost give the impression of walls on either side of the ground plane. At a finer spatial scale, the individual trees are easily discerned, and at a still-finer scale, the markings on individual trees are evident.

chosen. The extent of the neighborhood—the spatial scale of the operator—determines the number of samples that are used to perform the operation.

As the scale of an operator is increased, a greater number of samples is used in the calculation. Large scales are useful for operators that are meant to detect large trends—in other words, big visual structures that are only apparent over a large spatial extent. An important drawback is that increasing the scale (the neighborhood size) increases the number of places in the image where the neighborhood encompasses two or more separate subsets of the data, blurring the distinction between different visual structures (Watt, 1991b). Detail within the larger region encompassed by a large-scale operator is therefore lost. To accurately detect and localize smaller structures, it is necessary to use operators having smaller scales. However, small-scale operators have their own drawback: they cannot detect large-scale structures, because they do not encompass enough samples.

Because of this tradeoff, it is necessary to take measurements at a variety of spatial scales (Lindeberg & ter Haar Romeny, 1994; Marr, 1982). This implies the need for multiple mechanisms, each performing the same visual operation at a different scale, centered about each location in the visual input. We will see below that natural vision systems embody exactly this strategy by using mechanisms ("simple cells", "complex cells", and other types of visually responsive neural circuits) having "receptive fields" of many different sizes.

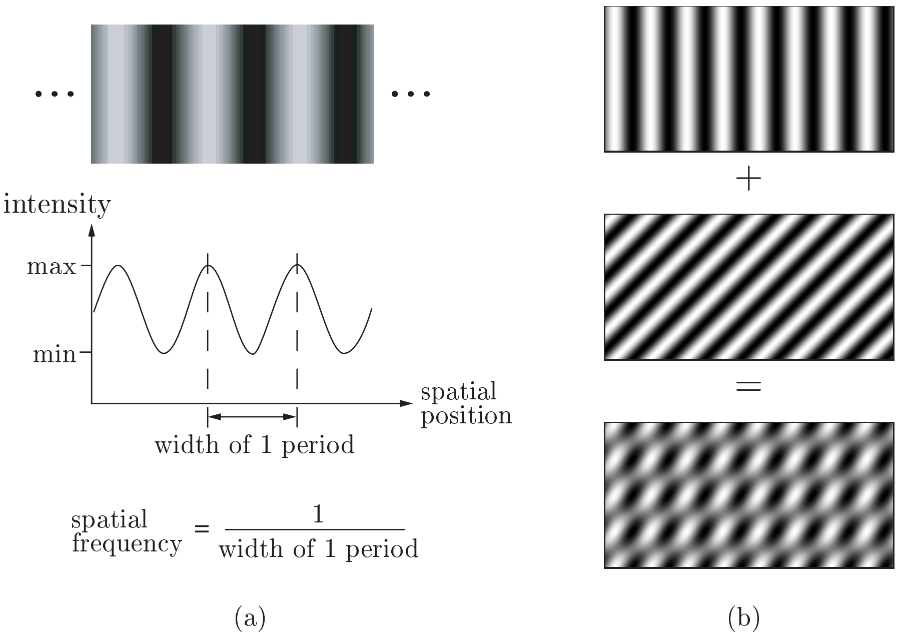


Figure A.2: (a) Illustration of the concept of spatial frequency. The top image shows a portion of a vertically-oriented, two-dimensional sinusoidal wave. The graph below it plots the luminance of the pattern as a function of spatial position. The number of cycles of the sinusoidal wave determines its spatial frequency, which is usually defined as the number of repetitions of a pattern per degree of visual angle. If the graph of luminance versus spatial position is scaled in terms of visual angle, then the reciprocal of one period of a periodic waveform gives its spatial frequency. So, for instance, a spatial frequency of 2 cycles per degree (c/deg) indicates a waveform that contains two periods of bright-dark change per degree of visual angle. (b) Simple example of a pattern resulting from superimposing two sinusoidal waves with identical frequencies but different orientations.

A.2 Spatial Frequency

Closely related to the concept of spatial scale is the concept of *spatial frequency*. It is illustrated in Figure A.2. A simple, repeating, two-dimensional pattern of parallel bright and dark bars can be described by several parameters: frequency, waveform, contrast, phase, and angle. *Spatial frequency* refers to the number of cycles of bright and dark bars per degree of visual angle; *waveform* refers to the shape of the pattern (e.g., a square wave or a sine wave); *contrast*, to the ratio of maximum to minimum light intensity in the pattern; *phase*, to the starting point of the pattern relative to some arbitrary fixed point; and *angle*, to the angle made between some arbitrary axis and a line perpendicular to the direction in which the pattern alternates. Figure A.2(a) illustrates a sine wave and the relationship between spatial frequency and the alternating bright and dark elements of the sinusoidal pattern.

A property of all two-dimensional patterns of light is that they can be decomposed into the sum of a set of more basic, component patterns. In particular, a two-dimensional pattern of light intensities, such as what gets projected onto a retina, can be constructed by superimposing a large number of sine-wave gratings that are scaled, stretched and rotated in different ways. Even a small number of simple patterns, when superimposed, can create more complex patterns; Figure A.2(b) shows how adding two sine-wave gratings of identical spatial frequencies but different orientations yields a textural pattern.

Note that describing a two-dimensional pattern of light in terms of a collection of sine wave gratings is merely another way of expressing the same pattern. It is only a transformation in the format of the description; in other words, a pattern may be described in *spatial terms*, as a collection of points having different intensity values, or it may be described in *spatial-frequency* terms, as a collection of sine wave gratings of different frequencies, contrasts, orientations, and phases. This kind of description is not restricted to entire images; any region larger than a single pixel can be described in spatial-frequency terms.

What is the relationship between this and spatial scale? By being sensitive to different spatial frequencies in the visual input, the operators used by a visual system can respond to different spatial scales. Imagine two operators applied to an image as illustrated in Figure A.3, one tuned to low spatial frequencies and the other to high spatial frequencies. The operator tuned to high spatial frequencies will respond to small-scale changes of bright and dark in the visual input, but will be insensitive to large-scale changes. Conversely, the operator tuned to low spatial frequencies will respond to large-scale changes in light intensity in the visual input, but will be too coarse to detect the small-scale changes within the large-scale changes.

There is a rich mathematical theory behind the concept of spatial frequency. It is based on an important analytical tool, the *Fourier transform* (Bracewell, 1986; Brigham, 1974). The Fourier transform is commonly used for assessing the properties of visual mechanisms and for describing filtering operations, a concept discussed further in Section A.6 below. Spatial-frequency analyses play a role in Chapter 3 in the development of a model of visually-sensitive neurons.

One more concept important to mention in the context of spatial-frequency analysis is *bandwidth*. This expresses the distance between two quantities on some scale. In vision research, bandwidth is most commonly expressed on a logarithmic scale using the expression

$$b = (\log f_h - \log f_l)/\log 2 \tag{A.1}$$

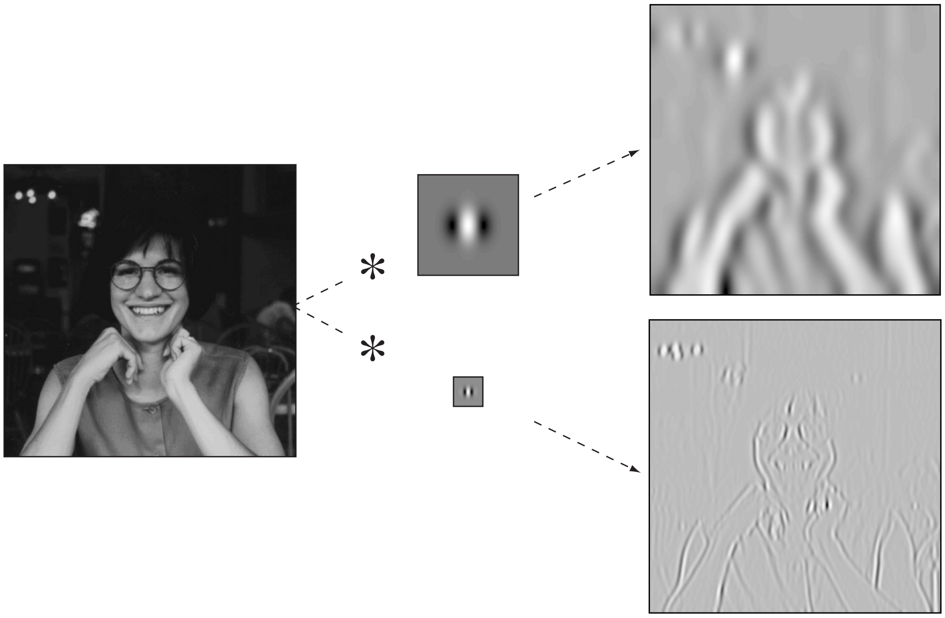


Figure A.3: Illustration of spatial scale and spatial frequencies. On the left is the original image. On the far right are two maps of response values displayed as shades of gray. Each pixel in each map represents the response of a filter applied to the original image at the corresponding location. Medium gray pixels represent zero, brighter pixels represent positive values, and darker pixels represent negative values. On the top right is the map of responses obtained by cross-correlating the image with a linear operator sensitive to low spatial frequencies; on the bottom right, the responses from cross-correlating the image with a linear operator sensitive to high frequencies. The linear operators are shown to scale in the middle. The top right map illustrates that filtering with a low-frequency filter picks out coarse-scale changes in intensity in the image, but misses fine details. Conversely, the bottom right map shows that filtering with a high-frequency filter picks out fine-scale details, but misses large-scale changes—large regions are reduced to zero response.

where f_h is the higher value, f_l is lower value, and b is the bandwidth in octaves (i.e., ratios of two to one.). Bandwidth is commonly used to express how narrowly tuned a mechanism is for some quantity such as spatial frequency. For example, if a mechanism is said to have an upper cut-off of 8 c/deg and a lower cut-off of 4 c/deg, then its spatial-frequency bandwidth is one octave. (Note that both f_h and f_l must be nonzero for Equation A.1 to make sense.)

A.3 Basic Neural Pathways and Neural Responses in the First Stages of the Visual System

The output signals from the retina of each eye are produced by a set of neurons called the *retinal ganglion cells*, and in humans and primates these signals are sent to several brain regions for processing. Each retinal ganglion cell produces an output that depends

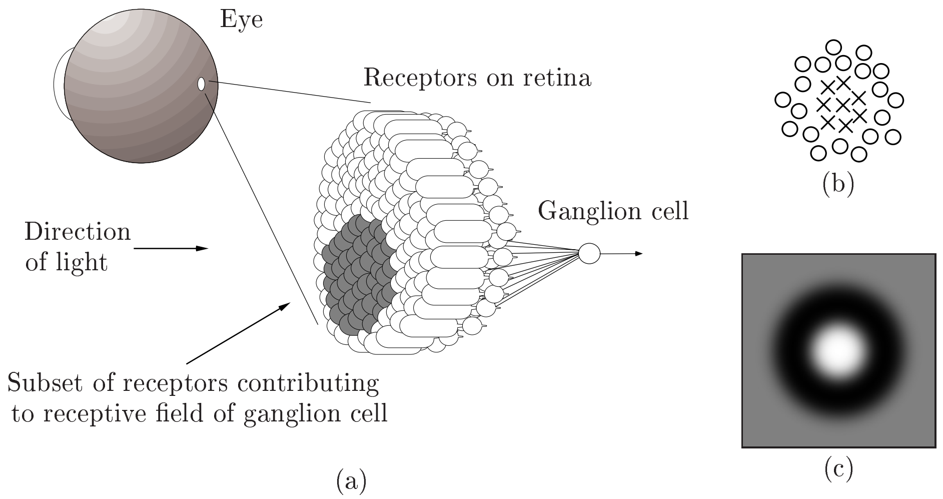

Eye
Receptors on retina
Ganglion cell
Direction of light
Subset of receptors contributing to receptive field of ganglion cell
(a)
(b)
(c)

Figure A.4: (a) Simplified illustration of the receptive field of a retinal ganglion cell. Such a neuron receives input from a number of receptors, and generates an output signal based on the strengths of the different inputs it receives. (In reality, the connection between receptors and ganglion cells is indirect; there are intermediate neurons between the receptors and the ganglion cell which for clarity are not shown in this diagram.) (b) If, in the area of the ganglion neuron's receptive field, the retina were stimulated with bright spots of light at different locations and the response of the ganglion cell plotted according to whether the spots increased the neuron's output signal or decreased it, one might obtain a map similar to that shown here. X's indicate the locations of spots that lead to excitation of the neuron, and O's indicate the locations that lead to inhibition. (c) If the receptive field of the neuron were mapped more carefully, the excitatory and inhibitory subregions could be shown to vary smoothly in strength. Here, bright areas indicate subregions where spots of light excite the ganglion neuron, and dark areas indicate subregions where spots inhibit the neuron. The ganglion cell depicted here would be called an *on-center* neuron, because its center is excitatory; there also exist *off-center* neurons with mirror-image arrangements of excitatory and inhibitory subregions. It is also worth noting that the type of ganglion cell depicted here is a so-called *P cell*; such cells comprise the majority of ganglion cells in primates, but other cells exist that do not have such neatly organized, circular receptive fields. (Kandel et al., 1991; Zeki, 1993)

on the light that falls within a particular region of the retina. This area of the retina where light can influence the activity of the neuron is called the *receptive field* of the neuron, illustrated in Figure A.4. When stimulated using spots of light flashed on and off in the visual field, the receptive fields of most ganglion neurons can be shown to have a circular organization. There are two subregions in the example receptive field shown in Figure A.4(b–c): a central area in which a spot of light will excite the neuron and increase its activity, and another, surrounding, antagonistic area in which a spot of light will inhibit the neuron and reduce its activity. Some retinal ganglion neurons have an excitatory central zone surrounded by an inhibitory annulus (such as that shown in Figure A.4); some have the reverse receptive field organization.

Approximately 90% of the ganglion outputs are projected to the cortex of the brain through an intermediate structure in the mid-brain called the lateral geniculate nucleus (LGN), shown in Figure A.5. Based on available evidence, the LGN seems to be a complex switching station (Crick & Koch, 1992; Kandel et al., 1991; Koch, 1987). Somewhat less than 10% of the total ganglion outputs are projected to a structure called the superior colliculus (Perry & Cowey, 1984). The remaining ganglion outputs go to several intermediate structures that handle such functions as controlling the pupils of the eyes in response to overall changes in brightness (Kandel et al., 1991), and beyond having these simple functions, these structures are not believed to play a role in the visual information processing that leads to awareness of space and objects. Figure A.5 summarizes some of these visual pathways.

The *superior colliculus* receives direct connections from the retinas as well as the cortex (Robinson & McClurkin, 1989), and in turn sends outputs to many areas, including the visual cortex. Portions of the superior colliculus are involved in the control of eye movements and visual orienting behaviors (Hepp, van Opstal, Straumann, Hess, & Henn, 1993; Kandel et al., 1991; Robinson & McClurkin, 1989; Sparks & Jay, 1987). In many non-primate species such as rats and cats, it is well-established that the superior colliculus plays a dominant role in visual processing (Rodman, Gross, & Albright, 1990; Trevarthen, 1968). In primates, the cortex is believed to be much more important to vision, but it is possible that the colliculus and an adjacent structure called the *pulvinar* serve a greater information-processing role than is usually attributed to them. For example, it is likely that the superior colliculus performs a general analysis of spatial layout as part of its role in visual orienting behaviors (Hughes, 1982; Lesperance, 1990; Trevarthen, 1968); there is also evidence that the pulvinar may be involved in assessing salience of visual stimuli and in attentional processes (Marrocco & Li, 1977; Robinson & McClurkin, 1989; Robinson & Petersen, 1992; Van Essen et al., 1992). However, the bulk of the processing that leads to visual perception in higher primates and humans is believed to take place in the cortex itself (Hubel, 1988; Van Essen et al., 1992; Zeki, 1993), and in the present research I focus entirely on the cortex.

The portion of the cortex receiving the bulk of the signals from the retina via the LGN is area *V1* ("visual area 1"). As shown in Figure A.5, V1 is located at the back of the brain. It is only one of many regions in the cortex involved in processing visual signals. It is also an area that contains a complete map of the retinal surface: adjacent points on the retina (and consequently in the visual field) are connected in an orderly way to adjacent points in V1, preserving the local geometry. This is known as *retinotopic mapping.* The neural circuits in V1 are believed to perform comparatively simple, stereotyped operations

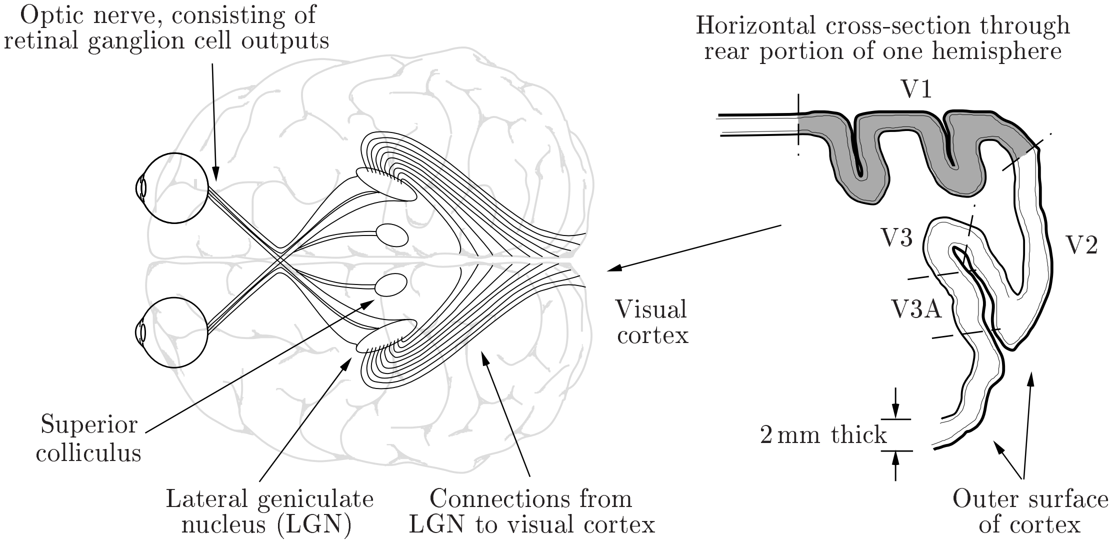

Optic nerve, consisting of retinal ganglion cell outputs
Horizontal cross-section through rear portion of one hemisphere
V1
V3
V2
V3A
Visual cortex
Superior colliculus
2 mm thick
Lateral geniculate nucleus (LGN)
Connections from LGN to visual cortex
Outer surface of cortex

Figure A.5: A cross-sectional view of a human brain, illustrating the neural connections from the eyes to the LGN, superior colliculus, and the visual cortex. The diagram on the right is a horizontal cross-section through the rear portion of one hemisphere of the brain, again seen from above. The convoluted curve is the cortical sheet, consisting of neurons and other cells. Area V1 is a portion of the cortex lying toward the inner rear of the brain; V2 is adjacent to it; V3 is a region of cortex adjacent to V2; and so on for other areas (Hubel, 1988; Zeki, 1993).

on every localized part of the visual input—every location in the visual field is simultaneously analyzed by a variety of neurons whose receptive fields overlap. Neurons in V1 are selectively responsive to the position, size, shape, motion, color and eye of presentation of visual stimuli (Carandini, Heeger, & Movshon, 1995). From there, signals are passed to and received from other visual areas of the brain: V2, V3, and others.

A.4 Simple and Complex Cells in Area V1

Retinal ganglion neurons are sensitive to location simply by virtue of being responsive only to a limited region of the visual field. They are also sensitive to size; a spot of light that is larger than the center of an on-center ganglion cell's receptive field will stimulate its inhibitory surround and thus inhibit the cell's activity.

Neurons in cortical area V1 build on these properties by aggregating the outputs of retinal ganglion neurons, via the LGN, in ways that produce properties not found in ganglion cell outputs. One of these properties is sensitivity to orientation. Hubel and Wiesel (1959, 1962) were the first to demonstrate that most neurons in V1 do not respond well to circular spots of light: the cells respond better to elongated stimuli such as bright lines at particular orientations. Hubel and Wiesel (1959) found that most neurons can be classified as either *simple cells* or *complex cells*, so named because the simple cells have simpler response properties than the complex cells. A simple cell receptive field is illustrated in Figure A.6.

When tested using stimuli such as bright and dark lines, *simple cells* can be identified on the basis of showing the following properties (De Valois & De Valois, 1990; Hubel & Wiesel, 1962): (1) distinct excitatory and inhibitory subregions within their receptive field; (2) summation within both the excitatory and the inhibitory subregions, so that the more of the subregion that a stimulus covers, the stronger its excitatory or inhibitory effect; (3) antagonism between the excitatory and inhibitory subregions, meaning that a stimulus that covers part of an excitatory subregion without covering an inhibitory subregion will cause the neuron to produce a stronger signal than if the stimulus also simultaneously covers part of an inhibitory subregion; and (4) sensitivity to the orientation of the stimulus. The fact that simple cells satisfy the first three criteria means that they exhibit *linearity of spatial summation*: their response to a stimulus pattern is a weighted sum of their responses to the individual components of the stimulus pattern.

Complex cells fail this simple test of linearity. To a first approximation, complex cells can be distinguished from simple cells by the absence of sensitivity to the absolute position of a stimulus within the receptive field—they have no discrete excitatory and inhibitory subregions. A stimulus can fall anywhere within the receptive field, and (as long as the stimulus is not so wide that it covers most of the field), the neuron will still respond to it, unlike simple cells, which respond only if the stimulus is also in a particular position within the receptive field (De Valois & De Valois, 1990; Hubel & Wiesel, 1962). Further, complex cells are also insensitive to contrast polarity, meaning that they respond whether the stimulus is brighter or darker than the background (Ohzawa, DeAngelis, & Freeman, 1990). These properties cannot be implemented by linearly summing the outputs of discrete excitatory and inhibitory subregions within a receptive field; the neural mechanisms must include nonlinear operations, such as something equivalent to the mathematical operation of absolute value. The significance of this is discussed in the next section.

A.5 Spatial Frequency Sensitivity in Simple and Complex Cells

A neuron that responds to a spot or line can also be expected to respond to a sine-wave grating, since a grating is simply a repeating set of lines or elongated spots. And indeed, simple and complex cells are responsive to spatial-frequency content, something

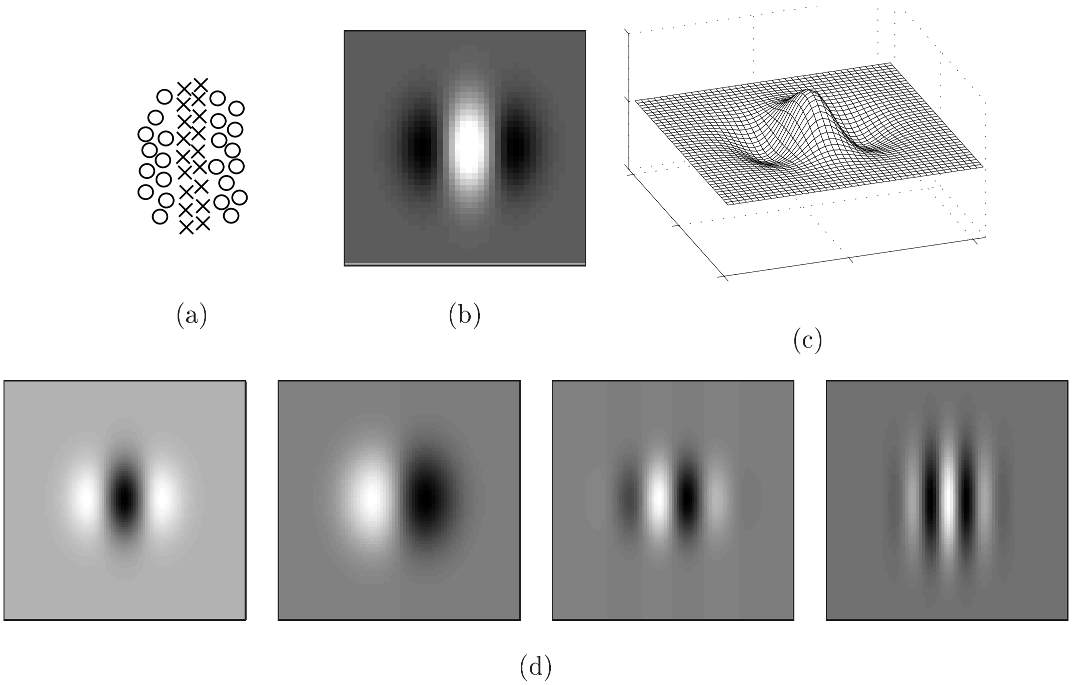


Figure A.6: Whereas most retinal ganglion neurons have circular receptive fields, most cells in the cortex do not. (a) If the receptive field of a simple cell is stimulated with bright spots of light at different locations, and the neuron's response is plotted according to whether the spots increase or decrease the neuron's output, one might obtain a map similar to that shown. (b) If the receptive field were mapped more carefully, one might obtain a map similar to that shown here. Bright areas indicate subregions that excite the simple cell, dark areas indicate subregions that inhibit the cell. (c) A plot of the same receptive field as a three-dimensional surface, in which inhibitory regions are regions of negative sensitivity and the central excitatory region is a region of positive sensitivity. (d) Simple cell receptive fields actually come in a wide variety of shapes and sizes, some examples of which are depicted here. They also come in all orientations around the clock (not shown).

first demonstrated by Campbell et al. (1969). To understand the principle behind spatial-frequency tuning in a visual neuron, imagine the light from a sine-wave grating pattern falling on a simple cell's receptive field. As described above, a linear neuron's response is a function of the sum of the stimulus elements that fall in the different subregions of its receptive field. If the field happens to be positioned such that the bright portions of the sine-wave stimulus fall on excitatory subregions and the dark portions fall on inhibitory subregions, the neuron will be maximally stimulated and will respond with a vigorous neural signal. Conversely, if the stimulus position is not well-matched to the receptive field, or it is misaligned with respect to the orientation of the receptive field, or the spatial frequency of the grating pattern is too high or too low (meaning that the sizes of the alternative bright and dark bars are too high and too low, respectively), the cell will not respond as vigorously.

In the visual neuroscience literature, cortical cell responses are often reported in terms of spatial-frequency quantities. Four of the most important quantities are the following:

Peak spatial frequency. Most simple and complex cells exhibit a preference for a particular spatial frequency. This is the frequency of a sine wave grating that excites the cell most strongly. Different cells are tuned for different frequencies.

Spatial frequency bandwidth. If the spatial frequency of a sine wave stimulus is not exactly matched to a cell's preferred frequency, the cell will not respond as strongly. The degree of selectivity for frequency is quantified in terms of the cell's spatial-frequency bandwidth. This is the distance in octaves of frequency between the upper and lower spatial frequencies at which the response falls to half of its maximum.

Preferred orientation. Most simple and complex cells respond best when the stimulus is oriented at a particular angle with respect to some reference direction, such as vertical. This is the cell's preferred orientation.

Orientation bandwidth. Similar to spatial-frequency bandwidth, the orientation bandwidth of the cell is the distance in degrees of angle between the points on either side of the preferred orientation at which the cell's response drops to half of its maximum.

There are many other parameters that describe the responses of simple, complex, and other cortical cells, including temporal properties and sensitivies to color. For the purposes of this research, I will not consider most of these other parameters. [This is in fact a strong simplification; it is known, for example, that the receptive field properties of many cortical cells change rapidly over the course of a stimulus presentation (Gaska et al., 1994; Young & Lesperance, 1993). Unfortunately, incorporating additional dimensions into this research would increase the complexity of the work to unmanageable proportions.]

A.6 Spatiotemporal Filtering and Energy Models

The approximate linear spatial summation in simple cells has lead to the view that they act as *linear spatiotemporal filters* whose outputs are modulated by nonlinearities. The filtering analogy comes from mechanical sieves and optical filters, devices that let pass certain quantities and reject others (Braddick, Campbell, & Atkinson, 1978). A spatiotemporal filter in a visual system takes a spatial and temporal distribution of light, measures the similarity of that local distribution of light to the structure of its receptive field, and produces a neural signal as output (Adelson & Bergen, 1991; Jones & Palmer, 1987a; Shapley & Lennie, 1985). In this view, the receptive field is a weighting function, and the response of a linear filter is a function of the weighted sum of stimulus intensities falling onto the filter's receptive field (Heeger, 1992b). Thus, simple cells do not explicitly encode the presence of features such as edges, nor particular image parameters such as orientation or spatial frequency; rather, they analyze *local image structure* in a way that encompasses *both* feature detection and spatial-frequency analysis (Koenderink & van Doorn, 1990; Young & Lesperance, 1993). The duality between the spatial and spatial-frequency domains means that local image structure can also be described in spatial-frequency terms. Consequently, the neuron's response is simultaneously a description of the spatial-frequency content of the local area as well as its spatial content. This notion plays a prominent role in Chapters 2 and 4.

The validity of this model depends on whether simple cells really are linear. In fact, overall they are not; they exhibit a number of nonlinearities (e.g., Heeger, 1992b, 1992a, 1993; Jacobson, Gaska, Chen, & Pollen, 1993; Spitzer & Hochstein, 1985b). For example, cortical neurons have a low maintained rate of firing—in the absence of stimulation, their signal is nearly zero. Unlike an idealized linear filter, they cannot have a negative response, and therefore they cannot directly signal pure inhibition. But the simple linear model can be extended to account for many of these problems. For instance, the output of a real simple cell can be assumed to consist of a linear spatial summation step followed by a *rectification* process, so that what would be negative values in an idealized filter are set to zero in the real neuron (Palmer, Jones, & Stepnoski, 1991). In general, many experiments do support the view that the spatial summation of light across the receptive field *is* linear, and that the nonlinearities are manifestations of *other* components of the overall mechanism that leads to a simple cell's final response (DeAngelis, Ohzawa, & Freeman, 1993; Heeger, 1992a; Palmer et al., 1991). The simple linear model can be amended to take these various nonlinear components into account while maintaining linear spatial summation in the receptive field (Heeger, 1992a, 1992b). This is an important benefit because it makes it possible to predict a simple cell's response to an input from knowledge of its response to some set of primitive patterns.

As mentioned above, complex cells, unlike simple cells, do not have distinct excitatory and inhibitory subregions within their receptive fields, which precludes a description in terms of a sum of purely linear mechanisms. However, they are selectively tuned for orientation and spatial frequency, just as simple cells are. How can this be?

Neurophysiological evidence supports the view that complex cell receptive fields are composed of subunits, or overlapping regions of sensitivity, and further, these subunits appear to be linear. In other words, while distinct inhibitory and excitatory subregions

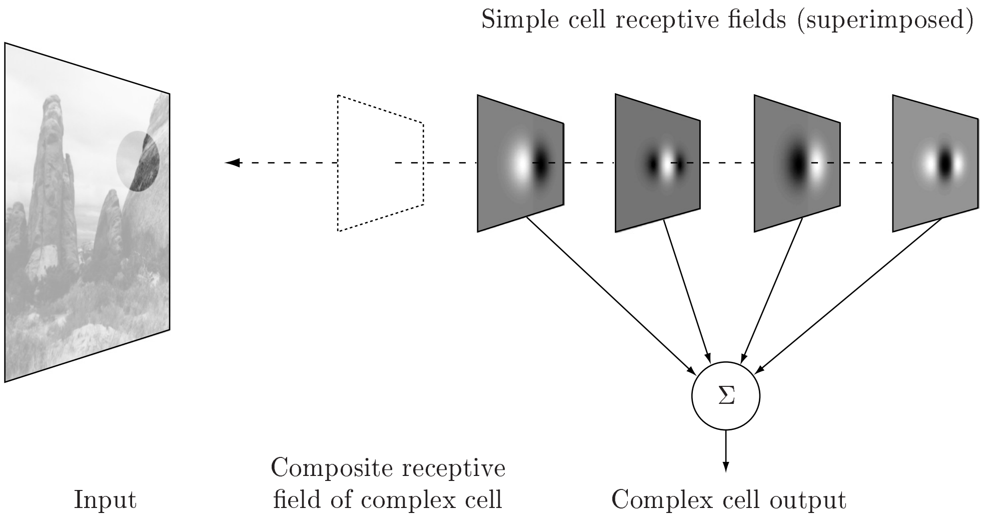

Simple cell receptive fields (superimposed)
Σ
Input
Composite receptive field of complex cell
Complex cell output

Figure A.7: An abstract and simplified model of a complex cell. The complex cell sums the outputs of four simple cells, and its receptive field is the composite of the superimposed simple cell fields. Each simple cell receptive field is of a different variety: two of them have *even* symmetry, meaning that that the peak of a subregion falls in the middle of the receptive field and the field is symmetrical about its cross-section, and two have *odd* symmetry, meaning that the field is not symmetrical about its cross-section. For each two such fields, the individuals also differ in whether the center of the receptive vield is excitatory or inhibitory.

cannot be mapped out in the same way as they can be in simple cells, there nevertheless do appear to be multiple such regions, arranged in an overlapping manner and summed together. This has lead to one popular class of functional models of complex cells that uses the combination of the outputs of a number of simple cells (Adelson & Bergen, 1985; Gaska et al., 1994; Heeger, 1991, 1992a, 1992b; Heitger et al., 1992; Pollen & Ronner, 1983; Spitzer & Hochstein, 1985a, 1988). The simple cells that form the subunits of the model complex cell are assumed to have receptive fields positioned at the same location in visual space, but with different configurations of excitatory and inhibitory subregions; thus a stimulus at any point within the collective receptive field will lead to activity in at least one of the neurons. The simple cells' outputs are summed together so that the overall network responds to either bright or dark stimuli in any position within the receptive field. A version of this model is illustrated in Figure A.7.

This model does have its failings: some complex cells have properties that are not present in simple cells and cannot be derived from a summation of simple cell outputs, and there is evidence that some complex cells may have direct inputs from the LGN (De Valois & De Valois, 1990). However, the simple cell summation scheme has so far turned out to be a reasonable working model (Heeger, 1992a, 1992b). The model is discussed in more detail in Chapter 3.

Appendix B

Detailed Methods

This appendix provides mathematical derivations and other details of certain formulas and procedures used in the rest of this dissertation.

B.1 Derivation of Spatial-Frequency Relationships for the Gaussian Derivative Receptive Field Model

The tuning characteristics of the Gaussian derivative receptive field model are calculated from the Fourier transform of the function. As given in Chapter 3, the Fourier transform of the Gaussian derivative function $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$ is

$$\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta) = k_n(j2\pi u)^n e^{-2\pi^2(\sigma_x^2 u^2 + \sigma_y^2 v^2)}, \tag{B.1}$$

where $j = \sqrt{-1}$ and k_n, n, σ_x and σ_y are parameters discussed in Chapter 3. The three tuning properties of interest are: the preferred spatial frequency, the spatial-frequency bandwidth, and the orientation bandwidth. These need to be related to the parameters n, σ_x and σ_y.

The simplest property to consider is the preferred spatial frequency. This is the spatial frequency at which $\mathcal{G}(u, v\,; \sigma_x, \sigma_y, \theta)$ reaches a maximum value for a particular combination of the other parameters. The preferred spatial frequency is measured at the preferred orientation of the receptive field; therefore, it is enough to consider a cross-section at $v = 0$ through the Fourier transform of a Gaussian derivative with $\theta = 0$. Further, as mentioned in Section 3.2.2, the imaginary factor j can be ignored in this situation because it only affects only the phase of the response and not the preferred spatial frequency. This reduces the equation for $\mathcal{G}(u, v\,; \sigma_x, \sigma_y, \theta)$ to

$$\mathcal{G}_n(u, 0\,; \sigma_x, \sigma_y, \theta) = k_n(2\pi u)^n e^{-2\pi^2 \sigma_x^2 u^2}. \tag{B.2}$$

This function will reach a maximum value where its first derivative goes to zero (Young, 1985). Taking the derivative, setting it to zero and solving for the value of $u = f_p$ gives

$$f_p = \frac{\sqrt{n}}{2\pi\sigma_x}. \tag{B.3}$$

This means that the preferred spatial frequency of a Gaussian derivative is a direct function of the derivative order n and σ_x. Solving this equation for σ_x gives Equation 3.10 on page 62.

Finding the remaining spatial-frequency parameters is somewhat more complicated. Experimental approaches to measuring spatial-frequency properties of cortical neurons are limited to observing how the cells attenuate the relative amplitudes of different sine-waves used as stimuli (Jones & Palmer, 1987a). In other words, the experiments deal only with the amplitude spectra of the cells. As explained in Chapter 3, the contrast-normalization model of simple cells assumes that the overt, experimentally-measured frequency properties reflect the squared responses of receptive field operators (c.f. Equation 3.3 on page 46). This means that the experimental results do not inform us directly about the amplitude spectrum of $\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta)$; rather, they inform us about the *squared* amplitude spectrum, $|\mathcal{G}_n|^2$. The presence of squaring affects the maximum amplitude of a Gaussian derivative function, as well as its spatial-frequency bandwidth and orientation bandwidth.

The maximum amplitude is found by substituting the expression for $u = f_p$ from above into Equation B.1 on the page before. Assuming a squared amplitude spectrum,

$$\max|\mathcal{G}_n|^2 = k_n^2 \left(\frac{n}{\sigma_x^2 e}\right)^n. \tag{B.4}$$

To proceed with finding an expression for the spatial-frequency bandwidth, we can begin again by fixing the orientation of the receptive field to lie along the x-axis, so that $v = 0$. The problem is now to find the higher and lower spatial frequencies at which $|\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta)|^2$ falls to one-half of its maximum amplitude. The maximum value of the function will come at $u = f_p$, the frequency given by Equation B.2. Let f_h be the spatial frequency at which the function falls to one-half its maximum at the high frequency end, and f_l be the frequency at which it falls to one-half its maximum at the low end. These two frequencies will be related by $f_h/f_l = 2^b$, where b is the frequency bandwidth.

It would be ideal if it were possible to solve analytically for f_h and f_l in terms of the other parameters, but unfortunately this turns out to be impossible: attempting to solve $[\mathcal{G}_n(f_h, 0\,; \sigma_x, 0, 0)]^2 = \frac{1}{2}[\mathcal{G}_n(f_p, 0\,; \sigma_x, 0, 0)]^2$ for f_h, for example, results in an expression containing both f_h and its logarithm. Instead, it is necessary to use a numerical approach.

The Gaussian derivative happens to have the property that the spatial-frequency bandwidth b is a function only of the derivative order n (Young, 1985). This leads to the following method for finding b corresponding to different values of n:

1. Pick convenient values for σ_x and σ_y, such as $\sigma_x = 1$ and $\sigma_y = 0$.

2. Find f_p using Equation B.3 and $\max|\mathcal{G}_n|^2$ using Equation B.4.

3. Numerically find the value of f_h that makes the following expression go to zero:

   $$|\mathcal{G}_n(f_h, 0\,; 1, 0, 0)|^2 - \tfrac{1}{2}|\mathcal{G}_n(f_p, 0\,; 1, 0, 0)|^2, \tag{B.5}$$

   using as a preliminary guess $f_h \approx f_p\, 2^{b/2}$. For the results shown in this dissertation, I used an algorithm that is a combination of bisection, secant and inverse quadratic interpolation (MathWorks, 1998b).

4. Numerically find the value of f_h that makes the following expression go to zero:

   $$|\mathcal{G}_n(f_l, 0\,; 1, 0, 0)|^2 - \tfrac{1}{2}|\mathcal{G}_n(f_p, 0\,; 1, 0, 0)|^2, \tag{B.6}$$

   using as a preliminary guess $f_l \approx f_p\, 2^{-b/2}$.

5. The spatial-frequency bandwidth is then given by $b = \log_2(f_h/f_l)$.

The method above permits tabulating b as a function of n, as shown in Table 3.1 on page 64. Going in the reverse direction—finding a derivative order n corresponding to a desired bandwidth—can be done by looking up the closest value (in the least-squares sense) from the table.

The final quantity of interest, σ_y, can be found as follows. First, translate the u, v parameters to polar coordinates using

$$\begin{aligned} u &= f\cos\theta, \\ v &= f\sin\theta. \end{aligned} \tag{B.7}$$

The orientation bandwidth of a receptive field is twice the angle subtended by the half-maximum outline of its frequency amplitude response. In other words, it is the twice the angle θ found by solving

$$\tfrac{1}{2}[k_n(2i\pi f)^n e^{-2\pi^2\sigma_x^2 f^2}]^2 = [k_n(2i\pi f\cos\theta)^n e^{-2\pi^2 f^2(\sigma_x^2\cos^2\theta+\sigma_y^2\sin^2\theta)}]^2 \tag{B.8}$$

given σ_x, σ_y, and the frequency f at which the orientation bandwidth is measured. Normally, $f = f_p$. Solving the equation above for σ_y and substituting the frequency gives

$$\sigma_y = \sqrt{\frac{\ln\sqrt{2} + n\ln(\cos\theta)}{2\,\pi^2 f_p^2 \sin^2\theta} + \sigma_x^2}, \tag{B.9}$$

which is Equation 3.12 on page 64. Thus, given the angle θ (measured from $u = 0$) at which the squared amplitude response should drop to one-half its maximum, and given σ_x, f_p and n, the equation above gives the necessary σ_y.

B.2 Procedure for Approximating the Hilbert Transform of a Gaussian Derivative

In the design of receptive fields discussed in Chapter 3, I make use of the fact that the Hilbert transform of an nth-order Gaussian derivative function can be approximated by a Gaussian derivative of order $n+1$. To make the approximation as close as possible, the derivative of order $n+1$ must be fitted to the shape of the true Hilbert transform. In this section, I describe the procedure that I used for performing the numerical fits using the space-domain form of the Gaussian derivative and its Hilbert transform.

Strictly speaking, the mathematical Hilbert transform is defined for functions of one real variable, but the Gaussian derivative functions being used here are two-dimensional. Fortunately, the two-dimensional Gaussian derivative is separable into the product of two functions, one along the x-axis and one along the y-axis (Young & Lesperance, 1993). By convention, the function along the x-axis is usually chosen to be the one whose derivative is taken; the function in the y-axis direction is a plain Gaussian. The Hilbert transform of this product is then the transform of the Gaussian derivative in the x-direction, multiplied by a plain Gaussian in the y-direction. Because of this separability, the process of fitting $G_{n+1}(x, y\,; \sigma_x, \sigma_y, \theta)$ to the Hilbert transform of $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$ can be simplified by performing it in the x- and y-directions separately. First, one fits a one-dimensional Gaussian derivative $G_{n+1}(x\,; \sigma_x)$ to the x profile of the transform of $G_n(x\,; \sigma_x)$, then one adjusts the parameters of another Gaussian, $G_0(y\,; \sigma_y)$, in the y direction to produce the desired orientation tuning in the final product, $G_{n+1}(x\,; \sigma_x)G_0(y\,; \sigma_y)$. An additional simplification is to consider only functions oriented along the x-axis; that is, with $\theta = 0$.

For the fitting process, Hilbert transforms of $G_n(x\,; \sigma_x)$ are needed for the range of derivative orders to be used in the simulation of complex cells. Ronse (1993, 1995) notes an additional property of the Hilbert transform that is useful in this context: the transform commutes with the derivative. This means that the Hilbert transform of the nth Gaussian derivative $G_n(x\,; \sigma_x)$ is the same as the nth derivative of the transform of the plain Gaussian, $G_0(x\,;\sigma_x)$, which simplifies the task of generating the transform functions.

Given expressions for the Hilbert transforms, the rest of the procedure is as follows:

1. Choose the values of n, σ_x and σ_y for the Gaussian derivative $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$ that forms the first operator in a given quadrature pair of receptive fields. The values are typically based on the desired spatial-frequency properties of the receptive field.

2. Create a discrete representation of the Hilbert transform of $G_n(x\,;\sigma_x)$ using the chosen values of n and σ_x. This one-dimensional profile can be created by sampling the values of the transform function at a large number of points along the x-axis.

3. Perform a numerical fit of the cross-section of $G_{n+1}(x\,;\sigma_x)$ to the discrete representation of the Hilbert transform of $G_n(x\,;\sigma_x)$. This is a nonlinear, least-squares curve-fitting problem in one parameter, σ_x; the parameter σ_y is irrelevant during this step. The best-fitting version of $G_{n+1}(x\,;\sigma_x)$ establishes the value of σ_x.

   The data-fitting process can be implemented in a number of ways. I used two methods for every case: the Levenberg-Marquart method and the Gauss-Newton method (MathWorks, 1998b; Press, Teukolsky, Vetterling, & Flannery, 1992). Both are general-purpose least-squares data fitting procedures. Computer programs for implementing them are widely available from a number of sources, including the Matlab environment (MathWorks, 1998b, 1998a). In order to reduce the chances of having the minimization procedures fall into local minima, I performed a sequence of minimizations for each case:

   (a) One run of the data-fitting process using the Levenberg-Marquart method, picking as a starting value of σ_x the value of σ_x used for $G_n(x\,;\sigma_x)$;

   (b) A second run using Levenberg-Marquart with a starting value of σ_x equal to the result from the first run;

   (c) Several additional runs using Levenberg-Marquart with a starting value of σ_x equal to the result of the first run modified by alternately adding or subtracting small random numbers;

   (d) One run using the Gauss-Newton method and a starting value of σ_x equal to the result from the first run of Levenberg-Marquart;

   (e) Several additional runs using Gauss-Newton and the result from the first run of Levenberg-Marquart modified by alternately adding or subtracting small numbers.

4. Calculate the true preferred frequency of the resulting Gaussian derivative function using Equation 3.10 on page 62 with $n = n + 1$ and σ_x as determined from the previous step. The resulting preferred spatial frequency may not be equal to the preferred frequency of $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$; however, it should be close, within a fraction of a cycle per degree.

5. Calculate the value of σ_y using Equation 3.12 on page 64 and the following parameters: the true orientation bandwidth of $G_n(x, y\,; \sigma_x, \sigma_y, \theta)$, the derivative order $n = n+1$, the value of σ_x calculated from the fitting process, and the true preferred spatial frequency of the fitted function (found in the previous step).

An alternative to performing a numerical fit in the space domain is to do it in the spatial-frequency domain. This follows from the fact that G_n and its quadrature partner should have identical Fourier amplitude spectra; thus, it is possible to fit the amplitude spectrum of G_{n+1} to that of G_n to find the parameter values. This approach does work, but in practice, I have obtained better results by performing the fits on the space-domain representations. The sum of the quadrature partners in the space domain tends to be smoother than when the quadrature partners are created by fitting the spatial-frequency domain representations.

B.3 Procedure for Adjusting Gaussian Derivative Magnitudes to Achieve Spatial-Frequency Tiling

In Section 3.3.2, I describe a method for finding values for the scaling parameter k_n for each Gaussian derivative receptive field in a simulation. The approach consists of first decomposing k_n into a product of two terms, such that $k_n = K_{\mathcal{G}_n}(n, \sigma_x)K_f(f_p)$. The value of $K_{\mathcal{G}_n}(n, \sigma_x)$ is determined analytically, but $K_f(f_p)$ must be found numerically. This section details the procedure that I used to find the $K_f(f_p)$ values.

When implementing a collection of receptive field mechanisms, the derivative order n and variances σ_x and σ_y for each receptive field become functions of the preferred frequency f_p and phase ϕ of each field. Let these functions be represented by $n(f_p, \phi)$, $\sigma_x(f_p, \phi)$, and $\sigma_y(f_p, \phi)$. Also let $\hat{\mathcal{G}}(u, v\,; f_p, \theta, \phi)$ be a rewritten form of the amplitude spectrum of $\mathcal{G}_n(u, v\,; \sigma_x, \sigma_y, \theta)$, such that

$$\hat{\mathcal{G}}(u, v\,; f_p, \theta, \phi) = (2\pi u_r)^{n(f_p,\phi)} K_{\mathcal{G}_n}[n(f_p, \phi), \sigma_x(f_p, \phi)]K_f(f_p)\, e^{-2\pi^2[\sigma_x(f_p,\phi)^2 u_r^2 + \sigma_y(f_p,\phi)^2 v_r^2]} \tag{B.10}$$

where

$$\begin{aligned} u_r &= u\cos\theta - v\sin\theta, \\ v_r &= u\sin\theta + v\cos\theta. \end{aligned} \tag{B.11}$$

In these formulas, f_p is the preferred frequency of the Gaussian derivative, θ is its orientation, ϕ is the phase, and u and v index the spatial frequencies in the x- and y-axis directions, respectively. Equation B.10 is essentially the same as Equation 3.9 on page 61, with the addition of a parameter allowing for arbitrary rotations of the field, and the imaginary term j removed.

The goal is to find the values of the factors $K_f(f_p)$ for each preferred spatial frequency implemented in the simulation. As explained in Section 3.3.2, one approach is to consider a slice at $v = 0$ through the sum of all the receptive fields' frequency spectra. The goal is to make the sum be as close as possible to a constant, one, everywhere in the range of spatial frequencies covered by the simulation.

To do this, first let f_s be the sampling frequency of the inputs. (In the present simulation, this is 64 pixels per degree.) Also let w be the width of an input image. In the Fourier transform of the receptive fields and the images, the frequency sampling rate is then given by $\Delta f = f_s/w$. Let f_l be the lowest preferred frequency implemented in the simulation, and f_h be the highest preferred frequency implemented in the simulation. (These are 1.0 c/deg and 22.6 c/deg, respectively, in the present work.) Now let $\mathbf{u}$ be a vector of all frequencies in the x-axis direction from f_l to f_h at intervals of Δf; i.e.,

$$\mathbf{u} = [\ f_l \quad f_l + \Delta f \quad f_l + 2\Delta_f \quad \cdots \quad f_h\ ] \tag{B.12}$$

The procedure consists of adjusting the values of the different $K_f(f_p)$ factors in Equation B.10 above so that the following quantity is minimized:

$$\sum_{\mathbf{u}} \left[1 - \sum_{f_p} \sum_{\theta} \sum_{\phi} \hat{\mathcal{G}}(u, 0\,; f_p, \theta, \phi)^2 \right]^2 \tag{B.13}$$

The inner summation is taken over all the preferred frequencies, preferred orientations, and phases to which the receptive fields in the simulation are tuned. Minimizing the quantity above is a nonlinear, least-squares curve-fitting problem in one parameter, $K_f(f_p)$. This data-fitting process can be implemented in a number of ways. I followed a variation of the procedure described in Section B.2 above, using a combination of the Levenberg-Marquart and the Gauss-Newton methods (MathWorks, 1998b; Press et al., 1992).

In order to reduce the chances of having the curve-fitting process fall into a local minimum, I performed a sequence of variations:

1. One run of the data-fitting process using the Levenberg-Marquart method, picking as a starting value $K_f(f_p) = 1$ for all values of f_p;
2. Two additional runs using the Levenberg-Marquart method, with starting values of $K_f(f_p)$ equal to the result from the first run plus or minus 0.25;
3. Several additional runs using Levenberg-Marquart with the different $K_f(f_p)$'s set to random numbers chosen from a uniform distribution in the range 0 to 1.0;
4. One run using the Gauss-Newton method and starting values for the $K_f(f_p)$'s equal to the results from the first run of Levenberg-Marquart;
5. Two additional runs using Gauss-Newton, with starting values of $K_f(f_p)$ equal to the previous result plus or minus 0.25;
6. Several additional runs using Gauss-Newton with the different $K_f(f_p)$'s set to random numbers chosen from a uniform distribution in the range 0 to 1.0;

I chose the values of the $K_f(f_p)$'s that produced the least squared error from among all the runs described above. This gave the set of values shown in Table 3.4 on page 86.

B.4 Bilinear Interpolation Formula for Slant Integration

In Section 4.3.4, I discuss the use of interpolation for overcoming a rectangular bias during the slant integration step of the shape estimation model. The formula I use for this interpolation is one given by Pratt (1991), modified to work specifically on the map of $\bar{O}_{3s}(x, y)$ values computed in Stage Four of the shape estimation model.

The formula for operator *diagval* used in Equation 4.29 on page 154 is

$$\begin{aligned}\mathrm{diagval}(x, y, p, q) = (1 - a)[(1 - b)\bar{O}_{3s}(x, y) + b\,\bar{O}_{3s}(x + \operatorname{sign} d_x, y)] \\ + a[(1 - b)\bar{O}_{3s}(x, y + \operatorname{sign} d_y) + b\,\bar{O}_{3s}(x + \operatorname{sign} d_x, y + \operatorname{sign} d_y)],\end{aligned} \tag{B.14}$$

where

$$d_x = p - x \tag{B.15}$$

$$d_y = q - y \tag{B.16}$$

$$d = |p - x| \tag{B.17}$$

$$a = \frac{d}{\sqrt{2}} - \left\lfloor \frac{d}{\sqrt{2}} \right\rfloor \tag{B.18}$$

$$b = \frac{d}{\sqrt{2}} - \left\lfloor \frac{d}{\sqrt{2}} \right\rfloor \tag{B.19}$$

The formulas above implicitly assume that (p, q) is a point lying on a diagonal from (x, y).

References

Abramowitz, M., & Stegun, I. A. (Eds.). (1970). *Handbook of mathematical functions with formulas, graphs, and mathematical tables (9th printing).* Washington: U. S. Government Printing Office.

Adelson, E. H., & Bergen, J. R. (1985). Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America A*, *2*(2), 284–299.

Adelson, E. H., & Bergen, J. R. (1991). The plenoptic function and the elements of early vision. In M. S. Landy & J. A. Movshon (Eds.), *Computational models of visual processing* (pp. 3–20). Cambridge, MA: MIT Press.

Aho, A. V., Hopcroft, J. E., & Ullman, J. D. (1983). *Data structures and algorithms.* Reading, MA: Addison-Wesley Publishing Company.

Akutsu, H., & Legge, G. E. (1995). Discrimination of compound gratings: Spatial-frequency channels or local features? *Vision Research*, *36*(19), 2685–2695.

Albrecht, D. G., & Hamilton, D. B. (1982). Striate cortex of monkey and cat: Contrast response function. *Journal of Neurophysiology*, *48*, 217–237.

Aloimonos, J. (1988). Shape from texture. *Biological Cybernetics*, *58*, 345–360.

Aloimonos, J., & Swain, M. (1985). Shape from texture. In *Proceedings of the international joint conference on artificial intelligence* (pp. 926–931). Los Angeles, Calif.

Azencott, R., Wang, J.-P., & Younes, L. (1997). Texture classification using windowed Fourier filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(2), 148–153.

Bajcsy, R. (1973). Computer description of textured surfaces. In *Proceedings of the international joint conference on artificial intelligence* (pp. 572–579). Morgan Kaufmann Publishers.

Bajcsy, R., & Lieberman, L. (1976). Texture gradient as a depth cue. *Computer Graphics and Image Processing*, *5*, 52–67.

Baylis, G. C., & Driver, J. (1993). Visual attention and objects: Evidence for hierarchical coding of location. *Journal of Experimental Psychology: Human Perception and Performance*, *19*(3), 451–470.

Beck, J. (1966a). Effect of orientation and of shape similarity on perceptual grouping. *Perception & Psychophysics*, *1*, 300–302.

Beck, J. (1966b). Perceptual grouping produced by changes in orientation and shape. *Science*, *154*, 538–540.

Beck, J. (1967). Perceptual grouping produced by line figures. *Perception & Psychophysics*, *2*(11), 491–495.

Beck, J., & Gibson, J. J. (1955). The relation of apparent shape to apparent slant in the perception of objects. *Journal of Experimental Psychology*, *50*(2), 125–133.

Beck, J., Sutter, A., & Ivry, R. (1987). Spatial frequency channels and perceptual grouping in texture segregation. *Computer Vision, Graphics and Image Processing*, *37*(2), 299–325.

Becker, W. (1991). Saccades. In R. H. S. Carpenter (Ed.), *Eye movements* (pp. 95–137). Boca Raton, Florida: CRC Press.

Bergen, J. R. (1991). Theories of visual texture perception. In D. Regan (Ed.), *Spatial vision* (pp. 114–133). CRC Press.

Bergen, J. R., & Adelson, E. H. (1988). Early vision and texture perception. *Nature*, *333*, 363–364.

Bergen, J. R., & Landy, M. S. (1991). Computational modeling of visual texture segregation. In M. S. Landy & J. A. Movshon (Eds.), *Computational models of visual processing* (pp. 254–271). MIT Press.

Bhandarkar, S. M., & Zeng, X. (1997). Figure-ground separation: A case study in energy minimization via evolutionary computing. In M. Pelillo & E. R. Hancock (Eds.), *Energy minimization methods in computer vision and pattern recognition (EMMCVPR '97)* (pp. 375–390). Springer-Verlag.

Bhushan, N., Rao, A. R., & Lohse, G. L. (1997). The texture lexicon: Understanding the categorization of visual texture terms and their relationship to texture images. *Cognitive Science*, *21*(2), 219–246.

Bilbro, G. L., Snyder, W. E., Garnier, S. J., & Gault, J. W. (1992). Mean field annealing: A formalism for constructing GNC-like algorithms. *IEEE Transactions on Neural Networks*, *3*.

Black, M. J., & Rosenholtz, R. (1995). Robust estimation of multiple surface shapes from occluded textures. In *IEEE international symposium on computer vision, Miami, Florida.*

Blake, A. (1983). The least-disturbance principle and weak constraints. *Pattern Recognition Letters*, *1*, 393–399.

Blake, A., Bülthoff, H. H., & Sheinberg, D. (1993). Shape from texture: Ideal observers and human psychophysics. *Vision Research*, *33*(12), 1723–1737.

Blake, A., & Marinos, C. (1990). Shape from texture: Estimation, isotropy and moments. *Artificial Intelligence*, *45*, 323–380.

Blake, A., & Zisserman, A. (1986a). Invariant surface reconstruction using weak continuity constraints. In *Proceedings of the conference on computer vision and pattern recognition* (pp. 62–67).

Blake, A., & Zisserman, A. (1986b). Some properties of weak continuity constraints and the GNC algorithm. In *Proceedings of the conference on computer vision and pattern recognition* (pp. 656–661).

Blake, A., & Zisserman, A. (1987). *Visual reconstruction.* Cambridge, MA: MIT Press.

Blakemore, C., & Campbell, F. W. (1969). On the existence of neurones in the human visual system selectively sensitive to the orientation and size of retinal images. *Journal of Physiology*, *203*, 237–260.

Blostein, D., & Ahuja, N. (1989). Shape from texture: Integrating texture-element extraction and surface estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *11*(12), 1233–1251.

Bonds, A. B. (1989). Role of inhibition in the specification of orientation selectivity of cells in the cat striate cortex. *Visual Neuroscience*, *2*, 41-55.

Bovik, A. C. (1991). Analysis of multichannel narrow-band filters for image texture segmentation. *IEEE Transactions on Signal Processing*, *39*(9), 2025–2043.

Bovik, A. C., Clark, N., & Geisler, W. S. (1990). Multichannel texture analysis using localized spatial filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(1), 55–73.

Bracewell, R. N. (1986). *The Fourier transform and its applications* (Second, revised ed.). New York: McGraw-Hill.

Braddick, O., Campbell, F. W., & Atkinson, J. (1978). Channels in vision: Basic aspects. In R. Held, H. W. Leibowitz, & H.-L. Teuber (Eds.), *Perception* (Vol. VIII, pp. 1–38). Berlin: Springer-Verlag.

Braunstein, M. L. (1976). *Depth perception through motion.* New York: Academic Press.

Brigham, E. O. (1974). *The Fast Fourier Transform.* Prentice-Hall.

Brodatz, P. (1966). *Textures: A photographic album for artists and designers.* New York: Dover.

Brown, L. G., & Shvaytser, H. (1988). Surface orientation from projective foreshortening of isotropic texture autocorrelation. In *Proceedings of the conference on computer vision and pattern recognition* (pp. 510–514). IEEE Computer Society Press.

Brown, L. G., & Shvaytser, H. (1990). Surface orientation from projective foreshortening of isotropic texture autocorrelation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(6), 584–588.

Bruce, V., Green, P. R., & Georgeson, M. A. (1996). *Visual perception: Physiology, psychology and ecology* (3rd ed.). East Sussex, U.K.: Psychology Press.

Buckley, D., Frisby, J. P., & Spivey, E. (1990). Stereo and texture cue integration in ground planes: An investigation using the table stereometer. *Perception*, *20*, 91.

Burbeck, C. A. (1991). Encoding spatial relations. In R. J. Watt (Ed.), *Pattern recognition by man and machine* (pp. 8–18). Boca Raton, Floriday: CRC Press.

Burt, P. J. (1981). Fast filter transforms for image processing. *Computer Graphics and Image Processing*, *16*, 20–51.

Burt, P. J. (1984). The pyramid as a structure for efficient computation. In A. Rosenfeld (Ed.), *Multiresolution image processing and analysis* (pp. 6–35). Berlin: Springer-Verlag.

Burt, P. J., & Adelson, E. H. (1983). The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, *COM-31*(4), 532–540.

Caelli, T. M. (1985). Three processing characteristics of visual texture segmentation. *Springer-Verlag*, *1*, 19–30.

Caelli, T. M. (1993). Texture classification and segmentation algorithms in man and machines. *Springer-Verlag*, *7*(4), 277–292.

Caelli, T. M. (1995). A brief overview of texture processing in machine vision. In T. V. Papathomas, C. Chubb, A. Gorea, & E. Kowler (Eds.), *Early vision and beyond* (pp. 79–87). Cambridge, MA: MIT Press.

Campbell, F. W., Cooper, G. F., & Enroth-Cugell, C. (1969). The spatial selectivity of the visual cells of the cat. *Journal of Physiology*, *203*, 223–235.

Campbell, F. W., & Robson, J. G. (1968). Application of Fourier analysis to the visibility of gratings. *Journal of Physiology*, *197*, 551–556.

Carandini, M., & Heeger, D. J. (1994). Summation and division by neurons in primate visual cortex. *Science*, *264*, 1333–1336.

Carandini, M., Heeger, D. J., & Movshon, J. A. (1995). *Linearity and gain control in V1 simple cells.* (Submitted as a chapter to *Cerebral Cortex, vol. X: Cortical Models*, Plenum Press, ed. E. G. Jones and P. S. Ulinski. Also available on the World Wide Web: `http://white.stanford.edu:80/~heeger/publications.html`.)

Casadei, S., Mitter, S., & Perona, P. (1992). Boundary detection in piecewise homogeneous textured images. In G. Sandini (Ed.), *Computer Vision—ECCV '92: Second European conference on computer vision, Santa Margherita Ligure, Italy* (pp. 174–183). Berlin: Springer-Verlag.

Cavanagh, P. (1989). Multiple analyses of orientation in the visual system. In *Neural mechanisms of visual perception* (pp. 261–279). The Woodlands, Texas: Portfolio Publishing Company.

Chang, T., & Kuo, C.-C. J. (1993). Texture analysis and classification with tree-structured wavelet transform. *IEEE Transactions on Image Processing*, *2*(4), 429–441.

Chubb, C., & Landy, M. S. (1991). Orthogonal distribution analysis: A new approach to the study of texture perception. In M. S. Landy & J. A. Movshon (Eds.), *Computational models of visual processing* (pp. 291–301). MIT Press.

Churchland, P. S., & Sejnowski, T. J. (1992). *The computational brain.* Cambridge, MA: MIT Press.

Clark, M., Bovik, A. C., & Geisler, W. S. (1987). Texture segmentation using Gabor modulation/demodulation. *Pattern Recognition Letters*, *6*, 261–267.

Coggins, J. M., & Jain, A. K. (1985). A spatial filtering approach to texture analysis. *Pattern Recognition*, *3*, 195–203.

Cohen, A., & Ivry, R. B. (1991). Density effects in conjunction search: Evidence for a coarse location mechanism of feature integration. *Journal of Experimental Psychology: Human Perception and Performance*, *17*(4), 891–901.

Crick, F., & Koch, C. (1992). The problem of consciousness. *Scientific American*, *267*(3), 152–159.

Cumming, B. G., Johnston, E. B., & Parker, A. J. (1993). Effects of different texture cues on curved surfaces viewed stereoscopically. *Vision Research*, *33*(5/6), 827–838.

Cutting, J. E. (1984). Reflections on surfaces: A cross-disciplinary reply to Stevens. *Journal of Experimental Psychology: General*, *113*(2), 221–224.

Cutting, J. E., & Millard, R. T. (1984). Three gradients and the perception of flat and curved surfaces. *Journal of Experimental Psychology: General*, *113*(2), 198–216.

D'Astous, F., & Jernigan, M. E. (1984). Texture discrimination based on detailed measures of the power spectrum. In *Seventh international conference on pattern recognition* (pp. 83–86). IEEE Computer Society Press.

Daugman, J. G. (1985). Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Journal of the Optical Society of America A*, *2*(7), 1160–1169.

Davis, L. S., Janos, L., & Dunn, S. M. (1983). Efficient recovery of shape from texture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *5*(5), 485–492.

De Valois, R. L., Albrecht, D. G., & Thorell, L. G. (1982). Spatial frequency selectivity of cells in macaque visual cortex. *Vision Research*, *22*, 545–559.

De Valois, R. L., & De Valois, K. K. (1990). *Spatial vision.* New York: Oxford University Press. (Originally published in 1988.)

De Valois, R. L., Thorell, L. G., & Albrecht, D. G. (1985). Periodicity of striate-cortex-cell receptive fields. *Journal of the Optical Society of America A*, *2*(7), 1115–1123.

DeAngelis, G. C., Ohzawa, I., & Freeman, R. D. (1993). Spatiotemporal organization of simple-cell receptive fields in the cat's striate cortex. II. Linearity of temporal and spatial summation. *Journal of Neurophysiology*, *69*(4), 1118–1135.

Donnelly, N., Humphreys, G. W., & Riddoch, M. J. (1991). Parallel computation of primitive shape descriptions. *Journal of Experimental Psychology: Human Perception and Performance*, *17*(2), 561–570.

Dunn, D., Higgins, W. E., & Wakeley, J. (1994). Texture segmentation using 2-D Gabor elementary functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *16*(2), 130–149.

Dyer, C. R., & Rosenfeld, A. (1976). Fourier texture features: Suppresion of aperture effects. *IEEE Transactions on Systems, Man, and Cybernetics*, *6*(20), 703–705.

Eckhorn, R., Dicke, P., Kruse, W., & Reitboeck, H. J. (1991). Stimulus-related facilitation and synchronization among visual cortical areas: Experiments and models. In H. G. Schuster (Ed.), *Nonlinear dynamics and neuronal networks* (pp. 57–75). Weinheim, F.R.G.: VCH Verlagsgesellschaft mbH. (Proceedings of the 63rd W. E. Hereaus Seminar, held at Friedrichsdorf, F.R.G., May 1990.)

Emerson, R. C. (1997). Quadrature subunits in directionally selective simple cells: Spatiotemporal interactions. *Visual Neuroscience*, *14*, 357–371.

Emerson, R. C., Bergen, J. R., & Adelson, E. H. (1992). Directionally selective complex cells and the computation of motion energy in cat visual cortex. *Vision Research*, *32*(2), 203–218.

Emerson, R. C., & Huang, M. C. (1997). Quadrature subunits in directionally selective simple cells: Counterphase and drifting grating responses. *Visual Neuroscience*, *14*, 373–385.

Enns, J. T., & Rensink, R. A. (1990). Sensitivity to three-dimensional orientation in visual search. *Psychological Science*, *1*(5), 323–326.

Enns, J. T., & Rensink, R. A. (1991). Preattentive recovery of three-dimensional orientation from line drawings. *Psychological Review*, *98*(3), 335–351.

Enns, J. T., & Rensink, R. A. (1992). A model for the rapid interpretation of line drawings in early vision. In D. Brogan, A. Gale, & K. Carr (Eds.), *Visual Search II* (pp. 73–89). London: Taylor and Francis.

Farrokhnia, F. (1990). *Multi-channel filtering techniques for texture segmentation and surface quality inspection.* Doctoral dissertation, Department of Electrical Engineering, Michigan State University, East Lansing, Michigan.

Findlay, J. M. (1980). The visual stimulus for saccadic eye movements in human observers. *Perception*, *9*, 7–21.

Findlay, J. M., Brogan, D., & Wenban-Smith, M. G. (1993). The spatial signal for saccadic eye movements emphasizes visual boundaries. *Perception & Psychophysics*, *53*(6), 633–641.

Fogel, I., & Sagi, D. (1989). Gabor filters as texture discriminator. *Biological Cybernetics*, *61*, 103–113.

Foltz, G. S., Poltrock, S. E., & Potts, G. R. (1984). Mental comparison of size and magnitude: Size congruity effects. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *10*(3), 442–453.

Foster, K. H., Gaska, J. P., Nagler, M., & Pollen, D. A. (1985). Spatial and temporal frequency selectivity of neurones in visual cortical areas V1 and V2 of the macaque monkey. *Journal of Physiology*, *365*, 331–363.

Fox, J., & Mayhew, J. E. W. (1979). Texture discrimination and the analysis of contour. *Perception*, *8*, 75–91.

Freeman, W. T., & Adelson, E. H. (1991). The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*(9), 891–906.

Gallant, J. L., Van Essen, D. C., & Nothdurft, H. C. (1995). Two-dimensional and three-dimensional texture processing in visual cortex of the macaque monkey. In T. V. Papathomas, C. Chubb, A. Gorea, & E. Kowler (Eds.), *Early vision and beyond* (pp. 89–98). Cambridge, MA: MIT Press.

Gårding, J. (1992a). Shape from texture for smooth curved surfaces. In G. Sandini (Ed.), *Computer Vision—ECCV '92: Second European conference on computer vision, Santa Margherita Ligure, Italy* (pp. 630–638). Berlin: Springer-Verlag.

Gårding, J. (1992b). Shape from texture for smooth curved surfaces in perspective projection. *Journal of Mathematical Imaging and Vision*, *2*, 329–352.

Gårding, J. (1995). Surface orientation and curvature from differential texture distortion. In *Proceedings of the fifth international conference on computer vision.*

Gaska, J. P., Jacobson, L. D., Chen, H.-W., & Pollen, D. A. (1994). Space-time spectra of complex cell filters in the macaque monkey: A comparison of results obtained with pseudowhite noise and grating stimuli. *Visual Neuroscience*, *11*, 805–821.

Geiger, D., & Girosi, F. (1991). Parallel and deterministic algorithms from MRF's: Surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*(5), 401–412.

Geiger, D., & Yuille, A. (1991). A common framework for image segmentation. *International Journal of Computer Vision*, *6*(3), 227–243.

Geman, D., Geman, S., Graffigne, C., & Don, P. (1990). Boundary detection by constrained optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(7), 609–628.

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

Gibson, J. J. (1950a). The perception of visual surfaces. *The American Journal of Psychology*, *58*(3), 367–384.

Gibson, J. J. (1950b). *The perception of the visual world.* Boston, MA: Houghton Mifflin Company.

Gibson, J. J., & Cornsweet, J. (1952). The perceived slant of visual surfaces—optical and geographical. *Journal of Experimental Psychology*, *44*, 11–15.

Gibson, J. J., & Dibble, F. K. (1952). Exploratory experiments on the stimulus conditions for the perception of a visual surface. *Journal of Experimental Psychology*, *43*(6), 414–419.

Gilbert, C. D. (1993). Circuitry, architecture, and functional dynamics of visual cortex. *Cerebral Cortex*, *3*, 373–386.

Gilbert, C. D. (1994). Circuitry, architecture and functional dynamics of visual cortex. In G. R. Bock & J. A. Goode (Eds.), *Higher-order processing in the visual system (Ciba Foundation symposium 184)* (pp. 35–56). Chichester, UK: John Wiley & Sons.

Gilbert, C. D., Hirsch, J. A., & Wiesel, T. N. (1990). Lateral interactions in visual cortex. *Cold Spring Harbor Symposia on Quantitative Biology*, *LV*, 663–677.

Gilbert, C. D., & Wiesel, T. N. (1989). Columnar specificity of intrinsic horizontal and corticocortical connections in cat visual cortex. *Journal of Neuroscience*, *9*(7), 2432–2442.

Gilbert, C. D., & Wiesel, T. N. (1990). The influence of contextual stimuli on the orientation selectivity of cells in primary visual cortex of the cat. *Vision Research*, *30*(11), 1689–1701.

Gilbert, C. D., & Wiesel, T. N. (1992). Receptive field dynamics in adult primary visual cortex. *Nature*, *356*, 150–152.

Glezer, V. D., Ivanoff, V. A., & Tscherbach, T. A. (1973). Investigation of complex and hypercomplex receptive fields of visual cortex of the cat as spatial frequency filters. *Vision Research*, *13*, 1875–1904.

Graham, N. (1991). Complex channels, early local nonlinearities, and normalization in texture segregation. In M. S. Landy & J. A. Movshon (Eds.), *Computational models of visual processing* (pp. 273–290). Cambridge, MA: MIT Press.

Graham, N. (1992). Breaking the visual stimulus into parts. *Current Directions in Psychological Science*, *1*(2), 55–61.

Graham, N., Beck, J., & Sutter, A. (1992). Nonlinear processes in spatial-frequency channel models of perceived texture segregation: Effects of sign and amount of contrast. *Vision Research*, *32*(4), 719–743.

Graham, N., & Sutter, A. (1996). Effect of spatial scale and background luminance on the intensive and spatial nonlinearities in texture segregation. *Vision Research*, *36*(10), 1371–1390.

Graham, N., Sutter, A., & Venkatesan, C. (1993). Spatial-frequency- and orientation-selectivity of simple and complex channels in region segregation. *Vision Research*, *33*(14), 1893–1911.

Graham, N., Sutter, A., Venkatesan, C., & Humaran, M. (1992). Nonlinear processes in perceived region segregation: Orientation selectivity of complex channels. *Ophthalmic and Physiological Optics*, *12*, 142–146.

Graham, N. V. S. (1989). *Visual pattern analyzers.* New York: Oxford University Press.

Grannan, E. R., Kleinfeld, D., & Sompolinsky, H. (1993). Stimulus-dependent synchronization of neuronal assemblies. *Neural Computation*, *5*, 550–569.

Gurnsey, R., & Browse, R. A. (1989). Asymmetries in visual texture discrimination. *Spatial Vision*, *4*(1), 31–44.

Gurnsey, R., & Laundry, D. S. (1992). Texture discrimination with and without abrupt texture gradients. *Canadian Journal of Psychology*, *46*(2), 306–332.

Haralick, R. M., Sternberg, S. R., & Zhuang, X. (1987). Image analysis using mathematical morphology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *9*(4), 532–550.

Harris, J. G., Koch, C., & Luo, J. (1989). Resistive fuses: Analog hardware for detecting discontinuities in early vision. In *Analog VLSI implementation of neural systems* (pp. 27–55). Kluwer Academic Publishers.

Harris, J. G., Koch, C., & Luo, J. (1990). A two-dimensional analog VLSI circuit for detecting discontinuities in early vision. *Science*, *248*, 1209–1211.

Harris, J. G., Koch, C., Staats, E., & Luo, J. (1990). Analog hardware for detecting discontinuities in early vision. *International Journal of Computer Vision*, *4*, 211–223.

Harvey, L. O., & Gervais, M. J. (1978). Visual texture perception and Fourier analysis. *Perception & Psychophysics*, *24*(6), 534–542.

Harvey, L. O., & Gervais, M. J. (1981). Internal representation of visual texture as the basis for the judgment of similarity. *Journal of Experimental Psychology: Human Perception and Performance*, *7*(4), 741–753.

Hawken, M. J., & Parker, A. J. (1987). Spatial properties of neurons in the monkey striate cortex. *Proceedings of the Royal Society of London B*, *231*, 251–288.

Hawken, M. J., & Parker, A. J. (1991). Spatial receptive field organization in monkey V1 and its relationship to the cone mosaic. In M. S. Landy & J. A. Movshon (Eds.), *Computational models of visual processing* (pp. 83–93). Cambridge, MA: MIT Press.

He, D.-C., & Wang, L. (1991). Textural filters based on the texture spectrum. *Pattern Recognition*, *24*(12), 1187–1195.

He, Z. J., & Nakayama, K. (1992). Surfaces versus features in visual search. *Nature*, *359*, 231–233.

He, Z. J., & Nakayama, K. (1994a). Apparent motion determined by surface layout, not disparity or 3-dimensional distance. *Nature*, *367*, 173–175.

He, Z. J., & Nakayama, K. (1994b). Perceiving textures: Beyond filtering. *Vision Research*, *34*(2), 151–162.

Heathcote, A., & Mewhort, D. J. K. (1993). Representation and selection of relative position. *Journal of Experimental Psychology: Human Perception and Performance*, *19*(3), 488–516.

Hebb, D. O. (1980). *Essay on mind.* Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Heeger, D. J. (1987). Model for the extraction of image flow. *Journal of the Optical Society of America A*, *4*(8), 1455–1471.

Heeger, D. J. (1990). Nonlinear model of cat striate physiology. *Society for Neuroscience Abstracts*, *16*, 229.

Heeger, D. J. (1991). Nonlinear model of neural responses in cat visual cortex. In M. S. Landy & J. A. Movshon (Eds.), *Computational models of visual processing* (p. 119-133). Cambridge, MA: MIT Press.

Heeger, D. J. (1992a). Half-squaring in responses of cat striate cells. *Visual Neuroscience*, *9*, 427–443.

Heeger, D. J. (1992b). Normalization of cell responses in cat striate cortex. *Visual Neuroscience*, *9*, 181–197.

Heeger, D. J. (1993). Modeling simple-cell direction selectivity with normalized, half-squared, linear operators. *Journal of Neurophysiology*, *70*(5), 1885–1898.

Heeger, D. J., & Adelson, E. H. (1989). Nonlinear model of cat striate cortex. *Optics News*, *15*, A-42.

Heitger, F., Rosenthaler, L., von der Heydt, R., Peterhans, E., & Kübler, O. (1992). Simulation of neural contour mechanisms: From simple to end-stopped cells. *Vision Research*, *32*(5), 963–981.

Hepp, K., van Opstal, A. J., Straumann, D., Hess, B. J., & Henn, V. (1993). Monkey superior colliculus represents rapid eye movements in a two-dimensional motor map. *Journal of Neurophysiology*, *63*(3), 965–979.

Hess, R. F., Dakin, S. R. G., & Badcock, D. (1994). Localization of element clusters by the human visual system. *Vision Research*, *34*(18), 2439–2451.

Hess, R. F., & Holliday, I. E. (1992). The coding of spatial position by the human visual system: Effects of spatial scale and contrast. *Vision Research*, *32*(6), 1085–1097.

Hine, T., Cook, M., & Rogers, G. T. (1997). The Ouchi illusion: An anomaly in the perception of rigid motion for limited spatial frequencies and angles. *Perception & Psychophysics*, *59*(3), 448–455.

Hirsch, J., & Mjolsness, E. (1992). A center-of-mass computation describes the precision of random dot displacement discrimination. *Vision Research*, *32*(2), 335–346.

Hirsch, J. A., & Gilbert, C. D. (1991). Synaptic physiology of horizontal connections in the cat's visual cortex. *Journal of Neuroscience*, *11*(6), 1800–1809.

Hofmann, T., Puzicha, J., & Buhmann, J. M. (1997). Deterministic annealing for unsupervised texture segmentation. In M. Pelillo & E. R. Hancock (Eds.), *Energy minimization methods in computer vision and pattern recognition (EMMCVPR '97)* (pp. 212–228). Springer-Verlag.

Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, *81*, 3088-3092.

Hopfield, J. J., & Tank, D. W. (1985). Neural computation of decisions in optimization problems. *Biological Cybernetics*, *52*, 141-152.

Horn, B. (1986). *Robot vision.* Cambridge, MA: MIT Press.

Hsu, T.-I., Calway, A. D., & Wilson, R. (1992, June). *Analysis of structured texture using the multiresolution Fourier transform.* (Unpublished manuscript, University of Warwik, U.K.)

Hubel, D. H. (1988). *Eye, brain, and vision.* New York: Scientific American Library [Distributed by W. H. Freeman and Company].

Hubel, D. H., & Wiesel, T. N. (1959). Receptive fields of single neurons in the cat's striate cortex. *Journal of Physiology*, *148*, 574–591.

Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *Journal of Physiology*, *160*, 106–154.

Hubel, D. H., & Wiesel, T. N. (1968). Receptive fields and functional architecture of monkey striate cortex. *Journal of Physiology*, *195*(1), 215–243.

Hughes, H. C. (1982). Search for the neural mechanisms essential to basic figural synthesis in the cat. In D. Ingle, M. A. Goodale, & R. J. W. Mansfield (Eds.), *Analysis of visual behavior* (pp. 771–800). MIT Press.

Ikeuchi, K. (1984). Shape from regular patterns. *Artificial Intelligence*, *22*(1), 49–75.

Jacobson, L. D., Gaska, J. P., Chen, H.-W., & Pollen, D. A. (1993). Structural testing of multi-input linear–nonlinear cascade models for cells in macaque striate cortex. *Vision Research*, *33*(5/6), 609–626.

Jain, A. K., & Farrokhnia, F. (1991). Unsupervised texture segmentation using Gabor filters. *Pattern Recognition*, *24*(12), 1167–1186.

Jain, A. K., & Karu, K. (1995). Texture analysis: Representation and matching. In C. Braccini (Ed.), *Image analysis and processing: Proceedings of the 8th international conference (ICIAP '95)* (pp. 3–10). Berlin: Springer-Verlag.

Jain, R., Kasturi, R., & Schunk, B. G. (1995). *Machine vision.* New York: McGraw-Hill.

Jau, Y. C., & Chin, R. T. (1988). Shape from texture using the Wigner distribution. In *Proceedings of the conference on computer vision and pattern recognition* (pp. 515–523). IEEE Computer Society Press.

Johnston, E. B., Cumming, B. G., & Parker, A. J. (1993). Integration of depth modules: Stereopsis and texture. *Vision Research*, *33*(5/6), 813–826.

Jones, D. G., & Malik, J. (1992). Determining three-dimensional shape from orientation and spatial frequency disparities. In G. Sandini (Ed.), *Computer Vision—ECCV '92: Second European conference on computer vision, Santa Margherita Ligure, Italy* (pp. 661–669). Berlin: Springer-Verlag.

Jones, J. P., & Palmer, L. A. (1987a). An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, *58*(6), 1233–1258.

Jones, J. P., & Palmer, L. A. (1987b). The two-dimensional spatial structure of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, *58*(6), 1287–1211.

Jones, J. P., & Palmer, L. A. (1987c). The two-dimensional spectral structure of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, *58*(6), 1212–1232.

Julesz, B. (1962). Visual pattern discrimination. *IRE Transactions on Information Theory*, *IT-8*, 84–92.

Julesz, B. (1965). Texture and visual perception. *Scientific American*, *212*, 38–48.

Julesz, B. (1984). Toward an axiomatic theory of preattentive vision. In G. M. Edelman, W. E. Gall, & W. M. Cowan (Eds.), *Dynamic aspects of neocortical function* (pp. 585–612). New York: John Wiley & Sons.

Julesz, B. (1986). Texton gradients: The texton theory revisited. *Biological Cybernetics*, *54*, 254–251.

Julesz, B., & Bergen, J. R. (1983). Textons: The fundamental elements in preattentive vision and perception of textures. *Bell System Technical Journal*, *62*(6), 1619–1645.

Julesz, B., & Kröse, B. (1988). Features and spatial filters. *Nature*, *333*, 302–303.

Kagan, J., Linn, S., Mount, R., Reznick, J. S., & Hiatt, S. (1979). Asymmetry of interference in the dishabituation paradigm. *Canadian Journal of Psychology*, *33*(4), 288–305.

Kanatani, K. (1984). Detection of surface orientation and motion from texture by a stereological technique. *Artificial Intelligence*, *23*, 213–237.

Kandel, E. R., Schwartz, J. H., & Jessell, T. M. (1991). *Principles of neural science* (3rd ed.). New York: Elsevier Science Publishing Co., Inc.

Kanerva, P. (1988). *Sparse distributed memory.* Cambridge, MA: MIT Press.

Kanizsa, G. (1976). Subjective contours. *Scientific American.* (Reprinted in *The Mind's Eye*, ed. Jeremy M. Wolfe, New York: W. H. Freeman and Company, 1986.)

Kaplan, R., & Kaplan, S. (1989). *The experience of nature: A psychological perspective.* Cambridge, MA: Cambridge University Press.

Kaplan, S., & Kaplan, R. (1982). *Cognition and environment: Functioning in an uncertain world.* New York: Praeger. (Republished by Ulrich's, Ann Arbor, Michigan, 1989.)

Kaplan, S., Sonntag, M., & Chown, E. (1991). Tracing recurrent activity in cognitive elements (TRACE): A model of temporal dynamics in a cell assembly. *Connection Science*, *3*(2).

Kender, J. R., & Kanade, T. (1980). Mapping image properties into shape constraints: Skewed symmetry, affine-transformable patterns, and the shape-from-texture paradigm. In *Proceedings of the national conference on artificial intelligence* (pp. 4–6).

Kernighan, B. W., & Ritchie, D. M. (1988). *The C programming language* (2nd ed.). Englewood Cliffs, New Jersey: Prentice-Hall.

Khotanzad, A., & Chen, J.-Y. (1989). Unsupervised segmentation of textured images by edge detection in multidimensional spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *11*(4), 414–421.

Kingdom, F. A. A., & Keeble, D. R. T. (1996). A linear systems approach to the detetion of both abrupt and smooth spatial variations in orientation-defined textures. *Vision Research*, *36*(3), 409–420.

Kirkpatrick, S., Jr., C. D. G., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*, 671–680.

Kleffner, D. A., & Ramachandran, V. S. (1992). On the perception of shape from shading. *Perception & Psychophysics*, *52*(1), 18–36.

Knierim, J. J., & Van Essen, D. C. (1992). Neuronal responses to static texture patterns in area V1 of the alert macaque monkey. *Journal of Neurophysiology*, *67*(4), 961–980.

Koch, C. (1987). The action of the corticofugal pathway on sensory thalamic nuclei: A hypothesis. *Neuroscience*, *23*(2), 399–406.

Koch, C., Marroquin, J., & Yuille, A. (1986). Analog "neuronal" networks in early vision. *Proceedings of the National Academy of Sciences, U.S.A.*, *83*, 4263–4267.

Koch, C., & Poggio, T. (1987). Biophysics of computation: Neurons, synapses, and membranes. In G. M. Edelman, W. E. Gall, & W. M. Cowan (Eds.), *Synaptic function.* New York: Wiley.

Koenderink, J.-J., Kappers, A., & van Doorn, A. (1992). Local operations: The embodiment of geometry. In G. A. Orban & H.-H. Nagels (Eds.), *Artificial and biological vision systems* (pp. 1–23). Berlin: Springer-Verlag.

Koenderink, J. J., & van Doorn, A. J. (1990). Receptive field families. *Biological Cybernetics*, *63*, 291–297.

Koenderink, J. J., & van Doorn, A. J. (1992). Generic neighborhood operators. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *14*(6), 597–605.

Koenig, R., Dunn, H. K., & Lacey, L. Y. (1946). The sound spectrograph. *Journal of the Acoustical Society of America*, *18*, 19–49.

Kosslyn, S. M., Flynn, R. A., Amsterdam, J. B., & Wang, G. (1990). Components of high-level vision: A cognitive neuroscience analysis and accounts of neurological syndromes. *Cognition*, *34*, 203–277.

Kowler, E., Anderson, E., Dosher, B., & Blaser, E. (1995). The role of attention in the programming of saccades. *Vision Research*, *35*(13), 1897–1916.

Kruizinga, P., & Petkov, N. (1995). Person identification based on multiscale matching of cortical images. In B. Hertzberger & G. Serazzi (Eds.), *Proceedings of the international conference and exhibition on high-performance computing and networking, HPCN Europe '95* (p. 420-427). (Volume 919 of *Lecture Notes in Computer Science*, Berlin: Springer-Verlag, 1995.)

Krumm, J. (1993). *Space frequency shape inference and segmentation of 3D surfaces.* Unpublished doctoral dissertation, Robotics Institute, Carnegie Mellon University, Carnegie Mellon University, Pittsburgh, PA. (Also available as Technical Report CMU-RI-TR-93-32.)

Krumm, J., & Shafer, S. A. (1994). Segmenting textured 3D surfaces using the space/frequency representation. *Spatial Vision*, *8*(2), 281–308.

Krumm, J., & Shafer, S. A. (1995). Texture segmentation and shape in the same image. In *Proceedings of the fifth international conference on computer vision* (pp. 121–127). Washington, D.C.: IEEE Computer Society Press.

Kubota, T., & Huntsberger, T. (1997). Adaptive anisotropic parameter estimation in the weak membrane model. In M. Pelillo & E. R. Hancock (Eds.), *Energy minimization methods in computer vision and pattern recognition (EMMCVPR '97)* (pp. 179–194). Springer-Verlag.

Kulikowski, J. J., & Bishop, P. O. (1981). Linear analysis of the responses of simple cells in the cat visual cortex. *Experimental Brain Research*, *44*, 386–400.

Kulikowski, J. J., Marčelja, S., & Bishop, P. O. (1982). Theory of spatial position and spatial frequency relations in the receptive fields of simple cells in the visual cortex. *Biological Cybernetics*, *43*, 187–198.

Landy, M. S., & Bergen, J. R. (1991). Texture segregation and orientation gradient. *Vision Research*, *31*(4), 679–691.

Landy, M. S., & Movshon, J. A. (Eds.). (1991). *Computational models of visual processing.* Cambridge, MA: MIT Press.

Laws, K. I. (1980). *Textured image segmentation* (USCIPI Report No. 940). Department of Electricial Engineering-Systems, University of Southern California, Los Angeles: Signal and Image Processing Institute. (Ph.D. dissertation published as a technical report.)

Lee, T. S. (1993). *Surface inference by minimizing energy functionals: A computational framework for the visual cortex.* Unpublished doctoral dissertation, Division of Applied Sciences, Harvard University, Cambridge, MA.

Lee, T. S. (1995). A Bayesian framework for understanding texture segmentation in the primary visual cortex. *Vision Research*, *35*(18), 2643–2657. (The published journal version contains several printing errors. The article, without the errors, is also available via FTP: `ftp://ftp.hrl.harvard.edu/pub/papers/VRtai.ps.Z`.)

Lee, T. S. (1996). Image representation using 2D Gabor wavelets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(10), 959–971.

Lee, T. S., Mumford, D., & Yuille, A. (1991). *Texture segmentation by minimizing vector-valued energy functionals: The coupled-membrane model* (Technical Report No. 91-22). Harvard University, Boston, Massachusetts: Harvard Robotics Laboratory.

Lee, T. S., Mumford, D., & Yuille, A. (1992). Texture segmentation by minimizing vector-valued energy functionals: The coupled-membrane model. In G. Sandini (Ed.), *Computer Vision—ECCV '92: Second European conference on computer vision, Santa Margherita Ligure, Italy* (pp. 165–173). Berlin: Springer-Verlag.

Lendaris, G., & Stanley, G. (1970). Diffraction pattern sampling for automatic pattern recognition. In *Proceedings of the IEEE* (Vol. 58, pp. 198–216).

Lesperance, R. M. (1990). *The location system: Using approximate location and size information for scene segmentation.* Unpublished doctoral dissertation, University of Michigan, Ann Arbor, MI.

Lifshitz, L. M., & Pizer, S. M. (1990). A multiresolution hierarchical approach to image segmentation based on intensity extrema. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(6), 529–540.

Lindeberg, T. (1994). Scale-space theory: A basic tool for analysing structures at different scales. *Journal of Applied Statistics*, *21*(2), 225–270.

Lindeberg, T., & ter Haar Romeny, B. M. (1994). Linear scale-space. In B. M. ter Haar Romeny (Ed.), *Geometry-driven diffusion in computer vision* (pp. 1–77). Dordrecht, Netherlands: Kluwer Academic Publishers. (In press.)

Linn, S., Hans, S., & Kagan, J. (1978). Successive visual discrimination of forms in 10-month-old infants. *The Journal of Genetic Psychology*, *133*, 71–78.

Linn, S., Reznick, J. S., Kagan, J., & Hans, S. (1982). Salience of visual patterns in the human infant. *Development Psychology*, *18*(5), 651–657.

Liu, Z., Gaska, J. P., Jacobson, L. D., & Pollen, D. A. (1992). Interneuronal interaction between members of quadrature phase and anti-phase pairs in the cat's visual cortex. *Vision Research*, *32*(7), 1193–1198.

Lumsdaine, A., J. L. Wyatt, J., & Elfadel, I. M. (1991). *Nonlinear analog networks for image smoothing and segmentation* (A.I. Memo No. AIM-1280). Cambridge, MA: M.I.T. Artificial Intelligence Laboratory.

Malik, J., & Perona, P. (1990). Preattentive texture discrimination with early vision mechanisms. *Journal of the Optical Society of America A*, *7*(5), 923–932.

Malik, J., & Rosenholtz, R. (1993). *Computing local surface orientation and shape from texture for curved surfaces* (Technical Report No. CSD-93-775). Berkeley, California: Dept. of Electrical Engineering and Computer Science, University of California at Berkeley.

Malik, J., & Rosenholtz, R. (1994a). A computational model for shape from texture. In G. R. Bock & J. A. Goode (Eds.), *Higher-order processing in the visual system (Ciba Foundation symposium 184)* (pp. 272–283). Chichester, UK: John Wiley & Sons.

Malik, J., & Rosenholtz, R. (1994b). Recovering surface curvature and orientation from texture distortion: A least squares algorithm and sensitivity analysis. In J.-O. Eklundh (Ed.), *Computer Vision—ECCV '94: Third European conference on computer vision, Stockholm, Sweden* (Vol. 1, pp. 393–364). Berlin: Springer-Verlag. (Also available as *Lecture Notes in Computer Science, vol. 800*, Jan-Olof Eklundh (Ed.), Springer-Verlag, 1994.)

Malik, J., & Rosenholtz, R. (1997). Computing local surface orientation and shape from texture for curved surfaces. *International Journal of Computer Vision*, *23*(2), 149–168.

Marr, D. (1976). Analyzing natural images: A computational theory of texture vision. *Cold Spring Harbor Symposia on Quantitative Biology*, *XL*, 647–662.

Marr, D. (1982). *Vision.* San Francisco, CA: W.H. Freeman.

Marr, D., & Nishihara, H. K. (1978). Representation and recognition of spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London B*, *200*, 269–294.

Marrocco, R. T., & Li, R. H. (1977). Monkey superior colliculus: Properties of single cells and their afferent inputs. *Journal of Neurophysiology*, *40*(4), 844–860.

Marroquin, J. L. (1984). *Surface reconstruction preserving discontinuities* (A.I. Memo No. AIM-792). Cambridge, MA: M.I.T. Artificial Intelligence Laboratory.

Marroquin, J. L. (1985a). *Optimal Bayesian estimators for image segmentation and surface reconstruction* (A.I. Memo No. AIM-839). M.I.T. Artificial Intelligence Laboratory.

Marroquin, J. L. (1985b). *Probabilistic solution of inverse problems* (A.I. Technical Report No. AITR-860). Cambridge, MA: M.I.T. Artificial Intelligence Laboratory. (The author's Ph.D. dissertation.)

Marčelja, S. (1980). Mathematical description of the responses of simple cortical cells. *Journal of the Optical Society of America*, *70*(11), 1297–1300.

MathWorks, T. (1998a). *Using* MATLAB. Natik, MA: The MathWorks, Inc.

MathWorks, T. (1998b). MATLAB *optimization toolbox user's guide.* Natik, MA: The MathWorks, Inc.

Mattingley, J. B., Davis, G., & Driver, J. (1997). Preattentive filling-in of visual surfaces in parietal extinction. *Science*, *275*, 671–674.

Menz, C., & Groner, R. (1987). Saccadic programming with multiple targets under different task conditions. In J. K. O'Reagan & A. Lévy-Schoen (Eds.), *Eye movements: From physiology to cognition* (pp. 95–103). Elsevier Science Publishers B.V. (North-Holland).

Mesrobian, E., & Skrzypek, J. (1995). Segmenting textures using cells with adaptive receptive fields. *Springer-Verlag*, *9*(2), 163–190.

Milner, P. M. (1974). A model for visual shape recognition. *Psychological Review*, *81*(6), 521–535.

Mishkin, M., Ungerleider, L. G., & Macko, K. A. (1983). Object vision and spatial vision: Two cortical pathways. *Trends in Neurosciences*, 414–417.

Moerdler, M. L. (1989). *Shape from textures: A paradigm for fusing middle level vision cues.* Unpublished doctoral dissertation, Graduate School of Arts and Sciences, Columbia University.

Moerdler, M. L., & Kender, J. R. (1987). An integrated system that unifies multiple shape from texture algorithms. In *Proceedings of the sixth national conference on artificial intelligence* (pp. 723–726). Morgan Kaufmann Publishers.

Morel, J.-M., & Solimini, S. (1995). *Variational methods for image segmentation.* Boston, MA: Birkhäuser.

Morgan, M. J., Hole, G. J., & Glennerster, A. (1990). Biases and sensitivities in geometrical illusions. *Vision Research*, *30*(11), 1793–1810.

Movshon, J. A., Thompson, I. D., & Tolhurst, D. J. (1978). Spatial and temporal contrast sensitivity of neurones in areas 17 and 18 of the cat's visual cortex. *Journal of Physiology*, *283*, 101–120.

Mumford, D., & Shah, J. (1985). Boundary detection by minimizing functionals, I. In *Proceedings of the conference on computer vision and pattern recognition* (pp. 22–26).

Nakayama, K., & He, Z. J. (1991). Attention to surfaces: Beyond a Cartesian understanding of focal attention. In M. S. Landy & J. A. Movshon (Eds.), *Computational models of visual processing* (pp. 181–188). Cambridge, MA: MIT Press.

Nakayama, K., & Shimojo, S. (1992). Experiencing and perceiving visual surfaces. *Science*, *257*(5075), 1357–1363.

Nakayama, K., & Shimojo, S. (1996). Experiencing and perceiving visual surfaces. *Science*, *257*, 1357–1363.

Nalwa, V. S. (1993). *A guided tour of computer vision.* Reading, Massachusetts: Addison-Wesley Publishing Company.

Nelson, R. C. (1988). *Visual navigation.* Ph.d. thesis, University of Maryland, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, Maryland. (Available as Technical Report CS-TR-2087)

Nestares, O., & Heeger, D. J. (1997). Modeling the apparent frequency-specific suppression in simple cell responses. *Vision Research*, *37*, 1535–1543.

Nicholls, J. G., Martin, A. R., & Wallace, B. G. (1992). *From neuron to brain: A cellular and molecular approach to the function of the nervous system* (3rd ed.). Sunderland, Mass.: Sinauer Associates.

Nielsen, M. (1997). Graduated nonconvexity by functional focusing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(5), 521–525.

Nothdurft, H.-C. (1985). Sensitivity for structure gradient in texture discrimination tasks. *Vision Research*, *12*, 1957–1968.

Nothdurft, H.-C. (1990). Texton segregation by associated differences in global and local luminance distribution. *Proceedings of the Royal Society of London B*, *239*, 295–320.

Nothdurft, H.-C., & Li, C. Y. (1985). Texture discriminators: representation of orientation and luminance differences in cells of the cat striate cortex. *Vision Research*, *25*(1), 99–113.

Nowak, L. G., Munk, M. H. J., Girard, P., & Bullier, J. (1995). Visual latencies in areas V1 and V2 of the macaque monkey. *Visual Neuroscience*, *12*, 371–384.

Ohta, Y., Maenobu, K., & Sakai, T. (1981). Obtaining surface orientation from texels under perspective projection. In *Proceedings of the international joint conference on artificial intelligence* (pp. 746–751).

Ohzawa, I., DeAngelis, G. C., & Freeman, R. D. (1990). Stereoscopic depth discrimination in the visual cortex: Neurons ideally suited as disparity detectors. *Science*, *249*, 1037–1041.

Olson, R. K., & Attneave, F. (1970). What variables produce similarity grouping? *The American Journal of Psychology*, *83*, 1-21.

O'Neill, B. (1966). *Elementary differential geometry.* New York: Academic Press.

Palmer, L. A., Jones, J. P., & Stepnoski, R. A. (1991). Striate receptive fields as linear filters: Characterization in two dimensions of space. In A. G. Leventhal (Ed.), *The neural basis of visual function* (pp. 246–265). Boca Raton, Florida: CRC Press.

Parker, A. J., & Hawken, M. J. (1988). Two-dimensional spatial structure of receptive fields in monkey striate cortex. *Journal of the Optical Society of America A*, *5*(4), 598–605.

Pentland, A. P. (1986). Shading into texture. *Artificial Intelligence*, *29*, 147–170.

Perona, P. (1995). Deformable kernels for early vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *17*(5), 488–499.

Perry, V. H., & Cowey, A. (1984). Retinal ganglion cells that project to the dorsal lateral geniculate nucleus in the macaque monkey. *Neuroscience*, *12*(4), 1101–1123.

Petkov, N., & Kruizinga, P. (1997). Computational models of visual neurons specialised in the detection of periodic and aperiodic oriented visual stimuli: Bar and grating cells. *Biological Cybernetics*, *76*(2), 83–96.

Picard, R., Graczyk, C., Mann, S., Wachman, J., Picard, L., & Campbell, L. (1995, April). *Vision texture database, version 1.0 [on-line].* World Wide Web: `http://www-white.media.mit.edu/vismod/imagery/VisionTexture/`.

Pickett, R. M. (1970). Visual analyses of texture in the detection and recognition of objects. In *Picture processing and psychopictorics* (pp. 289–308). New York: Academic Press.

Polat, U., & Sagi, D. (1993). Lateral interactions between spatial channels: Suppression and facilitation revealed by lateral masking experiments. *Vision Research*, *33*(7), 993–999.

Polat, U., & Sagi, D. (1994a). The architecture of perceptual spatial interactions. *Vision Research*, *34*(1), 73–78.

Polat, U., & Sagi, D. (1994b). Spatial interactions in human vision: From near to far via experience-dependent cascades of connections. *Proceedings of the National Academy of Sciences, U.S.A.*, *91*, 1206–1209.

Pollen, D. A., & Ronner, S. F. (1981). Phase relationships between adjacent simple cells in the visual cortex. *Science*, *212*, 1409–1411.

Pollen, D. A., & Ronner, S. F. (1983). Visual cortical neurons as localized spatial frequency filters. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-13*(5), 907–916.

POV-Team™. (1997). *Persistence of Vision™ ray-tracer (*POV-Ray™*) user's documentation (version 3.00) [computer software manual].* Available through the World Wide Web: `http://www.povray.org`.

Pratt, W. K. (1991). *Digital image processing* (Second ed.). New York: John Wiley & Sons.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1992). *Numerical recipes in C: The art of scientific computation* (2nd ed.). Cambridge: Cambridge University Press.

PRIP. (1997). *PRIP image database [on-line].* Available through the World Wide Web: `http://www.prip.tuwien.ac.at/prip/image.html`.

Ramachandran, V. S. (1988). Perception of shape from shading. *Nature*, *331*(6152), 133–166.

Ramachandran, V. S. (1990). Visual perception in people and machines. In A. Blake & T. Troscianko (Eds.), *AI and the eye* (pp. 21–77). Chichester, England: John Wiley & Sons. (Proceedings of the 11th European Conference on Visual Perception, held in Bristol, U.K., in the summer of 1988.)

Randen, T. (1997). *Images used in publications [on-line].* Available through the World Wide Web: `http://www.hsr.no/tele-signal-group/www/images.html`. Signal Processing Group, Høgskolen i Stavanger, Norway.

Randen, T., & Husøy, J. H. (1993). Filter banks for texture segmentation. *Piksel'n*, *3*, 22–29.

Randen, T., & Husøy, J. H. (1994). Multichannel filtering for image texture segmentation. *Optical Engineering*, *33*(8), 2617–2625.

Rao, R. P. N., & Ballard, D. H. (1995). An active vision architecture based on iconic representations. *Artificial Intelligence*, *78*, 461–505.

Reed, T. R., & Wechsler, H. (1990). Segmentation of textured images and gestalt organization using spatial/spatial-frequency representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(1), 1–12.

Reed, T. R., Wechsler, H., & Werman, M. (1990). Texture segmentation using a diffusion region growing technique. *Pattern Recognition*, *23*(9), 953–960.

Reichel, F. D., Todd, J. T., & Yilmaz, E. (1995). Visual discrimination of local surface depth and orientation. *Perception & Psychophysics*, *57*(8), 1233–1240.

Rentschler, I., Hübner, M., & Caelli, T. M. (1988). On the discrimination of compound Gabor signals and textures. *Vision Research*, *28*(2), 279–291.

Richards, W., & Polit, A. (1974). Texture matching. *Kybernetik*, *16*, 155–162.

Robinson, D. L., & McClurkin, J. W. (1989). The visual superior colliculus and pulvinar. In R. H. Wurtz & M. E. Goldberg (Eds.), *The neurobiology of saccadic eye movements* (pp. 337–360). Elsevier Science Publishers B.V. (North-Holland).

Robinson, D. L., & Petersen, S. E. (1992). The pulvinar and visual salience. *Trends in Neurosciences*, *15*(4), 127–132.

Robson, J. G. (1966). Spatial and temporal contrast-sensitivity functions of the visual system. *Journal of the Optical Society of America*, *56*, 1141–1142.

Rodman, H. R., Gross, C. G., & Albright, T. D. (1990). Afferent basis of visual response properties in area MT of the macaque. II. Effects of superior colliculus removal. *Journal of Neuroscience*, *10*(4), 1154–1164.

Ronse, C. (1993). On idempotence and related requirements in edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *15*(5), 484–491.

Ronse, C. (1995). *The phase congruence model for edge detection in two-dimensional pictures: A mathematical study* (Report No. 95/11). Strasbourg, France: LSITT-URA, Département d'Informatique, Université Louis Pasteur. (Available through the World Wide Web: `http://dpt-info.u-strasbg.fr:8080/~ronse/`.)

Rosenfeld, A. (1962). Automatic recognition of basic terrain types from aerial photographs. *Photogrammetric Engineering*, *28*, 115–132.

Rosenfeld, A. (1975). Fuzzy graphs. In *Fuzzy sets and their applications to cognitive and decision processes* (p. 77-95). Academic Press.

Rosenholtz, R., & Malik, J. (1997). Surface orientation from texture: Isotropy or homogeneity (or both)? *Vision Research*, *37*(16), 2283–2293.

Rossi, A. F., Rittenhouse, C. D., & Paradiso, M. A. (1996). The representation of brightness in primary visual cortex. *Science*, *273*, 1104–1107.

Rubenstein, B. S., & Sagi, D. (1990). Spatial variability as a limiting factor in texture-discrimination tasks: implications for performance asymmetries. *Journal of the Optical Society of America A*, *7*(9), 1632–1643.

Sachs, M. B., Nachmias, J., & Robson, J. G. (1971). Spatial-frequency channels in human vision. *Journal of the Optical Society of America*, *61*(9), 1176–1186.

Sagi, D. (1990). Detection of an orientation singularity in Gabor textures: Effect of signal density and spatial-frequency. *Vision Research*, *30*(9), 1377–1388.

Sagi, D. (1991). Spatial filters in texture segmentation tasks. In B. Blum (Ed.), *Channels in the visual nervous system: Neurophysiology, psychophysics and models* (pp. 397–424). London: Freund Publishing House, Ltd.

Sagi, D. (1995). The psychophysics of texture segmentation. In T. V. Papathomas, C. Chubb, A. Gorea, & E. Kowler (Eds.), *Early vision and beyond* (p. 69-79). Cambridge, MA: MIT Press.

Sagi, D., & Hochstein, S. (1985). Lateral inhibition between spatially adjacent spatial-frequency channels? *Perception & Psychophysics*, *37*(4), 315–322.

Sakai, K., & Finkel, L. H. (1994). A shape-from-texture algorithm based on human visual psychophysics. In *Proceedings of the conference on computer vision and pattern recognition* (pp. 527–532).

Sakai, K., & Finkel, L. H. (1995). Characterization of the spatial-frequency spectrum in the perception of shape from texture. *Journal of the Optical Society of America A*, *12*(6), 1208–1224.

Sakai, K., & Finkel, L. H. (1997). Spatial-frequency analysis in the perception of perspective depth. *Network: Computation in Neural Systems*, *8*(3), 335–352.

Sakai, K., Yen, S.-C., & Finkel, L. H. (1996). Enhancement of shape-from-texture perception by spatial frequency processing. *Investigative Ophthalmology and Visual Science*, *37*, 810.

Schiller, P. H., Finlay, B. L., & Volman, S. F. (1976). Quantitative studies of single-cell properties in monkey striate cortex. III. Spatial frequency. *Journal of Neurophysiology*, *39*(6), 1334–1351.

Schnörr, C., & Sprengel, R. (1994). A nonlinear regularization approach to early vision. *Biological Cybernetics*, *72*, 141–149.

Schyns, P. G., & Oliva, A. (1994). From blobs to boundary edges: Evidence for time- and spatial-scale-dependent scene recognition. *Psychological Science*, *5*(4), 195–200.

Sedgwick, H. A. (1983). Environment-centered representation of spatial layout: Available visual information from texture and perspective. In J. Beck, B. Hope, & A. Rosenfeld (Eds.), *Human and machine vision* (pp. 425–458). New York: Academic Press.

Shah, J. (1990). Parameter estimation, multiscale representation and algorithms for energy-minimizing segmentations. In *International conference on pattern recognition* (pp. 815–819). Los Alamitos, California: IEEE Computer Society Press. (Catalog Number 90CH2898-5.)

Shah, J. (1991). Segmentation by nonlinear diffusion. In *Proceedings of the conference on computer vision and pattern recognition* (pp. 202–207). Los Alamitos, California: IEEE Computer Society Press. (Catalog Number 91CH2983-5.)

Shah, J. (1992a). Properties of energy-minimizing segmentations. *SIAM Journal of Control and Optimization*, *30*(1), 99–111.

Shah, J. (1992b). Segmentation by nonlinear diffusion, II. In *Proceedings of the conference on computer vision and pattern recognition* (pp. 644–647). Los Alamitos, California: IEEE Computer Society Press. (Catalog Number 92CH3168-2.)

Shah, J. (1996a). A common framework for curve evolution, segmentation and anisotropic diffusion. In *Proceedings of the conference on computer vision and pattern recognition* (pp. 136–142). Los Alamitos, California: IEEE Computer Society Press. (Catalog Number 96CB35909.)

Shah, J. (1996b). Curve evolution and segmentation functionals: Application to color images. In *Proceedings of the international conference on image processing* (Vol. I, pp. 461–464).

Shapley, R., & Lennie, P. (1985). Spatial frequency analysis in the visual system. *Annual Review of Neuroscience*, *8*, 547–583.

Shepherd, G. M. (1994). *Neurobiology* (3rd ed.). New York: Oxford University Press.

Shepherd, G. M., & Koch, C. (1990). Introduction to synaptic circuits. In G. M. Shepherd (Ed.), *The synaptic organization of the brain* (pp. 3–31). New York: Oxford University Press.

Simoncelli, E. P., & Heeger, D. J. (1997). A model of neural responses in visual area MT. *Vision Research.* (In press.)

Singer, W. (1993). Synchronization of cortical activity and its putative role in information processing and learning. *Annual Review of Neuroscience*, *55*, 349–374.

Sparks, D. L., & Jay, M. (1987). The role of the primate superior colliculus in sensorimotor integration. In M. A. Arbib & A. R. Hanson (Eds.), *Vision, brain, and cooperative computation* (pp. 109–128). MIT Press.

Spitzer, H., & Hochstein, S. (1985a). A complex-cell receptive-field model. *Journal of Neurophysiology*, *53*(5), 1266–1286.

Spitzer, H., & Hochstein, S. (1985b). Simple- and complex-cell response dependencies on stimulation parameters. *Journal of Neurophysiology*, *53*(5), 1244–1265.

Spitzer, H., & Hochstein, S. (1988). Complex-cell receptive field models. *Progress in Neurobiology*, *31*, 285–309.

Stevens, K. A. (1981). The information content of texture gradients. *Biological Cybernetics*, *42*, 95–105.

Stevens, K. A. (1983). Slant-tilt: The visual encoding of surface orientation. *Biological Cybernetics*, *46*, 183–195.

Stone, J. V., & Isard, S. D. (1995). Adaptive scale filtering: A general method for obtaining shape from texture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *17*(7), 713–718.

Stork, D. G., & Wilson, H. R. (1990). Do Gabor functions provide appropriate descriptions of visual cortical receptive fields? *Journal of the Optical Society of America A*, *7*(8), 1362–1373.

Strang, G., & Fix, G. J. (1973). *An analysis of the finite element method.* Englewood Cliffs, N.J.: Prentice-Hall.

Sundaram, H., & Nayar, S. (1997). Are textureless scenes recoverable? In *Proceedings of the conference on computer vision and pattern recognition* (pp. 814–820). IEEE Computer Society Press.

Super, B. J., & Bovik, A. C. (1992). Shape-from-texture by wavelet-based measurement of local spectral moments. In *Proceedings of the conference on computer vision and pattern recognition* (pp. 296–301). Washington: IEEE Computer Society Press.

Super, B. J., & Bovik, A. C. (1995). Shape from texture using local spectral moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *17*(4), 333–343.

Sutter, A., Beck, J., & Graham, N. (1989). Contrast and spatial variables in texture segregation: Testing a simple spatial-frequency channels model. *Perception & Psychophysics*, *46*(4), 312–332.

Sutter, A., & Graham, N. (1995). Investigating simple and complex mechanisms in texture segregation using the speed-accuracy tradeoff method. *Vision Research*, *35*(20), 2825–2843.

Sutter, A., Sperling, G., & Chubb, C. (1995). Measuring the spatial frequency selectivity of second-order texture mechanisms. *Vision Research*, *35*(7), 915–924.

Tadmor, Y., & Tolhurst, D. J. (1994). Discrimination of changes in the second-order statistics of natural and synthetic images. *Vision Research*, *34*(4), 541–554.

Tamura, H., Mori, S., & Yamawaki, T. (1978). Textural features corresponding to visual perception. *IEEE Transactions on Systems, Man, and Cybernetics*, *8*(6), 460–473.

Todd, J. T., & Akerstrom, R. A. (1987). Perception of three-dimensional form from patterns of optical texture. *Journal of Experimental Psychology: Human Perception and Performance*, *13*(2), 242–255.

Tolhurst, D. J., & Heeger, D. J. (1997a). Comparison of contrast-normalization and threshold models of the responses of simple cells in cat striate cortex. *Visual Neuroscience*, *14*, 293–309.

Tolhurst, D. J., & Heeger, D. J. (1997b). Contrast normalization and a linear model for the directional selectivity of simple cells in cat striate cortex. *Visual Neuroscience*, *14*, 19–25.

Treisman, A. (1986). Properties, parts, and objects. In K. R. Boff, L. Kaufman, & J. P. Thomas (Eds.), *Handbook of perception and human performance* (Vol. II). John Wiley and Sons.

Trevarthen, C. B. (1968). Two mechanisms of vision in primates. *Psychologische Forschung*, *31*, 299–337.

Turner, M. R. (1986). Texture discrimination by Gabor functions. *Biological Cybernetics*, *55*, 71–82.

Turner, M. R., Gerstein, G. L., & Bajcsy, R. (1991). Underestimation of visual texture slant by human observers: A model. *Biological Cybernetics*, *65*, 215–226.

Unser, M., & Eden, M. (1990). Nonlinear operators for improving texture segmentation based on features extracted by spatial filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20*(4), 804–815.

Van Essen, D. C., Anderson, C. H., & Felleman, D. J. (1992). Information processing in the primate visual system: An integrated systems perspective. *Science*, *255*, 419–423.

Van Essen, D. C., DeYoe, E. A., Olavarria, J. F., Knierim, J. J., Fox, J. M., Sagi, D., & Julesz, B. (1989). Neural responses to static and moving texture patterns in visual cortex of the macaque monkey. In D. M.-K. Lam & C. D. Gilbert (Eds.), *Neural mechanisms of visual perception* (pp. 137–154). The Woodlands, Texas: Portfolio Publishing Company.

Van Essen, D. C., & Gallant, J. L. (1994). Neural mechanisms of form and motion processing in the primate visual system. *Neuron*, *13*, 1–10.

Van Hulle, M. M., & Tollenaere, T. (1993). A modular artificial neural network for texture processing. *Neural Networks*, *6*(1), 7–32.

van Santen, J. P. H., & Sperling, G. (1985). Elaborated Reichardt detectors. *Journal of the Optical Society of America A*, *2*(2).

Verghese, P., & Stone, L. S. (1997). Spatial layout affects speed discrimination. *Vision Research*, *37*(4), 397–406.

Victor, J. D., & Conte, M. M. (1989). Cortical interactions in texture processing: Scale and dynamics. *Visual Neuroscience*, *2*, 297–313.

Victor, J. D., & Conte, M. M. (1991). Spatial organization of nonlinear interactions in form perception. *Vision Research*, *31*(9), 1457–1488.

Vincent, A., & Regan, D. (1997). Judging the time to collision with a simulated textured object: Effect of mismatching rate of expansion of object size and of texture element size. *Perception & Psychophysics*, *59*(1), 32–36.

von der Heydt, R. (1995). Form analysis in visual cortex. In M. S. Gazzaniga (Ed.), *The cognitive neurosciences* (pp. 365–382). Cambridge, MA: MIT Press.

von der Heydt, R., Peterhans, E., & Dürsteler, M. R. (1991). Grating cells in monkey visual cortex: Coding texture? In B. Blum (Ed.), *Channels in the visual nervous system: Neurophysiology, psychophysics and model* (pp. 53–73). London: Freund Publishing House Ltd.

von der Heydt, R., Peterhans, E., & Dürsteler, M. R. (1992). Periodic-pattern-selective cells in monkey visual cortex. *Journal of Neuroscience*, *12*(4), 1416–1434.

Voorhees, H., & Poggio, T. (1987). Detecting textons and texture boundaries in natural images. In *Proceedings of the first international conference on computer vision* (pp. 250–258). Washington, DC: IEEE Computer Society Press.

Voorhees, H., & Poggio, T. (1988). Computing texture boundaries from images. *Nature*, *333*, 364–367.

Vos, P. G., Bocheva, N., Yakimoff, N., & Helsper, E. (1993). Perceived location of two-dimensional patterns. *Vision Research*, *33*(15), 2157–2169.

Watson, A. B. (1982). Summation of grating patches indicates many types of detector at one retinal location. *Vision Research*, *22*, 17–25.

Watson, A. B. (1983). Detection and recognition of simple spatial forms. In O. J. Braddick & A. C. Sleigh (Eds.), *Physical and biological processing of images* (pp. 100–114). Berlin: Springer-Verlag.

Watson, A. B., & Ahumada, A. J. (1985). Model of human visual-motion sensing. *Journal of the Optical Society of America*, *2*(2), 322–342.

Watt, R. J. (1987). Scanning from coarse to fine spatial scales in the human visual system after the onset of a stimulus. *Journal of the Optical Society of America A*, *4*(10), 2006–2021.

Watt, R. J. (1988). *Visual processing: Computational, psychophysical and cognitive research.* Lawrence Erlbaum Associates.

Watt, R. J. (1991a). Pattern recognition and visual perception. In R. J. Watt (Ed.), *Pattern recognition by man and machine* (pp. 1–7). Boca Raton, Florida: CRC Press.

Watt, R. J. (1991b). *Understanding vision.* London: Academic Press.

Watt, R. J. (1992). The analysis of natural texture patterns. In G. A. Orban & H.-H. Nagels (Eds.), *Artificial and biological vision systems* (pp. 298–323). Berlin: Springer-Verlag.

Watt, R. J. (1995). Some speculations on the role of texture processing in visual perception. In T. V. Papathomas, C. Chubb, A. Gorea, & E. Kowler (Eds.), *Early vision and beyond* (pp. 59–67). Cambridge, MA: MIT Press.

Watt, R. J., & Morgan, M. J. (1983). Mechanisms responsible for the assessment of visual location: Theory and evidence. *Vision Research*, *23*, 97–109.

Weaver, M. (1993). *An active-symbol connectionist model of concept representation and concept learning.* Unpublished doctoral dissertation, Computer and Communication Sciences, The University of Michigan, Ann Arbor, MI.

Webster, M. A., & De Valois, R. L. (1985). Relationship between spatial-frequency and orientation tuning of striate-cortex cells. *Journal of the Optical Society of America A*, *2*(7), 1124–1132.

Wilson, H. R. (1991a). Pattern discrimination, visual filters, and spatial sampling irregularity. In M. S. Landy & J. A. Movshon (Eds.), *Computational models of visual processing* (pp. 153–168). Cambridge, MA: MIT Press.

Wilson, H. R. (1991b). Psychophysical models of spatial vision and hyperacuity. In D. Regan (Ed.), *Spatial vision* (pp. 64–86). CRC Press.

Wilson, H. R., & Bergen, J. R. (1979). A four mechanism model for threshold spatial vision. *Vision Research*, *19*, 19–32.

Winder, S. A. J. (1998). A model for biological winner-take-all neural competition employing inhibitory modulation of NMDA-mediated excitatory gain. In J. B. Bower (Ed.), *Proceedings of the seventh annual computational neuroscience meeting (CNS'98).*

Witkin, A. P. (1980). A statistical technique for recovering surface orientation from texture in natural imagery. In *Proceedings of the national conference on artificial intelligence* (pp. 1–3).

Witkin, A. P. (1981). Recovering surface shape and orientation from texture. *Artificial Intelligence*, *17*(1), 17–45.

Wolfe, J. M., & Bennett, S. C. (1997). Preattentive object files: Shapeless bundles of basic features. *Vision Research*, *37*(1), 25–43.

Xiong, Y., & Shafer, S. A. (1994). *Variable window gabor filters and their use in focus correspondence* (Technical Report No. CMU-RI-TR-94-06). Pittsburgh, PA: Robotics Institute, Carnegie Mellon University. (A shorter version of this report appears in the *Proceedings of Computer Vision and Pattern Recognition 1994*)

Yang, J. (1992). Do Gabor functions provide appropriate descriptions of visual cortical receptive fields? Comment. *Journal of the Optical Society of America A*, *9*(2), 334–336.

Young, R. A. (1978). Orthogonal basis functions for form vision derived from eigenvector analysis. *Investigative Ophthalmology and Visual Science.* (Abstract)

Young, R. A. (1985). *The Gaussian derivative theory of spatial vision: Analysis of cortical cell receptive field line-weighting profiles* (Research Publication No. GMR-4920). General Motors Research Laboratories, Warren, Mich.: General Motors Research Laboratories.

Young, R. A. (1986). *The Gaussian derivative model for machine vision: Visual cortex simulation* (Research Publication No. GMR-5323). Warren, Michigan: General Motors Research Laboratories.

Young, R. A. (1991). *Oh say can you see? The physiology of vision* (Tech. Rep. No. GMR-7364). General Motors Research Laboratories, Warren, Mich.: General Motors Research Laboratories.

Young, R. A., & Lesperance, R. M. (1993). *A physiological model of motion analysis for machine vision* (Research Publication No. GMR-7878). Warren, Michigan: General Motors Research Laboratories.

Yuille, A. L., & Ullman, S. (1990). Computational theories of low-level vision. In D. N. Osherson, S. M. Kosslyn, & J. M. Hollerbach (Eds.), *Visual cognition and action* (pp. 5–39). MIT Press.

Zeki, S. (1993). *A vision of the brain.* Blackwell Scientific Publications.

Zhu, S. C., & Yuille, A. (1996). Region competition: Unifying snakes, region growing, and Bayes/MDL for multi-band image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(9).

Zhu, S. C., & Yuille, A. L. (1995). *Region competition and its analysis: A unified theory for image segmentation* (Technical Report No. 95-07). Harvard University, Cambridge, Mass.: Harvard Robotics Laboratory.

Zipser, K., Lamme, V. A. F., & Schiller, P. H. (1996). Contextual modulation in primary visual cortex. *Journal of Neuroscience*, *16*(22), 7376–7389.

Zucker, S. W. (1992). Early vision. In S. C. Shapiro (Ed.), *Encyclopedia of artificial intelligence, 2nd ed.* (pp. 394–420). New York: John Wiley & Sons.