High-level Perceptual Contours from a Variety of Low-level Physical Features

by

Brian M. Scassellati

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

Master of Engineering in Electrical Engineering and Computer Science

and

Bachelor of Science in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1995

Signature of Author ....
Department of Electrical Engineering and Computer Science
May 26, 1995

Certified by ....
Lynn Andrea Stein
Class of 1957 Associate Professor of Computer Science
Thesis Supervisor

Accepted by ....
Frederic R. Morgenthaler
Chairman, Department Committee on Graduate Theses

High-level Perceptual Contours from a Variety of Low-level Physical Features

by

Brian M. Scassellati

Submitted to the Department of Electrical Engineering and Computer Science
on May 26, 1995, in partial fulfillment of the
requirements for the degrees of
Master of Engineering in Electrical Engineering and Computer Science
and
Bachelor of Science in Computer Science and Engineering

Abstract

For years, researchers in machine vision have focused on extracting object boundary information from luminance derivatives, color contrast, depth information, and textural patterns. However, no single one of these methods is sufficient for detecting all of the contour information that humans perceive. Humans have little difficulty discriminating depth-based contours from the fused images in a stereogram, extracting illusory contours from a Kanizsa square, or even in completing contours through the blind spot.

This thesis reformulates the problem of "edge detection" into the recognition of high-level perceptual features which I will call "contours". I will present a background on the psychophysical data that describes different low-level physical means of creating contours as well as influences from high-level sources. A novel visual processing framework that deals with contours as high-level image features will be developed. Finally, I will demonstrate the operation of this new model on real-world images using a prototype active vision head that was designed to support this visual architecture.

Thesis Supervisor: Lynn Andrea Stein
Title: Class of 1957 Associate Professor of Computer Science

Contents

List of Figures

Acknowledgments

This work was performed at the MIT Artificial Intelligence Laboratory. Support for this work was provided by an award from the Bose Foundation. Professor Stein's research is supported by the National Science Foundation under National Science Foundation Young Investigator Award Grant No. IRI–9357761. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation. Professor Stein's research is also supported by the Advanced Research Projects Agency of the Department of Defense under contract number F30602–94–C–0204, monitored through Rome Laboratory and Griffiss Air Force Base; by the Mitsubishi Electric Research Laboratories; by the General Electric Foundation Faculty for the Future Award; and by the Class of 1957 Career Development Chair.

I would like to thank Prof. Lynn Andrea Stein for her patience in helping me find a topic, for her encouragement, for her guidance and thoughtful criticisms, and for not waiting to give birth on the day my thesis was due. I also want to thank Rod Brooks for giving the kind of advice only an uncle can give. His dedication to pushing the boundaries of research are a continual inspiration.

My thanks also go out to the members of the Cog lab. Cynthia Ferrell, our den mother, for keeping a watchful eye on us all. To Matt Marjanovic, for his help in building frame grabbers, and for teaching me that another gallon of jello never hurts. To Matt Williamson, for his continual cheerfulness in keeping research a privilege. To Robert Irie, for his help in building the C40 processor board, and for keeping the late-night watch with me. To Nick Schectman, for his practical, and always correct, advice. And to Yoky Matsuoka, for helping me maintain my insanity.

Thanks also go to Gideon Stein, for his assistance in building the C40 boards, and to the two UROP students who were instrumental in building Charlotte: Matt Scheuring and Diego Syrowicz.

I would never have thought to pursue my academic career without the interventions of Prof. Augustus Witt. His faith in my work gave me the confidence to continue through the last five years.

And finally, to Kristi. For carrying the torch to light my way.

It is what lies beneath the surface, the subtle as opposed to the obvious; the hint as opposed to the statement. It is applied to the grace of a boy's movements, to the restraint of a nobleman's speech and bearing, or when notes fall sweetly and delicately to the ear... To watch the sun sink behind a hill, to wander on and on in a huge forest with no thought of return, to stand upon a shore and gaze after a boat that drifts beyond sight, to ponder on the journey of wild geese seen, and lost behind clouds, these are the ways of yu-gen. - Zeami

Chapter 1

Introduction

People are incredibly adept at finding edges. Without conscious thought, we can guide our hands to the edge of a soda can, determine where the hole in a wooden fence lies, and spot a deer hidden in dense undergrowth. Computational attempts to extract edges from static images have become increasingly more accurate in recent years, but they still fall short of biological systems. In this thesis, I propose that one reason for this discrepancy is the variety of methods available to biological systems for extracting boundary information from scenes. This work presents a brief review of the shortcomings of current approaches to studying boundary extraction problems and suggests a model for integrating various existing methods of edge detection. A prototype active vision head was constructed to allow testing of this novel visual architecture and a partial implementation of this model was implemented.

1.1 Motivations: A Visual Turing Test

Artificial Intelligence researchers have taken as their goal, at the highest level, to study the computational aspects of intelligence. Whether our actual research is composed of building insect-like robots, solving computational logic problems, proving theorems about perceptron learning, or retrieving images from large databases, we all base our work on the desire to implement and understand intelligence. The field that I have chosen to specialize in is machine vision. Why begin with vision? Humans rely on vision for much of their perceptions of the external world. Our descriptions of objects are dominated by visual perception. (When did you last hear "use the book that smells like vinegar" instead of "use the red book"?) From a biological standpoint, a large percentage of brain tissue is directly involved in visual processing. Perhaps the most realistic reason to study computational vision is that the biology of vision has been much more clearly studied than that of other sensory systems. Because vision can easily be tested in both awake and anesthetized human and animal subjects, a wealth of data on the visual process has been accumulated in the fields of psychophysics, psychology, and physiology to guide our computational investigations.

There are many different motivations that can cause a researcher to begin the study of computational vision. Some are interested in solving difficult engineering tasks such as visual navigation for mobile robots, or recognition of faces. Other researchers are motivated by a desire to investigate large scale parallel theories of computation. Others are interested in industrial and military tasks like tracking enemy tanks or assembling parts on a conveyor belt. Finally, some researchers begin to study vision in order to understand the way that human beings are capable of recognizing friends, catching fastballs, and operating in such a visually complicated environment. It is this last area that I find most interesting, and it is this motivation to understand human vision that has resulted in a re-evaluation of computational vision.

As a step towards understanding artificial and biological vision, I propose a type of visual Turing test: The challenge is to create a machine that gives behavioral responses to a visual scene that are indistinguishable from those of a human observer. At the extreme, this problem requires the machine to have solved many of the traditional machine vision tasks such as object recognition, tracking, and figure/ground segmentation, as well as the additional capabilities of memory, basic reasoning, and motor control. While it is not necessary to set out to complete this goal in one fell swoop, it does serve as a guiding influence for this work.

Artificial vision is too large a puzzle to solve all at once. The piece that I have chosen to examine is the problem of contour detection. As will be demonstrated later, contours can be caused by differences in luminance, color, depth, or a variety of other low-level features. Contours can also be created by high-level expectations, influences from other senses, and other effects. This thesis will present a novel methodology for combining these low-level features and high-level influences into an integrated perceptual contour representation. A custom-built active vision robot named Charlotte was constructed to provide a platform for experimentation with this new visual representation (see Figure 1-1). These experiments indicate that the combined contour representation demonstrates better performance on a class of stimuli.

1.2 Organization of Thesis

The remainder of this thesis is organized into the following eight chapters:

Chapter 2: Problems in Current Visual Architectures. We begin by examining a conglomeration of visual architecture models, which I shall call the perfect slate model. Two strong objections to this type of modeling are presented, and guidelines for construction of a new model are outlined.

Chapter 3: What is a Contour? The third chapter outlines the basic problems of contour detection. Details on the types of physical properties that define contours and the computational approaches for extracting those properties are discussed.

Chapter 4: The Integrated Contour Model. In the fourth chapter, we begin to build a new visual cognitive architecture for perceptual contour detection. A method for integrating low-level feature detectors and high-level conceptual information is outlined.

Chapter 5: Charlotte: An Active Vision Platform. The fifth chapter describes the design and construction of Charlotte, an active vision platform that embodies some of the objections to standard machine vision systems that were described in Chapter 2.

Chapter 6: Implementation of Integrated Contours. The sixth chapter presents the actual implementation of the visual architecture for contour identification developed in the preceding chapters.

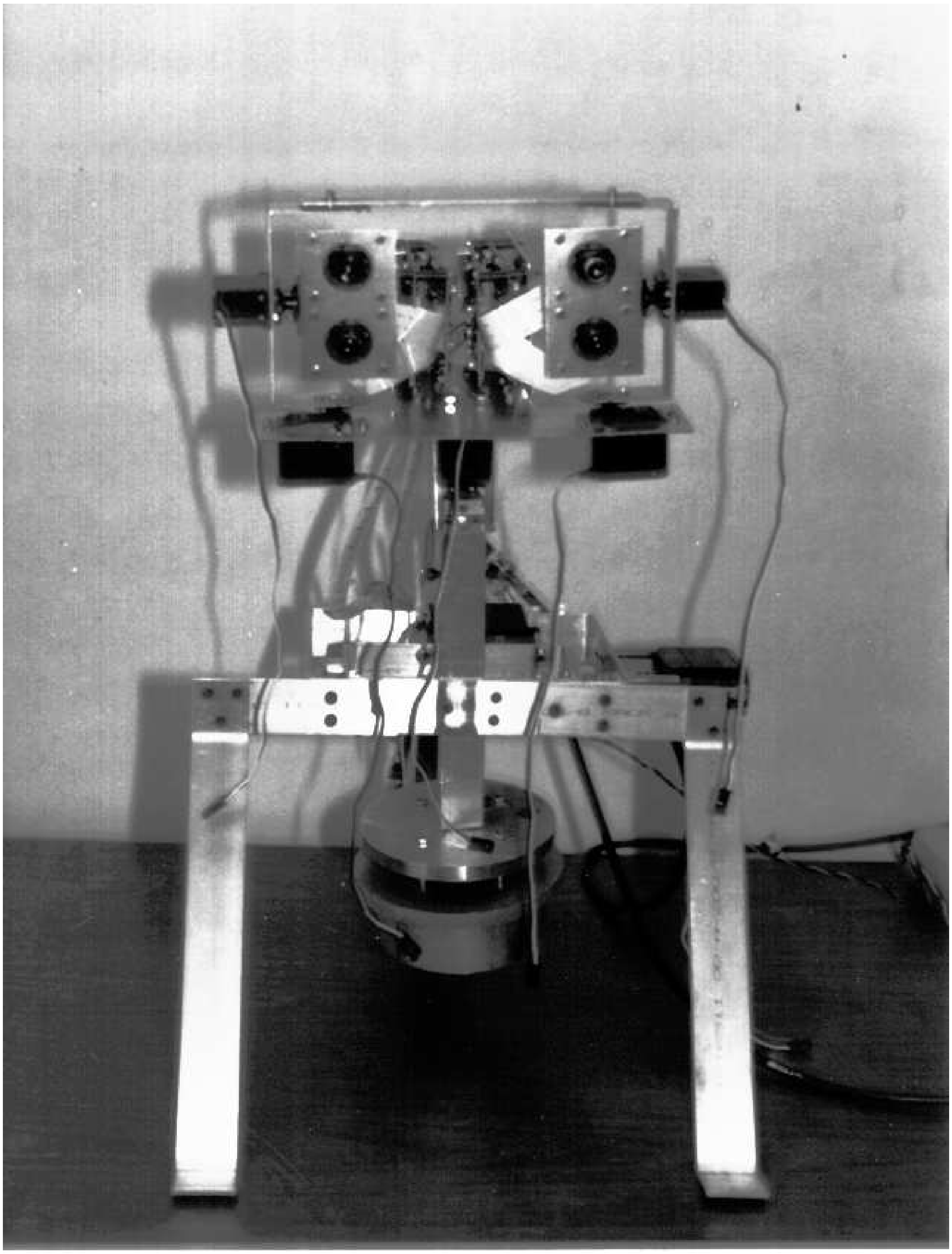

Figure 1-1: Charlotte: An active vision platform.

Chapter 7: Experimental Results. Experimental comparisons between the integrated perceptual contours and individual low-level modalities are presented.

Chapter 8: Conclusions. In the final chapter, the major contributions of this work are summarized and future avenues of research are proposed.

Chapter 2

Problems in Current Visual Architectures

While the studies of computational and biological vision have unearthed many of the complexities and the wonders of vision, scientists still rely upon simple models to begin their investigations. Perhaps the simplest of these models is the "pinhole camera" model used extensively in early optics and vision courses. In this section, we will consider a generalization of the pinhole camera model which I will call the "perfect slate" model. This generalization combines many of the assumptions that machine vision researchers have made in their first attempts at an overall model. Two objections to this model will be detailed, and a few guidelines for building visual architectures will emerge.

2.1 The Pinhole Camera and the Perfect Slate Models

Almost all computer vision (and many biological vision) texts begin with the "pinhole camera" model of vision. Light reflects from surface objects, passes through a focal point, and impinges on some type of recording surface. This simple model, while providing a basic bootstrap onto the many challenging problems of vision, is noticeably different from what happens both in the human eye and in mechanical camera systems. Lens distortion effects, blooming (or saturation in biological systems), and many other problems leave the pinhole model as an acknowledged simplification of a much more complex problem. Any vision researcher will be quick to point out these flaws. To avoid many of the most common criticisms of this model that have already been effectively dealt with by the vision community, we will instead examine a conglomeration of current visual architecture models in order to expose a few basic theoretical assumptions that hinder further progress. The individual theories that I have lumped together under this single heading all share a common theoretical substrate, and it will be convenient to refer to them collectively as the "perfect slate" model.

In the perfect slate model, light originates from an external source, is reflected and refracted by surface objects and then travels towards the camera. Light is focused by a lens, passing through a single common point (the focal point). The light is then projected upon a rectangular photosensitive field. The photosensitive region is composed of equally-spaced, identical, ideal imaging elements. These imaging elements (or pixels) record the intensity of the light striking their surface, and sometimes the wavelength of the light as well. The perfect slate models then attempt to convert pixel values into a coherent description of the real-world objects that originally reflected or emitted the light that struck the photosensitive array. These computations can be very complicated and computationally expensive, but they are generally framed as a series of mathematical or filtering operations upon the pixel values. The information contained in the pixel values is processed by a series of increasingly more complex operators to derive higher level properties which are then combined into a world model.

2.2 Objections to the Perfect Slate

While this theory is well-known in some form to most students in computer science and artificial intelligence, it was never intended as a description of biological vision systems except on the most coarse level. This departure from biological visions systems has resulted in a task as hopeless as trying to build a working replica of a locomotive from a photograph. In recent years there has been a resurgence of biologically motivated vision models and task-based systems. However, many of these "biologically plausible" attempts have missed a few critical departures that early researchers made from the biological models in order to build operational systems. There are two assumptions generally made by the perfect slate model that I will challenge in this thesis. The first regards the perfection of the photosensitive slate, and the second the method by which higher level information is obtained. In evaluation of these problems, I will return to the human visual system as an example of operational systems, just as an engineer should be sure to look at a real locomotive while building his model.

2.2.1 First Objection: The "Perfect Array" Assumption

The first objection to the perfect slate model is the nature of the stimuli available for processing. Most vision algorithms begin not with the light reflectances of real-world objects, but with the intensity values of a perfect, equally-spaced, rectangular grid. The advent of inexpensive CCD cameras has made this assumption even easier to accept at face value. However, if we examine the biological systems that are capable of vision (or of any other sensation), it is easy to see that the nature of the sensory cells is of great importance to the types of processing that the entity is capable. For example, consider the jumping spider. The retinal structure of the spider contains <-shaped and >-shaped photoreceptors. These oddly shaped receptors have evolved to match the size and shape of the legs of other jumping spiders. This adaptation allows for quick recognition of potential mates. However, without additional input, the spider cannot discriminate between potential mates and potential prey. While another organism might be capable of the same discrimination given a rectangular grid of photoreceptors, this example demonstrates that the types of sensors available greatly effects the processing necessary for a given task. The perfect slate model ignores the importance of the means of collecting sensory data, both in the nature of

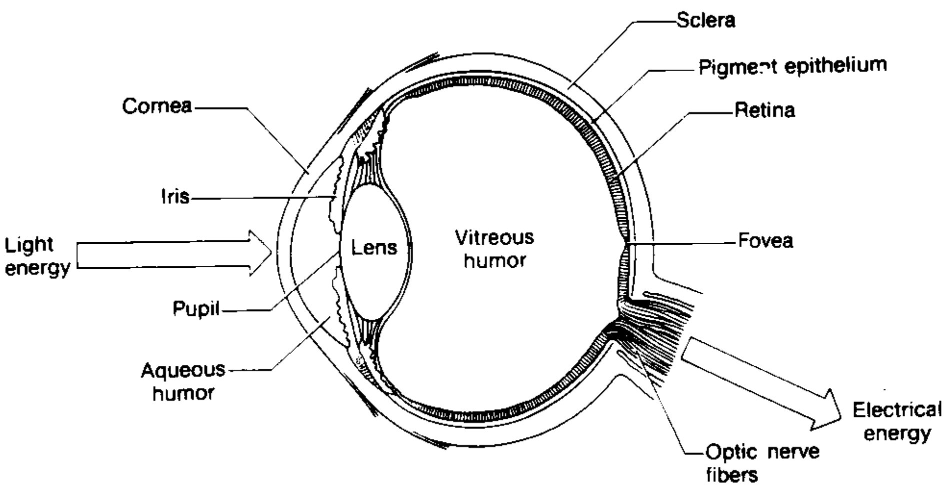

Figure 2-1: Gross Anatomy of the Human Eye: Light striking the photoreceptors on the back of the retina must pass through many layers of tissue and fluids before reaching its destination. From [Gol89, p. 71].

the stimuli available to the receptors and in the nature of the receptors available to receive the stimulus.

In the human eye, light undergoes a variety of transformations before it strikes the photoreceptors in the retina. As light enters the eye, it is distorted by imperfections of the lens and cornea. After light enters the eye through the cornea, it must pass through the aqueous humor and the lens, through the vitreous humor to the back of the eye, through layers of translucent retinal cell bodies, and through a layer of veins and capillaries before it strikes the photoreceptive cell bodies at the back of the retina (see Figure 2-1). While passing through these different media, the light from a single point diffuses and projects onto an area of retinal photoreceptors. From a biological standpoint, this dispersion serves a vital role. Since all neurons have random spontaneous action potentials, in order to ensure a stable image we need only consult the surrounding neurons to determine whether a given neuron is firing as the result of incoming light or merely a spontaneous excitation. On a computational level, this appears to be only a loss of resolution that is imposed by biological limit. However, there are features of this system that are not accounted for simply by changing the resolution of the artificial system. The biological system allows for a first level of failure recovery, since the actions of a single individual cell cannot result in a false

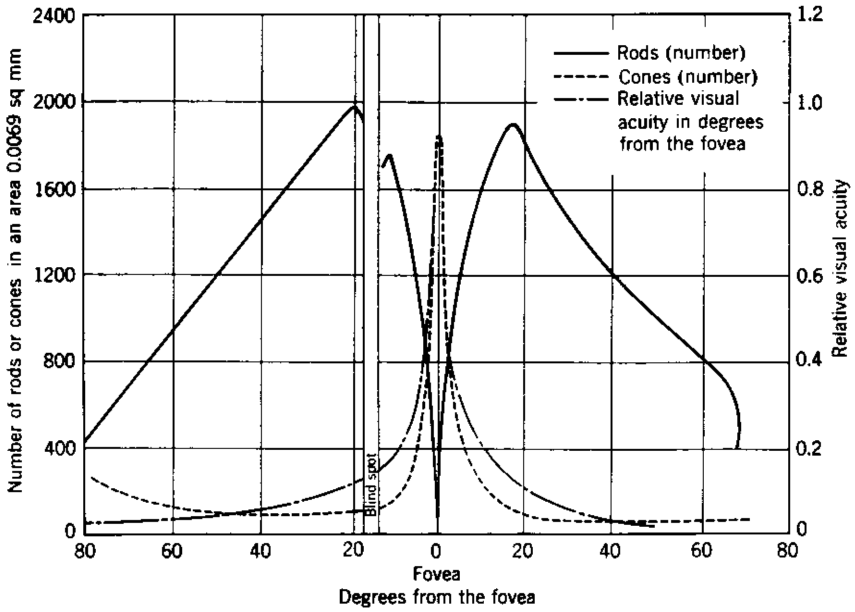

Figure 2-2: Density of retinal photoreceptors as a function of location. Rod cells are prevalent in the periphery of human visual field, while cone cells are prevalent in a densely packed region called the fovea. At the optic disc, axons from the photoreceptors that form the optic nerve crowd out photoreceptor cell bodies. From [Gra65, p. 49].

response. Artificial vision systems can often be greatly misled by flipping a single pixel value from black to white, or even from one shade of grey to another.

The retina also does not have the perfectly-packed, equally-spaced rectangular array of photoreceptors. We must first recognize that the retinal cells are not even identical in their responses to a given stimulus. Some retinal cells, called rods, are generally sensitive only to luminance changes, are very sensitive to moving stimuli, and generally have very large receptive fields. Other cells, called cones, are color sensitive, generally have a small receptive field, and have one of three wavelength tuning curves. However, even within one of these classifications the individual cells are each unique and react differently from their fellows. Retinal cells also vary greatly in the density of their packing (see Figure 2-2). In a region called the fovea, cone cells are packed very densely. In the area called the optic disc, the axons of the photoreceptive cells push through the retina to form the optic nerve, leaving no space for any receptors. In most other places, photoreceptors follow a general hexagonal tiling, increasing in receptive field size as their distance from the fovea grows. The types of receptors, their locations, and positionings, all serve to make the human visual system very different from the perfect slate model. These biological peculiarities have a great impact upon the types of visual behavior that humans exhibit. For example, visual attention can more easily be simulated if you consider the proportion of foveal cells to non-foveal cells. This packing density provides more information toward the direction of our gaze. Similarly, if we cannot read the lettering on a sign in the periphery of our vision, the simplest solution is often to simply saccade to that point. These differences will resurface again when we discuss the types of stimuli that produce high-level contours.

To start addressing these problems, a novel active vision platform was designed and built at the MIT Artificial Intelligence Laboratory. The design of this robot, called Charlotte, is described in Chapter 5.

2.2.2 Second Objection: The "Direct Computation" Assumption

The second objection that I will raise with the perfect slate model concerns the computations that extract higher level traits from the low-level pixel values. Most previous attempts at early visual processing have characterized the flow of information from receptive cells (or pixels) to higher level conclusions (or world models) as a feed-forward pipeline. Pixel values are processed by a series of convolutions, differentiations, subtractions, and other operations to derive high-level properties. To illustrate my objections to this assumption, I will rely upon three different aspects of human physiology and psychophysics: top-down processing in the human brain, influences on vision from other sensory modalities, and the "filling-in" effects of the blind spot.

The abstraction of visual processing as a uni-directional pipeline is computationally and conceptually simple, but removes a great deal of functionality from our model. Students in introductory neuroscience classes (and often computer vision sur-

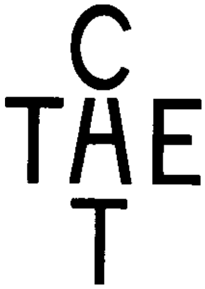

Figure 2-3: Demonstration of the effects of context. The same visual stimulus can be perceived as either the letter "A" or the letter "H". From [And90, p. 76].

vey courses) are presented with detailed information of how information is passed from the simple receptive fields of retinal cells to the lateral geniculate nucleus and then to cortical areas V1 and V2, but very little is said about the equal number of projections that travel in the reverse direction. Neuroscience has given us a great deal of information on how individual retinal receptive fields map into the retinotopic maps of V1, the orientation selective cells of V1, and the luminance differential cells in the lateral geniculate body, but relatively little is known about the enormous number of projections that travel from upper cortex back to V1 and the lateral geniculate. The results of this top-down processing can easily be observed from a psychophysical level. For example, the classic CAT/THE context example shown in Figure 2-3 demonstrates the influence of higher-level processing upon "low level" features. If vision is a purely feedforward process, why does the perception of the central letter change from being an "A" when read vertically to an "H" horizontally? This figure makes clear that contextual effects from higher level sources can affect perceptual routines that have been traditionally labeled as low-level. The perfect slate model of vision does little to address the issues of higher-level influences, and most researchers have been content to address only a "low level" function that does not rely upon higher level information.

Even allowing for entry points from higher level visual processing, there is still information that humans use in visual perception that is not represented in our model. Even as a two-directional pipeline, our model of visual processing is lacking the inputs from other sensory modalities that biological systems utilize. Ramachandran has

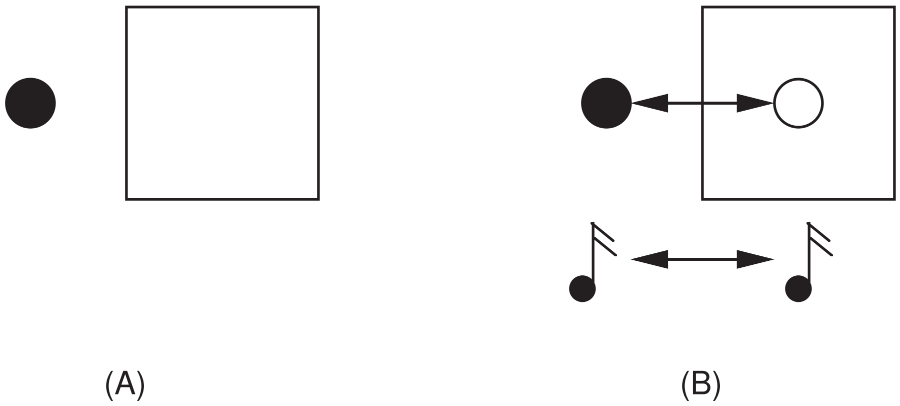

Figure 2-4: Experiment by Ramachandran showing the influence of audition on induced motion. A flashing dot beside a square has no induced motion. The addition of alternating left-right tones causes induced motion and implicit occlusion. From [CRS94, p. 31].

shown a simple example of auditory stimuli effecting visual perceptions [CRS94]. A subject is shown a single flashing dot to the left side of a display, while a solid box fills the other (see Figure 2-4). Without auditory stimuli, there is nothing unusual about this display. However, if subjects are presented with a series of tones that alternate from left ear (while the flashing light is on) to the right ear (while the flashing light is off), the subject perceives the dot to move back and forth across the display, becoming obscured by the box at the right side. The addition of the auditory stimulus is sufficient to induce apparent motion in a display where none previously existed. The influences of audition, and the other senses, should be an integral part of our model of visual processing.

Perhaps the most challenging of phenomena to explain with a pipeline processor model is the "filling in" effects of the blind spot and scotomas. In the patchwork of retinal receptive fields there is a patch called the optic disc which is devoid of receptors. This missing patch creates an area where no sensory input is available, that is, a blind spot. However, in our perceptual experiences, these missing patches are hardly ever noticed; only in extreme circumstances does the missing information become apparent. Our visual processing routines cover over the missing spot by "filling in" the lost data. While this may seem to be a hardwired response to a biological necessity, the same result can be seen in accident victims with retinal damage. Damaged portions of the retina leave regions of the visual field, called scotomas, that are insensitive to light stimuli. These scotomas exhibit the same filling-in behavior that is seen with the blind spot. Our perfect slate model does little to explain the methods by which missing information is reconstructed to form a coherent perception of the entire visual field.

The problems of top-down processing, influences from other sensory modalities, and filling-in behavior offer serious challenges to our perfect slate model of computation. But how seriously should we take these challenges? Until we have a complete understanding of biological (or artificial) machine vision systems, there will always be exceptions to our models. In the next section I will detail arguments that indicate the necessity of accounting for these missing variables.

2.3 Are these Objections Sufficient to Abandon this Model?

Scientists develop models to provide a simpler means of understanding complex data or computations. No model is designed to explain every detail of the environment that it represents, only to provide a framework that works well enough for most situations. However, a model must go far enough in explaining phenomena so that it gives an accurate portrayal of what is occuring. In examining the perfect slate model, we must ask two questions: What exactly are the situations that we are trying to model? and Has our model gone far enough in explaining these visual phenomena?

While most researchers refer to this field as "machine vision" or "artificial vision," it is important to remember that our real goal is not to study vision itself, but perception. As was described in the introduction, our goal is to strive towards a visual Turing test, a machine that interprets visual scenes in the same way that humans do. To keep this goal in sight, we must remember that it is the final results of our system that are important, not the individual processing stages themselves. The perfect slate model gives only a simple explanation for the early vision architecture and processing stages. Perfect slate models do nothing to describe the end behaviors desired by the system. For our purposes, this model has an insufficient goal.

Our next dilemma is in deciding what behaviors are critical to model. Are the activities of the blind spot and the differences in size of receptive fields of importance to modeling human perception, or are these anomalies that need not be represented in our model? While we can certainly debate the saliency of each of these anomalies to our perceptual experience, perhaps it is best to approach this issue from a more systematic perspective. In preparing this thesis, I was introduced to many of the rigorous methods of electrical engineering, and most notably circuit board debugging. Electrical engineers talk of an unknown circuit design as a "black box," which has a specific set of inputs and a set of outputs which are a function of the inputs. Debugging a circuit is the methodology of forcing the input-output characteristics of the circuit to match the conceptual model of what the circuit should do. By examining the cases where the black box works (or fails) as your model predicts, you can determine which subcircuits are operational. But the most interesting and useful cases are those in which the black box does something that the model does not predict, since these give insights into the actual workings of the circuit. If we look at vision as a black box, it becomes beneficial to examine anomalies of visual processing because they place limitations on the actual circuitry within this black box. This does not directly imply that we need to model all of these anomalies in our visual architecture, only that we need to understand the insights that they provide.

Has the perfect slate model gone far enough in explaining the anomalies of visual processing? Probably not. When the number of phenomena that a model classifies as "anomalies" grows to be too large, the model is no longer explaining the natural behavior of the system. As Churchland, Ramachandran, and Sejnowski have noted:

> To be sure, a theory can always accommodate any given "anomaly" by making some corrective adjustment or other. Nevertheless, as anomalies accumulate what passed as corrective adjustments may come to be deplored as ad hoc theory-savers. A phenomenon is an anomaly only relative to a background theory, and if the history of science teaches us anything, it is that one theory's anomaly is another theory's prototypical case. Thus "retrograde motion" of planets was an anomaly for geocentric cosmologists but a typical instance for Galileo; the perihelion advance of

> Mercury was an anomaly for Newtonian physics, but a typical instance for Einsteinian physics. Any single anomaly on its own may not be enough to switch investment to a new theoretical framework. The cumulative effect of an assortment of anomalies, however, is another matter. [CRS94, p. 33]

The perfect slate model, like Newtonian mechanics, serves its purpose as an introductory model to give a rapid overview of the visual system architecture, but fails to provide a usable model of real systems.

2.4 Methods of Modeling a Visual System

How then are we to approach our study of the visual system? In this section, I will outline three basic guidelines for studies of the visual system. Some of these will be obvious from our previous discussions of the perfect slate model, but their summary here should serve to refine their descriptions and emphasize their importance. The three guidelines are:

1. Make clear your motivations.

2. Discriminate between physical and perceptual properties.

3. Divide the problem into meaningful, but smaller pieces.

While these are not intended as a comprehensive listing of the types of considerations necessary in building a visual architecture, they will be sufficient for our purposes.

The first pitfall that we must avoid is the problem of motivation. Why are we interested in building this visual architecture? If our purpose is to enable a machine to successfully manipulate the tip of a soldering iron across parts on an assembly line, then the visual architecture should be concerned with speed and accuracy. If our purpose is to investigate interesting algorithmic solutions to the problems of compact image representations, we should be concerned mainly with data structures. If we are interested in building a navigation system for a mobile robot, error recovery and robustness might be our primary concerns. While this point has been mentioned before, it deserves a special place here. We cannot begin constructing models without

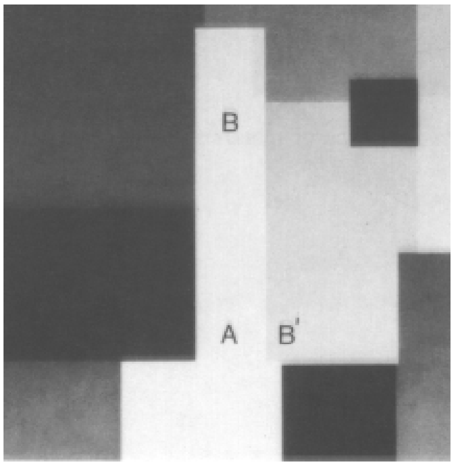

Figure 2-5: Luminance Mondrian showing the distinction between physical and perceptual properties. Both points B and B' have the same physical luminance, but different perceptual brightness. From [MS83, p. 539].

a clear goal in mind, nor can we blindly apply models that have a specific goal to all problems in vision research. For the remainder of this thesis, we will take the visual Turing test as our ultimate motivation. The behavior that we will be most interested in modeling for this work will be object segmentation, that is, the ability to discriminate an object and its subcomponents from other objects.

The second pitfall lies in the blurred lines between physical and perceptual properties. Physical properties are those real-world phenomena that can be quantitatively measured independent of an observer. For example, the wavelengths of light reflected by a surface and the luminance of a light source are physical properties. Perceptual properties are cognitive beliefs that are relative to a given individual. For example, the color of a surface and the brightness of a light source are perceptual properties of objects. While it is often easy to confuse physical and perceptual properties, the distinction is vital to constructing a proper visual architecture. For example, the points labeled B and B' in Figure 2-5 both have the same luminance value, but B is perceived to be brighter than B'. The nature of the perfect slate model often results in the confusion of perceptual with physical properties; when simply following a series of mathematical operations on a series of pixel values it is often easy to forget that the goal is not merely to transform the physical stimulus values, but to simulate the perceptual results of the system.

The last guideline that I will offer in this section is to be sure to break the problem down into smaller, but meaningful pieces. Determining what are the correct pieces of a problem to work on is often the most challenging problem of research. If you begin with the wrong pieces, you may find interesting results, but they will never fit back into the big picture. The part of vision that this work will cover is the study of contour detection. The next chapter will discuss the nature of this problem, and attempt to justify contour detection as a meaningful piece of vision.

Chapter 3

What is a Contour?

Shape is one of those elusive ideas that people have very little difficulty in using, yet have great difficulty in defining. No simple mathematical models fully account for the ways that people describe shapes in simple, elegant methods [Mum91]. This thesis is concerned with the extraction of boundary information using vision, but what exactly does this imply? This chapter will begin to explore the basis of shape and the methods by which a contour can be created.

In the pursuit of a solution to the visual Turing test, the problem of contour detection is an interesting research subgoal since many different high-level tasks can be greatly simplified with the same basic contour detection subroutines. Contours are vital for tasks involving discrimination of figure/ground relationships, such as object discrimination [BW93], robot navigation [Hor93], and manipulation [HI84]. Contours can also be used to track objects [KWT87], or to find motion in a scene [AP93]. Higher level cognitive tasks, such as trying to solve a maze, reproducing a pen-and-ink drawing of an object, or determining where a wire on a circuit board leads, will also be much simpler to construct with a contour detection subroutine.

To help differentiate between physical and perceptual stimuli, the first section of this chapter will be devoted to developing strict definitions that will be applied to the words "edge" and "contour." The basic properties of a contour will then be defined, as well as the methods by which we can differentiate contours from other phenomena. The majority of this chapter is devoted to exploring the types of physical

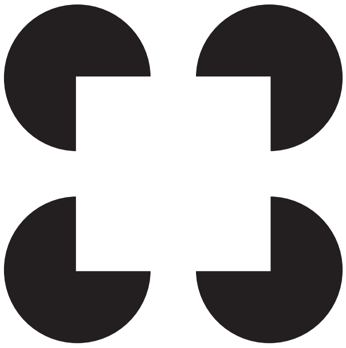

Figure 3-1: The Kanizsa Square. While there are no luminance, depth, or color differences to mark the boundary, the shape of a square superimposed on four circles is easily perceived. The Kanizsa square is one example of a stimulus that results in a contour where no physical properties are involved. Adapted from [Kan79].

properties and other high-level influences that can cause contours. Psychophysical justification for deriving contours from the physical properties of luminance, color, texture, motion, and depth will be detailed, and a brief review of the artificial vision solutions to these modalities will be discussed. Other sources of contours, such as illusory contours from the Kanizsa square (see Figure 3-1), filling-in phenomena of the blind spot, and expectations, will also be discussed.

3.1 Basic Definitions

So far, the words "edge", "boundary", and "contour" have been loose synonyms. However, among the communities of computer science, cognitive science, psychophysics, and physiology there are different usages for each of these words. Even more confusing are the variety of computational studies that use the word "edge" as an indication of both physical and perceptual phenomena. To avoid confusion, careful examination of the definitions used in this work is necessary.

I will use the word "edge" to refer to the contributions of a single physical property (or other source) to detection of boundary information; the results of a Canny operator are luminance-based edges, the results of the Marr-Poggio-Grimson stereo algorithm are disparity-based edges, and the illusion of the Kanizsa square creates Gestalt-based edges. The integration of many of these basic modalities results in a perceptual phenomenon that I will call a "contour," that is, a stable visual effect that divides two regions of similarity. A contour is a high-level perceptual property that is composed of a variety of low-level physical properties and interactions from other sources. Finally, I will use the word "boundary" to denote any separation between two regions that have particular features.

As we will see later, boundaries are a superclass of contours, which are composed partly of edges. Boundaries also include some transient phenomena that are not properly contours, and contours can be constructed out of properties that are not edges. While these distinctions may seem somewhat arbitrary, they will be invaluable in building a model of contour processing to perform the task of object segmentation. Let us look more closely now at the definition of contour.

I have claimed that contours are composed of a variety of low-level physical properties and interactions from other sources. A contour can certainly be made from only a single set of edge information, as you are now doing in discriminating the luminance-based edges of the letters of this sentence. Contours of naturally occuring objects also tend to have coinciding edges along many different modalities; the edge of a table has luminance, spectral, and disparity differences from the background. A contour can also be created without any purely physical properties, as occurs in the Kanizsa square (see Figure 3-1). Contours can be viewed as a type of abstraction for the many physical edge detection methods and the other contributing effects. However, the contour abstraction does not mask which individual physical edge detection modalities contributed to the perception; the concept of contour does not simply take the place of the edge information. There is no confusion between a contour that results from a luminance difference and one that results from changes in texture. The fact that this information is available to the conscious mind distinguishes contours as a high-level phenomenon unlike many other mental processes. For example, many memory retrieval models have proposed that there are a variety of low-level storage methods for maintaining memories. However, the type of storage used for any particular memory is unknown to the conscious individual.

I have also made the claim that contours are perceptual properties. Is a contour

Figure 3-2: Demonstration of the perceptual nature of contours. With no knowledge of the organization in the picture, observers often fail to see the perceptual contour of the Dalmation. Once the photograph has been annotated, the perception of the dog's contour is unmistakable. Photograph by R. C. James, from [Gol89, p. 197].

actually relative to the observer? Anyone will agree where the boundary of a table lies, or where the outlines of these letters are located. However, consider the case of Figure 3-2. If this figure is unfamiliar to you, it is easy to pick out the individual contours of the black spots, but there is no obvious organization to them. However, once you are shown the outline of a dog within the photograph, the contour separating the dog from the background can be seen anytime the image is viewed; the contour is perceptually stable and unmistakable. The physical stimulus has not changed, yet there is a dramatic organizational difference. For this figure, the contour of the dog is certainly a perceptual property.

While looking for the dog, or while fusing stereograms, observers often see transient edge-like connections between various parts of the image. Intuitively, these

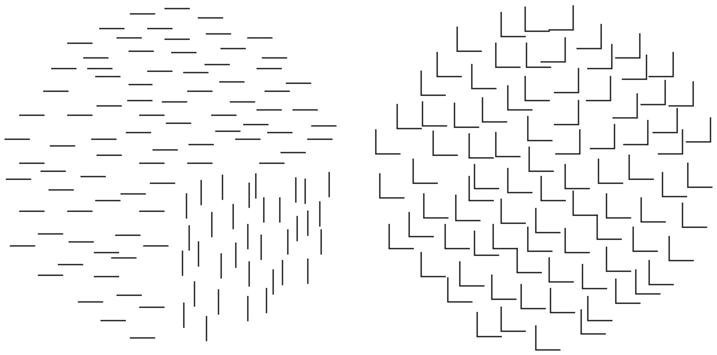

Figure 3-3: Demonstration of the pop-out effects of contours. The boundary between the vertical and horizontal lines in the left circle is a contour because it is distinguished immediately and automatically. The boundary between the left-facing angles and the right-facing angles in the right circle is not a contour since it requires attention to detect and maintain. Stimuli from Olson and Attneave, as reported in [Gol89, p. 210].

transients seem to be more the intermediate steps of some part of the contour processing mechanism than the final results. Can we classify these transient responses as contours, or are they some other phenomenon? If transients are allowed within our definition of contours, we must then address many more issues on the creation and destructions of these perceptions. Why would these transients disappear? Can transient phenomena still reflect processing on a stable world model? To simplify these issues, we restrict our concept of contours to stable phenomena. By this definition, transients are boundaries, but not contours.

Contours can be distinguished from transient phenomena, acts of imagination, and cognitive pattern recognition by their stability and their automatic appearance. Contours seem to "pop out" from an image, and require little or no attention to maintain them. For example, the boundary between the vertical and horizontal lines in the left circle of Figure 3-3 is easily distinguished, while the boundary between the regions in the right circle is much more difficult to establish. The boundary in the left circle is an example of a contour; it appears automatically upon viewing and requires no attention to maintain. The boundary in the right circle is not a contour; it requires visual search to determine the boundary between the regions and this boundary requires conscious attention to maintain. Note that the stimuli for both patches are composed of the same vertical and horizontal components, yet the boundary is a contour in only one instance.

3.2 Physical Properties that Cause Contours

Now that we have a basic working definition for contours and edges, we can begin to determine the types of edge detection components that contribute to a contour detector. In this section we will begin with the most obvious physical properties that result in contours: luminance, spectral composition, texture, binocular disparity, and motion. In each case, we will begin with an example of the type of phenomenon we are considering, followed by a justification of the psychophysical properties that identifies the modality as a contour. For each of these modalities, we will also review some of the computational methods that have been developed. In the following section, other methods of arriving at contour information will be examined. These two sections are not intended as a comprehensive listing of all possible methods for obtaining contour information, nor as an in-depth study of any of these specific methods. The implementation of the complete set of these methods is also beyond the range of this thesis. However, these descriptions should serve as a guideline for the types of processing that may be necessary for developing a computational model of contour perception.

3.2.1 Luminance-based Contours

The most obvious, and perhaps most well studied, method of gaining contour information is based on the luminance of a scene. Luminance-based contours are the result of differences in the intensity of light striking the retina. The psychophysics of this example are almost trivial; the dark lines on a page appear immediately and automatically. Luminance edge detection in the human visual system is accomplished by combining information from local patches of photoreceptors to form specialized receptive fields. By combining the positive influence from a central region with the negative influences from the surrounding cells, a simple center-surround receptive field

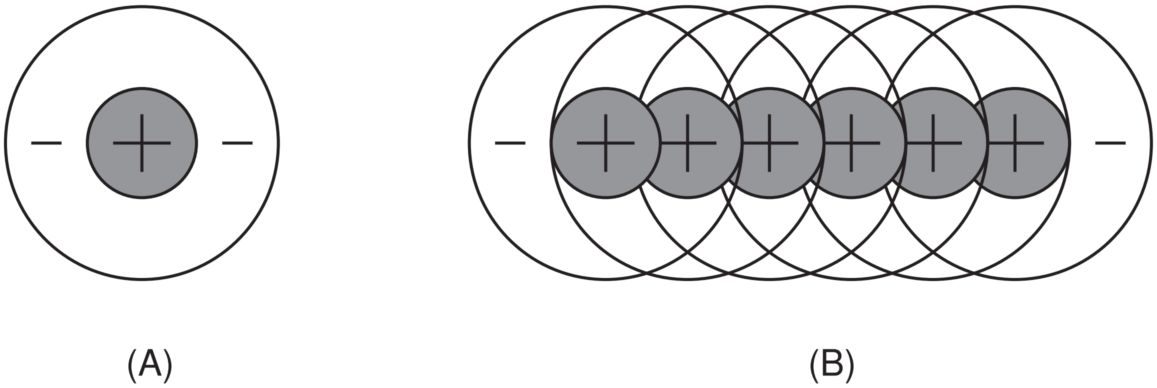

Figure 3-4: Luminance edge Detectors constructed from center-surround receptive fields: A single center-surround receptive field (A) takes positive feedback from a central region and negative feedback from the surrounding area. A simple edge detector (B) can be created by overlapping center-surround fields.

that responds to point stimuli is formed. By combining center-surround cells along a row, a simple edge detection scheme can be constructed (see Figure 3-4). David Hubel and Torsten Wiesel first located these feature-detector cells in human cortical area V1 in 1963 [Gle91, p. 208].

The luminance-based edge detection problem has been extremely well studied in machine vision literature. While a full survey of the methods of extracting luminance-based edges is not necessary for our purposes, examining two of the most common algorithms will provide a basic understanding of the complexity of this task. The simplest means of detecting luminance contours is to convolve a simple local filter with the pixel array, much like the processing that occurs in the center-surround cells in the human visual system [Hor86]. Oriented edges can be detected by convolving an image with a simple local filter, like the one shown in equation 3.1 below:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{3.1}$$

This filter is sensitive to horizontal and vertical step changes in luminance, while allowing for a slight smoothing of the input data. Luminance edge locations in the filtered image are marked by a zero-crossing, that is, locations where the image values change from positive to negative or from negative to positive [Sob70].

A great deal of effort has been expended on finding filters that locate edges in all directions while filtering out the desired amount of noise. Perhaps the best known of these studies was the thesis of John Canny [Can83]. Canny defined three criteria that a luminance edge detector should capture: First, that the detector have a low error probability. Second, that the points marked as edges should be as close as possible to the true edge. Third, that there be a low probability of responding to the same edge twice. Canny then developed a set of operators of varying width, orientation, and length that when convolved with an image gave results that were optimal for his three criteria.

3.2.2 Spectral-based Contours

Another intuitive means of determining contour information is from color. As described in section 2.2.1, the human visual system contains three types of cone cells with different spectral characteristics. These cone cells combine to give information about the distribution of wavelengths striking the retina. From the physical property of the spectral decomposition of light, the perceptual sensation of color is determined. Color, like brightness, is a perceptual property as can be demonstrated by color Mondrians similar to the luminance Mondrian shown in Figure 2-5 (see [MMT76]). As was possible with luminance, the pop-out nature of color contrast is simple to justify. Even when the absolute luminance of two colored patches are perfectly matched, the spectral composition serves as an easy means of distinguishing between two different patches.

Machine vision researchers working in this area have been concerned mostly with classifying the transformation from spectral properties to the color sensation. Attempts to determine spectral-based contours have been mostly influenced by the nature of camera hardware and the algorithms for luminance-based edge detection. Solid-state color video cameras contain the equivalent of the three different cone cells and produce numerical values for the ratio of red, green, and blue wavelengths that strike the photoarray. Most attempts at finding edge information from these color signals concentrate on performing luminance-based detection on the three different signals and then combining them. Simple means of performing this operation can be found in [MS83].

3.2.3 Texture-based Contours

While we all have an intuitive notion of the tactile texture of an object, what does it mean for an object to have a visual texture? Texture can be considered to be the perceptual portion of cognitive pattern recognition; that is, it is the instantaneous, low-level process that finds basic patterns in size, orientation, and other properties without conscious thought (see [Jul75]). For example, the left circle in Figure 3-3 is an example of a texture-based edge; the processing occurs automatically and at a very low-level. The pattern recognition that allows us to recognize the different quadrant in the right circle of Figure 3-3 occurs at a much more conscious level, and must be maintained.

Pure visual texture segmentation is certainly not a central phenomenon in everyday visual experience. Objects can be distinguished through many other methods without relying on this type of pattern detection, and pure instances of objects that can be discriminated only on the basis of texture are rare at best. However, this does not reduce the potential importance of textural processing as part of the visual process. As Bergen and Landy wrote:

> ...the study of pure texture differences (in the absence of differences in brightness, color, depth, or other properties) is analogous to the study of isoluminant color differences, which also are not very common in natural scenes. The relative rarity of isoluminant color discrimination in the real world does not imply that color perception is an unimportant component of seeing. Similarly, the rarity of pure texture differences does not reduce the potential importance of texture perception, especially in the visual processing of complex scenes. [LB91, p. 253]

For contour formation, texture can give an indication of boundary information that other techniques lack. Texture accounts for the patterns in size and orientation across wide areas that other operations ignore.

Computational approaches to texture segregation originated with the investigations of Julesz, who began a mathematical description of textural patterns [Jul75]. Voorhees and Poggio brought a computational classification to these texture patterns by defining basic texture elements, called textons, for gray-scale images [VP88]. There have also been a variety of computational attempts at segmenting pure texture from images. Malik and Perona have used an inhibitory scheme between even-symmetric linear filters to identify textures in grayscale images [MP90]. Bergen and Landy have proposed an alternate method for detecting texture based upon the opponency of oriented filters [LB91].

3.2.4 Disparity-based Contours

While much research has been spent on building edge detectors from single images, biological systems use binocular vision to vastly simplify some edge location tasks. The correlated information from both eyes can result in depth information, which for real-world objects can be used as a strong indication of object segmentation. Light from a surface object will strike each eye at different retinal locations (relative to the fovea), based upon the distance from the observer to the object, the direction of gaze of each eye, and the distance between eyes. The binocular disparity is the difference in locations between the projection of a single point in the world onto the right retina and the projection onto the left retina. Disparity alone can result in depth perception, which was first popularized by the random dot stereograms of Julesz [Jul75]. An example random dot stereogram is shown in Figure 3-5. By crossing your eyes and fusing the two images, depth information can be recovered from the scene.

There are two interesting points that can be drawn from the stereo perceptions of random dot stereograms. First, humans are capable of extracting depth information solely from retinal disparities. While other information may be helpful, retinal disparity alone is sufficient for depth perception. The second interesting point is that the brain is capable of making the difficult correlation between these seemingly random dots. The most difficult part of extracting stereo depth from retinal disparity is determining the correlation between pairs of points on each retina. For the case of

Figure 3-5: Random dot stereograms demonstrating the effect of perceived depth on contour creation. From [Gol89, p. 244].

random dot stereograms, the brain must determine for each pixel in the left image if there is a corresponding point in the right image, and at what disparity the match is located. This process may be easier for real-world objects, since all black spots in the random dot stereogram can match with any other black spot.

Computational approaches to the stereo problem have focused on correlating points between the two stereo images. Brute force methods at correlating pixel values can be expensive, and it is not clear how to determine what the criterion for correlating a pair of pixels should be. A more clever feature-based approach was first developed by Marr and Poggio [PM76], and later expanded by Grimson [Gri81]. In the Marr-Poggio-Grimson algorithm, the stereo images are first processed with the second derivative of a Gaussian function (a type of edge detection). The zero-crossings of these points are then taken as feature elements to be matched between images. As an additional verification to the matching, these points also have associated with them the direction of the slope of the luminance field at that point. Given a pair of stereo images, the Marr-Poggio-Grimson algorithm attempts to find corresponding feature points between the two images and then label each pixel in the original images with a distance from the camera. By first correlating the feature points derived from the luminance edges, local ranges of disparity around those points can be more easily determined. The Marr-Poggio-Grimson algorithm gives a reasonable depth-map of the stereo scene presented, but is often very expensive to compute.

While it seems intuitive to reconstruct all of the depth information from a scene, this may not be necessary for the computations we are interested in performing. There is also evidence that biological systems do not actually compute exact depth reconstructions. In 1970, Richards proposed that the human disparity processing is composed of three different "pools" of neurons, one set that is sensitive to crossed disparities, one set for uncrossed disparities, and one set for near-zero disparities [Ric70]. Similar to the reconstruction that occurs from the three types of cone cells to create color perception, these three pools of disparity-sensitive neurons combine to localize the possible disparities of a scene. This "Pool Hypothesis" had only limited psychophysical basis until Gian Poggio investigated the disparity-sensitive neurons of cortical area V1 and found only three different classes of neurons that responded as predicted by Richards [SW90, p. 323-5]. This type of depth processing may be very useful to artificial systems which do not need exact depth reconstructions, only enough depth for successful interactions with the world. In this case, the correlation problem becomes much simpler, since an algorithm needs only to check if there is a corresponding pixel with the correct attributes at a certain retinal disparity, as opposed to searching some region of the retina for pixels with the correct features.

3.2.5 Motion-based Contours

Even for images that have no well-defined luminance or color differentials, moving patches can create contours. Evolution has given humans an extraordinary ability to detect motion. As early as the late 1880's, Sigmund Exner demonstrated that even when observers could not spatially resolve two adjacent sparks, they could detect the motion if the two sparks were presented at slightly different times [SW90, p. 205]. Since then, psychophysicists have demonstrated that motion also provides a strong sense of continuity. If a subset of dots in a random dot display are moved in the same relative direction and distance while the other dots are moved in a random direction for a random distance, subjects have no difficulty in locating the contiguous patch. Even if only a small percentage of dots from an area are moved together, observers still are capable of linking together those few patches together to form a single moving screen in front of the background.

Computational methods for detecting motion have ranged from the most simple differencing schemes to complex optic flow calculations. The simplest means for detecting motion is to subtract the image intensities at time t from the image at time $t-1$. While this strategy is simple, it has many difficulties. Random noise in the image shows up as moving patches, and a moving object can only be seen as a blur with no sense of direction. At the other end of the spectrum, algorithms called optic flow calculations have been designed to track local groupings of pixels from one image to the succeeding [GGSF59]. Optic flow calculations can be computationally expensive, since local patches need to be compared with all other possible local patches in the second image [HS81]. Many of these problems can be assisted by higher temporal resolution, or by more intelligent matching methods for local patches.

3.3 Other Sources of Contours

The previous section examined some of the physical properties of stimuli that can lead to contour perception. This section will detail some of the other ways in which contours can be created. While many of the physical edge detection methods have been well explored by computational researchers, the other ways of forming contours have been primarily of interest to psychophysicists. Four different sources of contour information will be presented in this section: illusory contours from Gestalt effects, contours from expectations or prior knowledge, contours from filling-in effects, and contours from other sensory information.

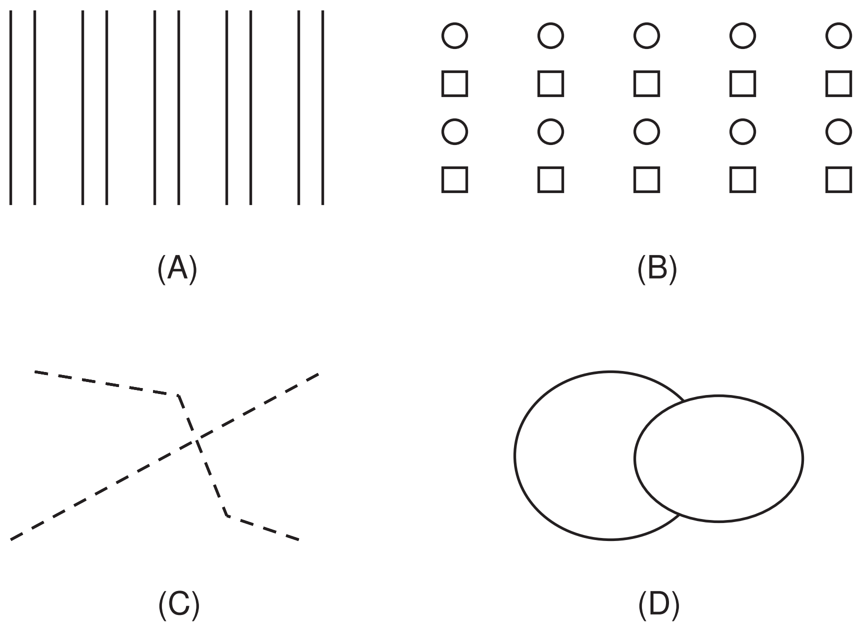

Figure 3-6: Gestalt Principles of Organization: (a) proximity, (b) similarity, (c) good continuation, and (d) closure and good form. Adapted from [And90, p. 67].

3.3.1 Gestalt-based Contours: Illusory Contours

Figures like the Kanizsa square (Figure 3-1) have been of great interest to psychologists because of the perceptual illusions that they create. These effects have been called "illusory contours," "subjective contours," "imaginary figures," and a host of similar names. For the purposes of this research, I will use the most common name for these effects: "illusory contours." This is not an ideal naming system: Contours are perceptual features, and are no more illusory than perceptions of "blue." All contours are subjective, and the boundaries they create are no more imaginary than luminance based contours.

Gestalt psychologists, especially Gaetano Kanizsa, were the first to explore the types of figural configurations that would produce these illusory contours [Kan79]. The Gestalt psychologists developed four principles of organization (see Figure 3-6):

(A) Proximity: Objects that are spatially close will tend to be grouped together.

(B) Similarity: Similar objects tend to be grouped together. Even though the columns are more proximal, the squares and circles are grouped into horizontal

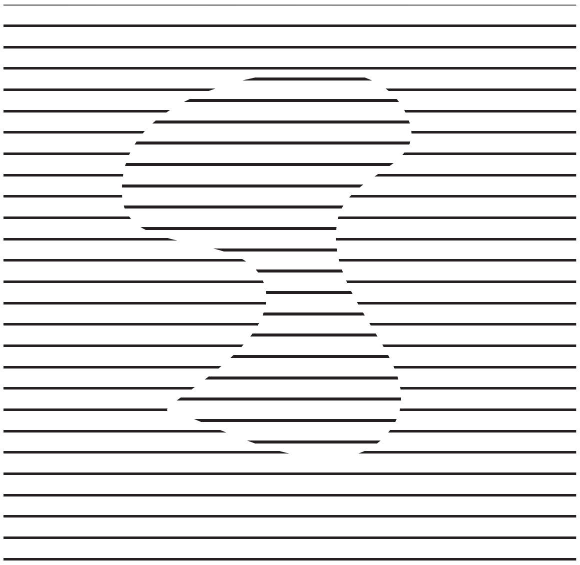

Figure 3-7: Illusory contour created by line endings. This type of contour and the Kanizsa square are illusory contours.

units.

(C) Good Continuation: Observers perceive this figure as two almost straight lines that cross at a single point, instead of two lines with sharp angles that branch either right-left or top-bottom.

(D) Closure and Good Form: Observers tend to view this figure as two ellipses, one occluding the other, instead of an ellipse and a crescent-moon figure.

For our purposes, any contour created by the configuration of stimulus elements that is not represented by a physical property will be classified as an illusory contour. This includes the line-continuation effects of the Kanizsa square as well as effects caused by the ends of lines, like the contour in Figure 3-7. Illusory contours have a pop-out effect identical to that of a luminance or color edge. The perceptions of illusory contours are stable, automatic, and require no attentional upkeep [PM87]. Illusory contours are not modal specific; they can be created from black and white drawings, from random dot stereograms, or from color fields [SW90, p. 289].

Computational attempts at describing illusory contours are much less prevalent than the other modalities we have examined so far. Guy and Medioni have proposed a special point-field operator that can be convolved with a luminance image to produce a "saliency map" showing both luminance-based edges and illusory contours [GM93]. Rudiger von der Heydt et al. have proposed a similar solution using biologically plausible grouping operators that operate over local regions in an intensity image [HvdHK94]. Both of these algorithms are slightly expensive computationally, but accurately reveal simple illusory contours.

3.3.2 Contours from Expectations and Prior Knowledge

The formation of contours can also be very effectively influenced by prior knowledge and expectations. The case for prior knowledge is simple; once you have determined the outline of the dog in Figure 3-2, the contour is immediately obvious on a second viewing. By allowing prior knowledge as part of our contour formation scheme, we also allow for specialized learning to become an integral part of the process for contour identification. The influence of expectations is slightly more complicated. If you are instructed to turn the page and look at the pictures of the young lady at the top of the next page (Figure 3-9) and then quickly return to this point in the text, you will have little difficulty in filling in the details of the woman's chin, eyes, etc. However, if you are instructed to turn the page and look at the figure of the old crone on the next page, the stimulus will be slightly altered in your perceptions. The image in Figure 3-9-A is most easily interpreted as a young woman, looking over her shoulder away from the viewer. The image in Figure 3-9-C has been slightly modified, and can often be seen as an old woman huddled in her coat. (The young woman's chin corresponds to the tip of the nose in the old crone.) The central image in Figure 3-9-B is ambiguous, and will often be seen as whichever age had been previously viewed. While this priming example gives only a very brief, non-stable look at the influences of expectation, it does lend some weight to the idea that expectation can effect perception.

While computational approaches to vision have not yet begun to include high-level expectations, there are some conclusions that can be drawn from these influences. Our visual systems are not designed to operate on single images, like a signal processor.

Figure 3-8: Blind spot phenomena. For viewing instructions, see text. Adapted from [Pin90].

They operate on an almost continuous stream of information that remains almost stable for a large portion of the time. This allows the visual system to make use of time or energy spent on prior computations in refining new visual inputs. This type of stream-oriented approach is very different from the perfect slate approach of blindly processing each image individually. Our computational models must begin to account for this type of stream-based processing architecture in order to achieve reliable, real-time results. This topic will be discussed again in section 4.5 when we begin to formulate a model of contour perception.

3.3.3 Contours from Filling-in Effects

For the amusement of party guests, the 18th century European aristocrats are said to have caused the moon to disappear by carefully aligning its image with the blind spot. While demonstrations of this sort are now interesting sidenotes to introductory psychology classes, the filling-in effects of the blind spot are still very poorly understood. The human visual system does more than simply cover over the blind spot (or any other scotoma) with a blank "background" patch; it actively seeks to fill in the missing information with the most consistent choices. For example, if you close your left eye while staring at the letter "A" in Figure 3-8, you can observe your own blind spot. By adjusting the distance between the page and your eye, you can make the solid circular disk at the right disappear. Once the stimulus from the solid circle falls onto the optic disc, the visual system fills in the missing information with what it

A B C

Figure 3-9: An ambiguous figure. Image (A) is often seen as a young woman, looking over her shoulder away from the viewer. Image (C) is that of an old woman huddled in her coat. Image (B) is ambiguous between the two. See text for viewing instructions. From [Gle91, p. 221]

expects, that is, more blank paper. However, if you then close your left eye, stare at the letter "B" in Figure 3-8, and adjust the distance between your eye and the page, a different filling-in effect occurs. Instead of simply leaving the missing gap in the bar stimulus, the visual system joins the two halves of the bar into one continuous piece. The visual system creates the contours of the bar through the area of the blind spot and fills in the area with a black color.

The psychophysics of these filled-in contours are not well studied, although they seem to behave according to the Gestalt grouping principles. It may even be possible that the illusory contour figures are side-effects of the neural circuitry designed to compensate for the retinal blind spot. Little is understood about the means by which the visual system chooses to provide the missing information from blind areas, but this type of processing does allow us an interesting, and unexpected, look inside the black box of the visual system.

3.3.4 Contours from other Sensory Information

While contours are a visual phenomenon, the perception of contours can be influenced by many other sensory stimuli. Ramachandran has shown the influence of audition on visual motion detection (see section 2.2.2), but the psychophysics of creating contours through senses other than vision is poorly understood. However, influences from tactile sensations, audition, and other senses may provide both verification and a type of prediction for contour information. By palpating an unseen object, I can both form expectations of the types of contours that it contains and later verify the contours that the visual system reports. Similarly, as I enter a dark room and tap about with my cane, the force feedback from the cane striking objects and the sound of the impacts are likely to verify my visual perceptions and to enhance my visual expectations. Although contours are visual phenomena, the influences of other senses should be made salient in our models of contour perception.

3.4 Are these the Right Modules?

While these methods for contour formation are by no means a canonical listing, we must entertain the question: Are these the right modules, at the right abstraction level, for the job that we are interested in? Have we found the right level to look at this problem, and have we broken the problem down into the right pieces? By separating the problem of contour identification from other visual tasks, we can provide higher-level systems with a useful module for object segmentation, figure-ground separation, or object tracking. A module that could reliably deliver contour information from a scene would be extremely useful for navigation, identification, and manipulation tasks. Therefore, examining this piece of vision seems to be a reasonable proposal.

What then of the individual pieces that compose contour identification? Psychophysics indicates that each of the physically based modalities (luminance, spectral composition, texture, retinal disparity, and motion) operates as an independent, low-level, automatic visual process. By the addition of influences from expectation, prior knowledge, other sensory information, illusory contours, and filling-in phenomena, we allow a way for high-level information to influence the low-level functions. Finally, by relying on a variety of methods, we provide for robustness while allowing for diversity in the range of stimuli that we can successfully process. While these may not be all of the subcomponents of contour identification, they provide good examples of the kinds of pieces that are certainly involved.

Now that we have a more rigorous and detailed description of the types of stimuli that we want our system to process, we can begin to construct a model of the ways in which these various components can interact. The next chapter will detail the theoretical model of interactions between the edge-detection modules and the means by which they can be integrated to form high-level contours. Chapter 6 will detail the subset of this model that was implemented for this thesis.

Chapter 4

The Integrated Contour Model

Armed with the more detailed description of contours developed in the preceding chapter, we can begin to build a theoretical computational model that challenges the classical methods of direct computation from the "perfect slate." This chapter presents a model of contour processing that allows for the integration of independent low-level edge detection modules (such as luminance and texture) with other higher-level sources of information (such as Gestalt effects and expectations). For the overview of this model, only a simplified view of the computational pathways will be presented. The exact specifications used in our implementation of a subset of this model will be covered in Chapter 6.

The central idea behind the integrated contour model is the proposition that complex, high-level perceptions can be created by using simple compositions of low-level routines coupled with input from other high-level processes. Our model of perceptual processing will be constructed out of many interacting modalities.

4.1 Overview of the Model

The basic model that will be presented in this chapter consists of multiple low level physical edge detectors coupled with information from high level sources to form a single consolidated contour image. The model itself is relatively simple: allow for each of the low-level edge detectors to operate independently, accept information from

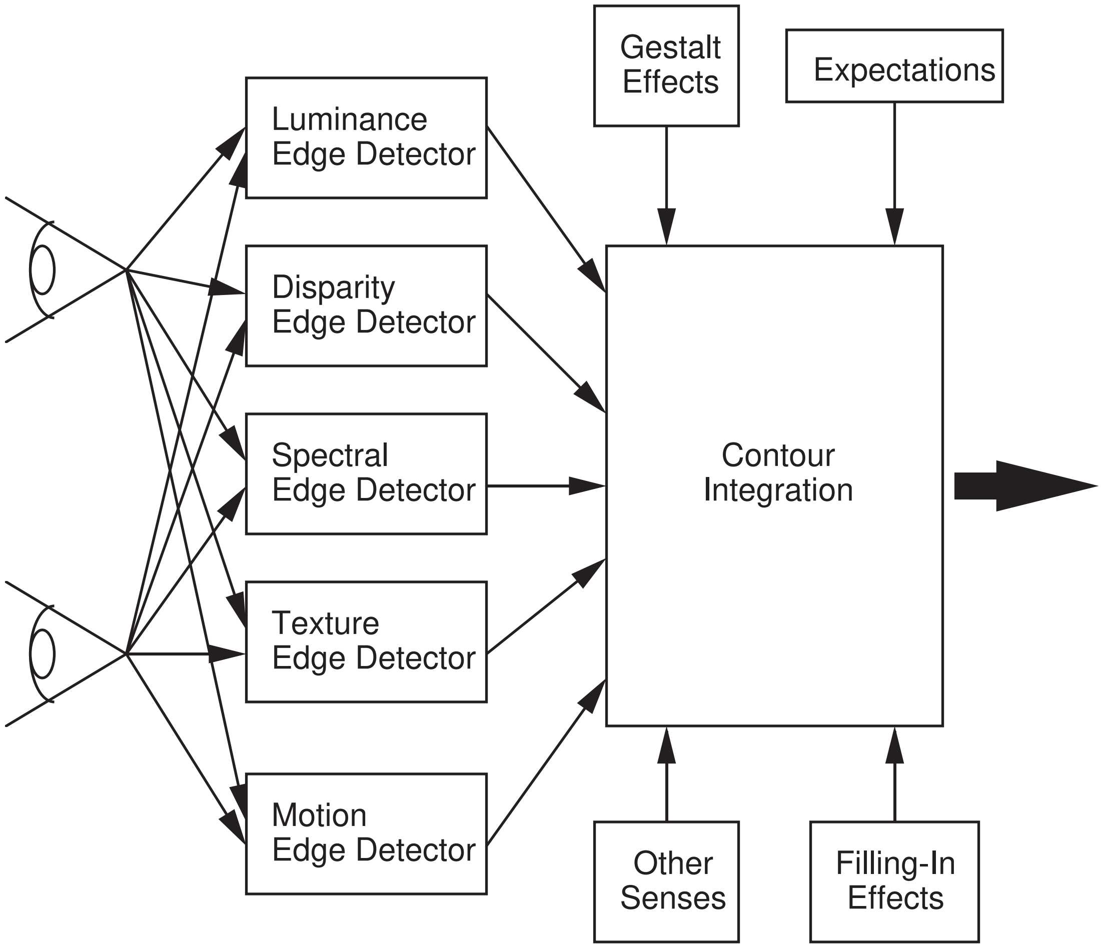

Figure 4-1: The Integrated Contour Model. Contours are high-level perceptual features that are formed by integrating many low-level physically-based edge detectors with the results from other high-level processes. This figure shows some of the types of edge detectors and other processes that may contribute to contour perception. The feedback loops that allow for the results of the integration box to be accepted as inputs to each sub-module are not shown.

higher level processes, and combine all of the information at a single integration point. Figure 4-1 shows the basic pathways that result in an integrated contour. In addition to the pathways shown in Figure 4-1, the integrated contour model also allows for the results from the integration to be passed back to each of the sub-modules. In this way, building a contour is an iterative process that can rely on multiple stages of processing; that is, the integration box does not perform one-shot processing on an input image, but rather allows for successive refinements in the contour image produced.

The rest of this chapter explains the implications of this simple model. Aspects of making the physical modalities independent processing components are discussed, along with the methods of inclusion for higher level information. The integration techniques are then examined from a mathematical and a practical standpoint. Finally, the complexities introduced by the addition of feedback are reviewed.

4.2 Low-Level Physical Modalities

In the model of Figure 4-1, each of the physical characteristics of an image is processed by an independent module. Luminance-based edges, spectral-based edges, motion-based edges, disparity-based edges, and texture-based edges are all computed independently, and in parallel, by low-level procedures. This type of processing matches the psychophysical results for these types of contours. The processing must occur at a very low level, since the appearance of contours from these modalities is almost instantaneous and automatic. Additionally, these modalities will be modeled as separate processing elements since you can easily distinguish between contours derived from luminance and from disparity.

To illustrate the functioning of this portion of the model, consider the box that deals with luminance-based edges. The inputs to this processing stage are the raw photoreceptor values from both retinas, and the output is a description of where in the image there is a high gradient of luminance. To account for noise and uncertainty, the result of this processing stage can be viewed as the probability of a luminance-based edge located at each pixel, normalized over the entire image. This information is then passed on to the integration box where it is combined with further information to produce a first pass at integrated contours. As is discussed below, the results of the integration can optionally be passed back to the luminance module as feedback, but this is not part of our current implementation.

4.3 Interactions with High-Level Processes

In the previous chapter, Gestalt effects and filling-in effects provided strong indications that contours can be created from sources that do not rely directly on the low-level physical image properties. Gestalt effects, filling-in effects, expectations, and other senses must be given some type of influence on our final integrated contours. In the model of figure 4-1, each of these modalities contributes to the integration in the same way that a physical modality would. We need not be concerned with having the higher level modules directly influence each individual physical edge-detection operation, since the iterative nature of the model will allow any given sub-module to influence any other sub-module (see section 4.5 below).

Each of the higher level modules takes input either from the raw retinal images or from an internal state and produces output suitable for use by the integration box. For example, the module that computes Gestalt effects takes as input the raw retinal images and produces a description of where in the image Gestalt effects predict contours. On the other hand, the expectations module maintains the state of the previous image, accesses visual memories, and produces a likely result. In either case, the output of these boxes is processed directly by the integration box.

4.4 Method of Integration

Until now, our model of contour processing had been quite simple: just allow for each modality to contribute an image to some centralized location that was responsible for putting together all of these pieces. In its most general form, this integration box combines the two-dimensional images from n different sources into an $n + 2$ dimensional output "contour" space. This "contour" space has two dimensions for the pixel locations and n dimensions for the individual probabilities from each of the contributing submodules. In this form, the output of the integration box is simply a conglomeration of the individual inputs. However, this type of integration does not simplify our higher-level goals like object segmentation or tracking. What is needed is a means for combining these different modalities together into a form that can be easily processed by many other systems.

In order to begin combining these modalities together, it is necessary to place some restrictions on the types of inputs that the integration box will accept. First, we restrict the inputs to remain in retinal coordinate frames. This gives us a simple means of identifying common spatial points among the images. Second, we restrict the inputs to the integration box to be probabilistic estimates of the likelihood of an edge at a given spatial location. This allows us to compare data from various modalities, since each is now simply a probability. For example, the output from the luminance-based edge detector would be an image in retinal coordinates of the probabilities of luminance edges at each pixel. Texture-based edge detectors would produce images in retinal coordinates that give the probability of texture edges at those coordinates, etc. These two simple restrictions allow us to compare the data across modalities in a simple and unified manner. Additional dimensions could be added (for example, not only the probability of an edge, but also the orientation) in a similar manner. Because it is not clear that all of our modalities can produce orientation information as well as probability information, we will consider only the probabilities.

Given the probabilities from each of the sub-modules, how are these results combined? Since each input is a probabilistic mapping over the same coordinate space, any of these tasks can be described as a weighted average of the inputs. By assigning weights to each of the inputs, we determine the relative importance of each modality. Is texture a more important indicator of contour than depth? Are expectations as important as luminance edges? There is no single set of answers to these questions; the relative importance of each of these modalities depends upon the type of task to be performed. For example, if the task is to trace the outline of a solid object, Gestalt edges are of little importance compared to luminance and depth. If the task is to visually track an object, the effects of motion edges may be more important than influences from other senses. If the task is to find the coffee mug in a scene, expectations might be the most important modality. Task dependency in the integration method also allows for completion of tasks like finding only spectral-based edges, or only disparity-based edges.

For the remainder of this thesis, the task that we will be concerned with is object segmentation. This task was chosen because each of the input modalities are of equal importance, allowing the effectiveness of combining various edge detection techniques to be demonstrated. By refining our task even further, we could achieve better end results, but allowing each technique to have an equal weight in the final result will allow us to demonstrate the areas in which each technique excels and the areas in which the combination of techniques is more effective than any of the individuals. For more refined task specifications, the weightings of the modalities could be subject to a learning algorithm.

4.5 Loops in the Model

In addition to the feed-forward capabilities described above, we would like the model to allow for some influences between the modalities. Each low-level modality is primarily independent, but there are cases where modalities can influence each other. For example, Gestalt effects sometimes create changes in the perceived brightness [PM87]. The center of the Kanizsa square is often perceived to be slightly lighter than the surrounding background. It is not clear whether this influence is only upon the high-level processing that leads to the concept of brightness, or whether this is actually an influence on the processing stages that deal with luminance. To allow for the possibility of this low-level interaction, we will add feedback loops in our contour perception model. The results of the integration box will be provided as inputs to each of the contributing modalities.

The use of these feedback loops allows for additional simplicity if the system is designed to process images continuously, instead of processing each image as a stand-alone stage. Each module can use the results of the integration step from the previous iteration to refine the probability estimates for the following iteration. Stable visual stimuli can be more readily processed, since the results from the previous loop can be used as an expectation of the current results. In a mostly stable visual world, this allows the individual processing modules to concentrate on changes in the visual scene. The system that we will implement in Chapter 6 does not yet take advantage of this feature.

4.6 Evaluation of the Integrated Contour Model

The model that has been presented in this chapter is a starting point for our explorations of visual cognition. It allows for rapid, parallel computation of low-level physical image features while providing guiding input from higher-level processing stages. The model also accounts for many of the "anomalies" of contour processing that were described in the preceding chapter. While this model should prove adequate for our computational tasks, it is currently missing two important features: a generalization to other cognitive tasks and a description of the interactions between the submodules.

Models of visual processing are more useful if they apply not only to single visual problems, but also to generic visual tasks and other cognitive processes. It is not clear how to generalize this model to other high-level cognitive tasks. How can this model be applied to brightness perception, or the perception of depth? It is conceivable that the same low-level modalities are at work, but what are the high-level influences and how is the integration performed? It is also not clear that this model is useful in evaluating non-visual tasks. For example, it is difficult to apply this model directly to the problem of memory retrieval. It is unclear how to divide memory retrieval into subtasks, or how the integration should occur. The model that has been presented here is not meant as a generalized theory for cognitive tasks. However, there are some conclusions from this work that are applicable to other tasks. The evolutionary viability of a processing stage that utilizes many different overlapping strategies is extremely good. By having many subsystems that compute different, but related, aspects of the same problem basic fault tolerance and error detection can be implemented. For memory retrieval, it would be beneficial to store memories in many different formats. For example, a pictorial version of an event coupled with language and temporal versions would promote fault tolerance.

The model also does not specify the exact interactions between the submodules. The presence of the feedback loops allows for many interactions between the modules, since each submodule (such as depth, color, or Gestalt effects) can have an influence on the other modules. The psychophysics of the interactions between these different modalities is sufficient to show that these influences exist, but there has been little work to classify all of them. However, as we shall see in chapter 7, one pass through this model will be sufficient for showing its usefulness for the task of object segmentation. For more complex tasks, these iterative loops may be necessary, but we will concentrate on a single pass through the algorithm in this work.

With this basic model of contour processing, we can begin to build a system to test these ideas. The next two chapters are concerned with the hardware and software implementations of a system that embodies the design principles outlined in Chapter 2. Chapter 5 outlines the hardware specifications of a system that attempts to avoid the standard "perfect slate" assumption of computer vision architectures. Chapter 6 then outlines a software implementation of a subset of the model of contour processing shown in this chapter.

Chapter 5

Charlotte: An Active Vision Platform

In chapter 2, the many problems of viewing vision as a "perfect slate" were discussed. This chapter continues that discussion with the implementation of an active vision platform that circumvents some of these difficulties. A foveated, binocular, active vision platform linked to a C40 digital signal processing board was constructed with the goal of avoiding the "perfect slate" assumptions. The details of the physical characteristics of the camera platforms, the motion capabilities, and the processing power will be provided in this chapter.

A series of three active vision heads were built at the MIT Artificial Intelligence lab for various research into active vision. The three-heads project was an outgrowth of the Cog project, which aims at building an upper-torso humanoid robot [BS93]. The robots were designed with assistance from Mike Wessler and Sajit Rao, and were constructed with the assistance of two undergraduate students: Matt Scheuring, who constructed the aluminum support frame, and Diego Syrowicz, who populated many of the circuit boards. The second of these robot heads, called Charlotte, is shown in Figure 5-1. Charlotte currently is used as a platform for the contour integration model that was described in the previous chapter, but the design of the robot was not motivated by this singular goal. Instead, Charlotte was designed to be a generalized platform for a variety of machine vision research. The remainder of this chapter de-

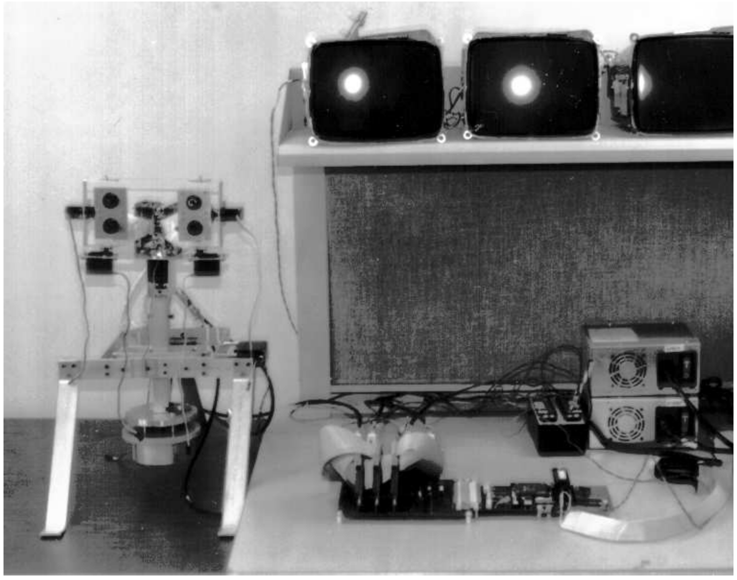

Figure 5-1: Photograph of Charlotte and the supporting hardware.

scribes the design decisions and implementation of the visual, motion, and processing capabilities of a general active vision system.

5.1 Visual Capabilities

In our efforts to model the anomalies of human visual processing, a camera system that models some aspects of the human visual system was introduced. In designing the system, it became clear that for this project three capabilities in particular were important to build in hardware: binocular vision, foveated vision, and active vision. These three capabilities are not sufficient for accounting for the entire range of anomalies that a complex system like the human visual system contains, but they provide a basic set of the issues that must be addressed in hardware. Other anomalies, like the blind spot, can be modeled in software without involving specialized hardware. Two additional considerations in the design were price and availability of components. For this reason, more accurate representations of the human retina (see, e.g., [YS87]) were discounted.

The first design goal of the camera system was to allow for binocular vision. While processing single images can provide interesting insights, the problems and simplifications that a stereo camera brings are exceedingly important. For example, the additional information provided by a stereo image requires us to explore the problems of integrating multiple overlapping images. Stereo processing also allows for simplifications of many normal visual tasks. For example, figure/ground segmentation can be roughly approximated with stereo by simply grouping pixels together based on disparity.

Another anomaly of visual processing that Charlotte incorporates is the non-uniformity of photoreceptor density. The presence of a densely packed fovea has a great impact on normal human visual processing. To discriminate among letters or other fine details, you must focus the direction of your gaze on an object in order to bring the object's retinal projection onto the fovea. While the technology for digital cameras with non-regular pixel grids is advancing rapidly, it is not currently cost-effective to implement systems that completely model the densities of photoreceptors in the human retina [YS87]. However, ignoring this important anomaly leads to over-simplified visual systems. As a compromise, Charlotte was given two cameras for each eye; one camera with a 4mm wide-angle lens, and the other with an 11mm narrow-field lens. This camera configuration allows us to process both a wide-field view and a foveated section of the image. To simplify the geometric relationship between the cameras, the cameras were mounted with the narrow field lens directly above the wide angle lens, with a separation of approximately 1 inch.

The third design goal for Charlotte's visual system was to allow for moving cameras. The advantages and complexities introduced with an active vision system have been studied by many different researchers (for a review, see [BY92]). Moving cameras create many problems for image processing through blurring of images and sampling problems. However, moving cameras also allow for a greater range of life-like behavior and simplify some computational issues. For example, suppose that you are interested in identifying faces. Performing object identification over the entire pixel range is very costly, as is having high resolution capabilities over a wide angle. However, with a narrow-field high resolution camera and motion capabilities it is possible to use a wide-angle low resolution camera to find motion or pattern match to find possible faces, move the cameras to foveate on the potential match, and then perform a single high-resolution computation. In this way, intensive computation is only carried out on a limited foveal area while still allowing for wide angle detection of interesting objects. As a general platform for vision research, Charlotte was designed to have a wide range of motion with relatively quick response time. The following section describes the motion capabilities of this active vision platform in greater detail.

For issues of price, compactness, and availability, Chinon CX-062 color micro-cameras were chosen as the standard for the active vision heads. The Chinon cameras consist of a 1 square inch circuit board containing the photosensitive circuitry and lens attached to a 2 inch by 4 inch board used to maintain power and produce standard NTSC video output. The Chinon cameras proved to be ideal for our design, since the split boards allowed for very close mounting of the lenses while reducing the amount of total weight that needed to be moved to change the direction of gaze.

5.2 Motion Capabilities

In building motion capabilities for Charlotte, the primary design goals were to allow for rapid eye movements that had a human-like range of motion while maintaining a simple interface. To simplify the design, standard model airplane servos were used. Servos allow a simple command interface, and need not be monitored as closely as standard shaft-encoder motors. Servos do not have the fine degree of resolution that other motors provide, but they are much simpler to install and control.

Charlotte was built with seven degrees of freedom: one degree of pan for each eye, one degree of tilt for each eye, and three degrees of freedom for the neck (see Figure 5-2). Each eye was given an independent degree of freedom for pan and for tilt to allow for binocular vergence and for vertical corrections. The three degrees of freedom in the neck were not absolutely necessary for allowing most visual behaviors, but do begin to give the added complexity of integrating other body movements with eye motion. Since the three active vision heads are an outgrowth of the Cog project, the problems of integrating various body motions are of great interest.

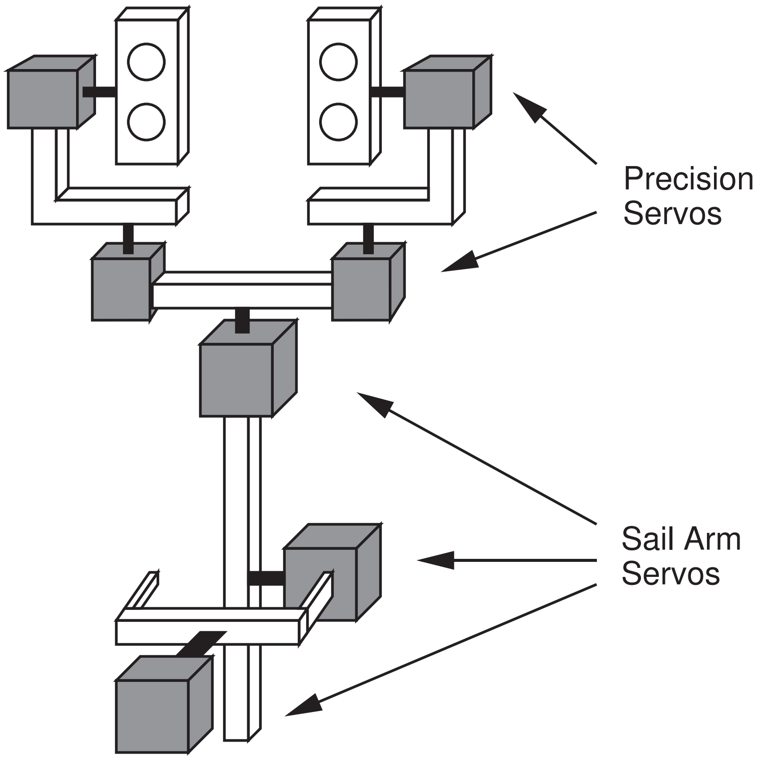

Figure 5-2: Servo locations on Charlotte. Two degrees of freedom for each eye and a three degree of freedom neck allows for a wide range of human-like motion. Counter-weights and connectors are not shown.

Two different servos were used in the construction of Charlotte. The first type of servo was chosen to be light-weight and accurate for the positioning of the eyes. This kept the overall weight of the head low while providing reasonably good image stability. The second type of servo was chosen to have higher power and slightly slower response time for neck positioning. For price and availability concerns, Futaba S148 low-profile precision servos were chosen for eye positioning and Futaba S3801 metal gear sail-arm servos were chosen for the three degrees of freedom in the neck. The positions of each of these servos is shown in Figure 5-2.

5.3 Processing Capabilities

The processing capabilities for each active vision head revolve around a Texas Instruments C40 Digital Signal Processor. The C40 is ideal for image processing tasks requiring a large data bus and high speed array operations. Each active vision head is equipped with a circuit board designed by Gideon Stein that supports the C40 chip with serial line interfaces, a 128 kilobyte local SRAM, an external 32 bit data bus, and 128k of EEPROM. The serial line interfaces are used to connect to a servo controller board and to a Macintosh computer, which serves purely as a file server. The local SRAM contains both C40 program and data structures, while the EEPROM contains the shell routines loaded on reset. The external data bus is used as an interface to the visual processing boards, as explained below. The C40 boards designed by Stein also allow for use of the C40 communication ports as interfaces between boards, allowing multiple boards to be chained together for parallel computations.

Linking the C40 board to a visual processing system requires a number of supporting structures (see Figure 5-3). To simplify the construction of the interfaces, many of the designs used by the Cog project were applied to the active vision heads. Since the heads were also meant to be development platforms for Cog, the similarities in hardware allow for rapid adaption of code between the two systems. This decision aided in construction and debugging, thanks to the help of the many local experts, but was not without its difficulties. The hardware currently used by the Cog project

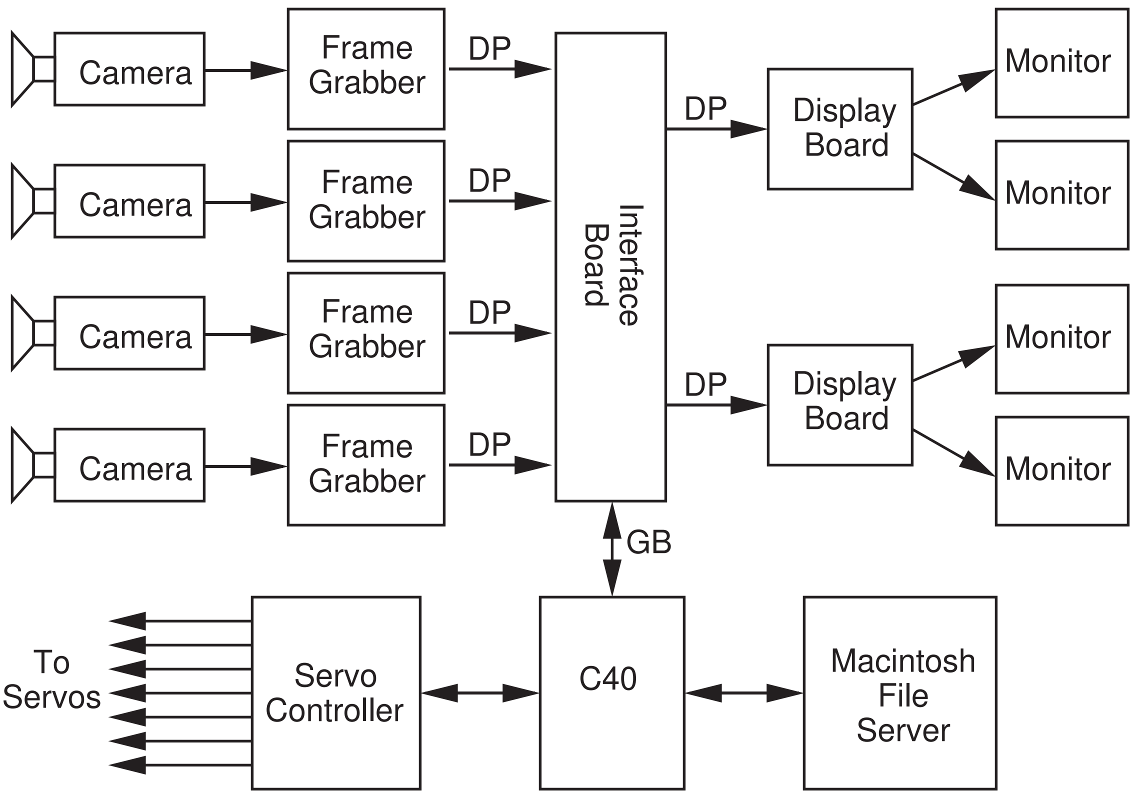

Figure 5-3: Visual processing hardware on Charlotte. Each camera has a dedicated frame grabber which connects to the C40 interface board through a dual-ported RAM ("DP" in the diagram). The interface board is also attached via dual-port RAM to a pair of NTSC monitor boards that convert pixel arrays into NTSC video output. The interface board plugs into the global bus ("GB") of the C40, which uses serial lines to communicate to a servo controller and a Macintosh file server.

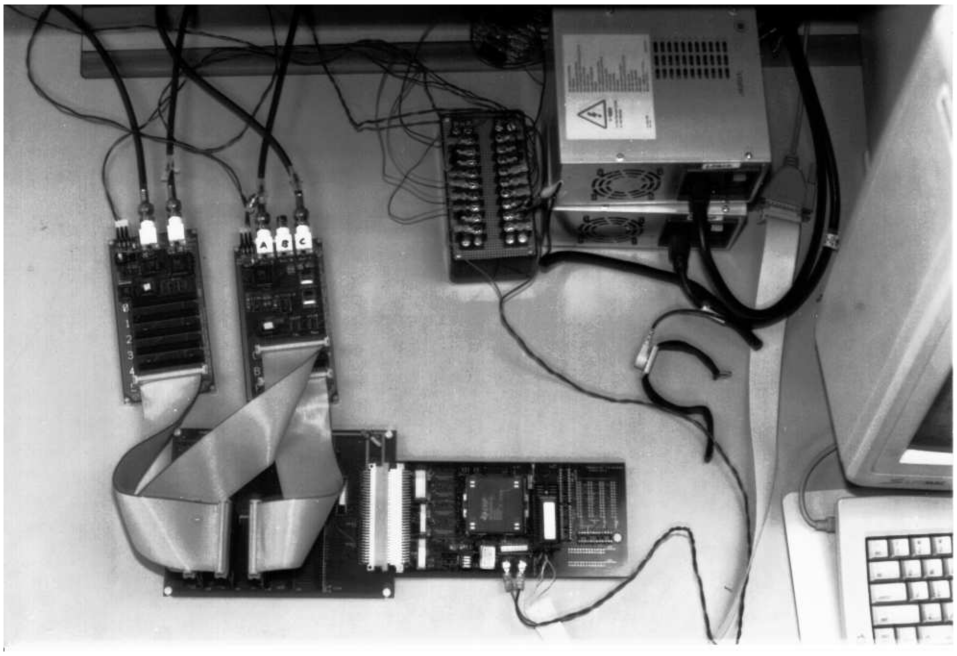

Figure 5-4: Photograph of the supporting hardware for one camera. A single frame grabber (top left) and display board (to the right of the frame grabber) connect to the C40 board (lower right) through dual-port RAMs attached to the interface board.

is limited to 128 $\times$ 128 resolution images of 8-bit grayscale. However, this limit can be surpassed later through hardware redesign or by using commercially available components.

The camera signals produced by the Chinon cameras are standard NTSC analog signals. To convert to a digital output, custom-built frame grabbers designed by Matt Marjanovic of the Cog group are used. Each frame grabber accepts a single NTSC input, subsamples the input to 128 $\times$ 128 resolution and writes out both the digitized signal and an NTSC signal of the digitized image. The digitized signal is written out to a 128k dual-ported RAM card that serves as the standard connector between all Cog hardware. The digitized image signal is sent to an interface board designed by Robert Irie that accepts 9 dual-ports as input-output slots, decodes which dual-port is sending data, and relays this data onto the 32 bit global bus on the C40 board. The interface board also allows for data to be written from the C40 board to a dual-ported RAM that connects to an NTSC display board. The display board, designed by Cynthia Ferrell and Matt Marjanovic, converts the digitized image into an NTSC signal which can then be sent directly to a standard television monitor. Unlike the frame grabbers, each NTSC display board can simultaneous translate three different images. This allows additional results or intermediate processing stages to be displayed without massive amounts of hardware. Finally, a single servo controller board, designed by Chris Barnhart, is used to simplify motor commands and relieve the C40 from the continuous requirements of servo control. The entire configuration, as shown in Figure 5-3, requires one frame grabber for each camera (for a total of four), one interface board, one C40 DSP board, one servo controller board, and two or more display boards. The number of display boards can be increased to allow for more intermediary results or messages to be displayed. A photograph of the hardware setup for a single camera (one frame grabber and one display board) is shown in Figure 5-4.

With these capabilities, Charlotte provides a unique platform to test novel visual architectures that are not possible with a "perfect slate" camera. Charlotte provides a visual system that provides a richer array of information than the standard "perfect slate" setup. While not all of the features of Charlotte will be important to each approach, and while some approaches will require features not presented here, these heads represent a new emphasis in computer vision. By focusing on the many anomalies of biological visual systems, we can construct systems that are capable of a wider range of interesting behaviors and that allow novel solutions to standard problems. In the following chapter, a suite of software for contour integration that was implemented on Charlotte will be presented.

Chapter 6

Implementation of Integrated Contours

A basic implementation of a subset of the model of contour processing developed in Chapter 4 was implemented on the active vision platform described in Chapter 5. This chapter describes the subset of the integrated contour model used in the software implementation, and the details of the algorithms used for the low-level edge detection modules.

Because the entire integrated contour model is too large of a task for a Masters thesis, the first decision in building an implementation of the integrated contour model is which modules to construct. There should be enough modalities to show the strength of combining different types of information, but not too many so that the important points are obscured in volumes of data. The physical limitations of the active vision platform prevent using color as a module, and with no higher level processing both expectations and other senses are not viable options. Motion can also be ruled out for a first implementation, since it is often very difficult to present the relevant data in a static format. Of the remaining modalities, we would like to include both standard forms of edge detection and forms that include some of the anomalous results described in Chapter 3. For this implementation, luminance edge detection, texture edge detection, and Gestalt effects were chosen as the modules. Luminance edge detection is simple, reliable, and gives results that can most easily be compared

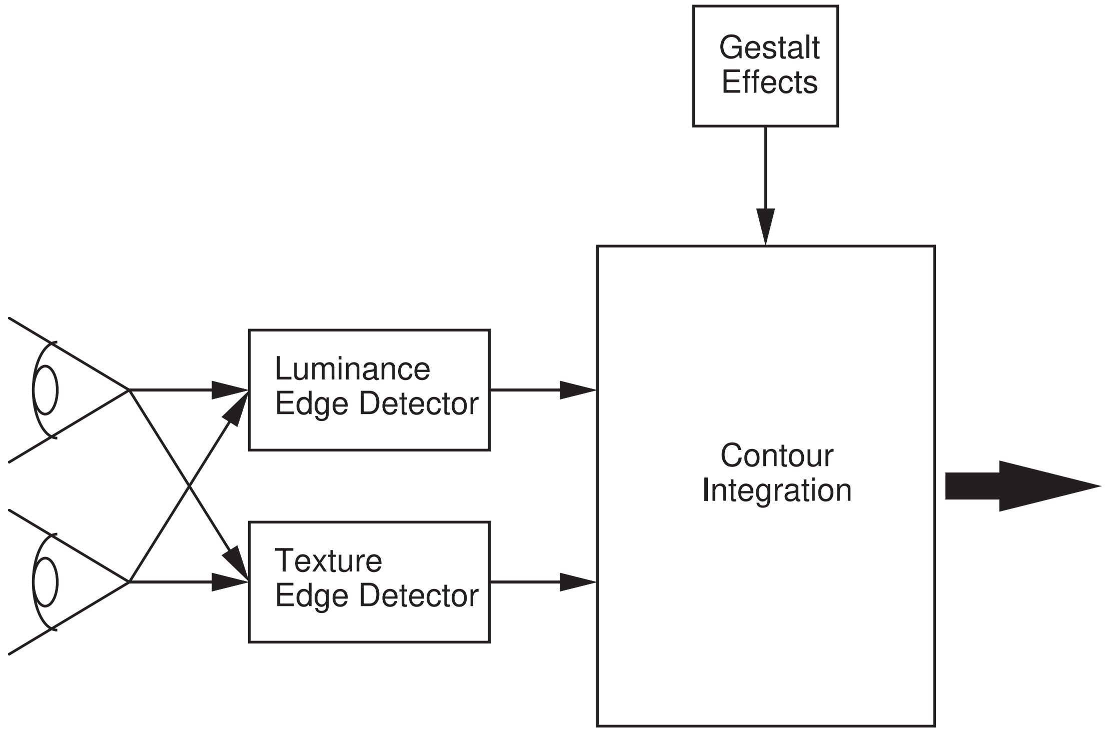

Figure 6-1: Subset of the integrated contour model that was implemented in this work.

with other vision systems. Texture edge detection was chosen since it detects information that is very different from luminance, that is, patterns of intensities across much larger scales. Gestalt effects was chosen as a module both because it has gained some attention by the computer vision community and because it serves as a good example of contour formation that is not based directly on physical information. It certainly would have been plausible to chose either disparity-based edge detection or filling-in effects as modules for this implementation, and it will be interesting to see what results these two modules bring to the current results. The subset of modules developed in this chapter is shown in Figure 6-1.

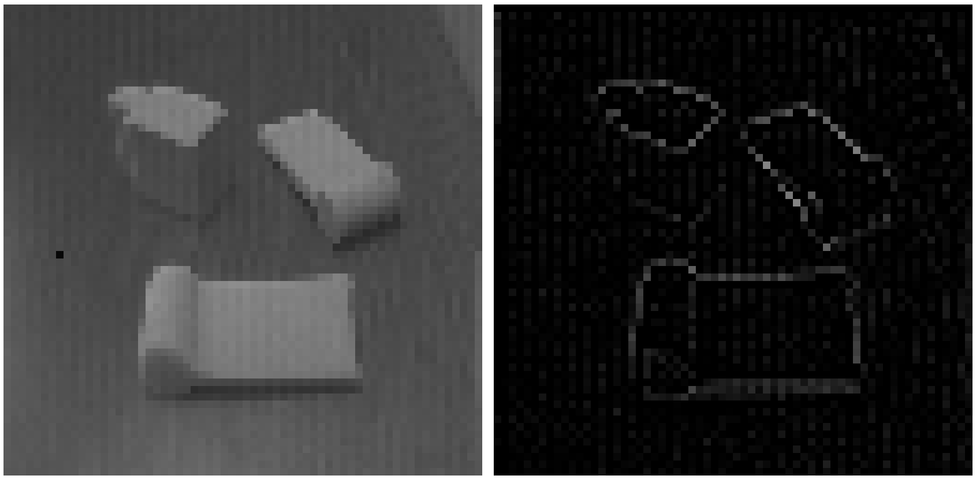

Figure 6-2: Luminance edge detection using a Laplacian filter. The original image of three foam blocks (left) is convolved with a 3×3 Laplacian filter shown in equation 6.1 to produce the image at right.

6.1 Luminance Edge Detection: The Sobel Operator

One of the major strengths of the integrated contour model is that each module can be composed of very simple operators, since special cases can usually be detected by another modality. For the case of luminance edge detection, we can rely on simple filtering to determine the luminance gradients. The luminance operator should include both horizontal and vertical edge detection, provide reasonably stable results, and have a very limited filter size.

The first luminance edge detector that was tested was a simple Laplacian filter of the form:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{6.1}$$

The results of applying this filter to a standard test image are shown in Figure 6-2. Unfortunately, the results of this filter are very noise-sensitive since both horizontal and vertical components are combined into a single filter.

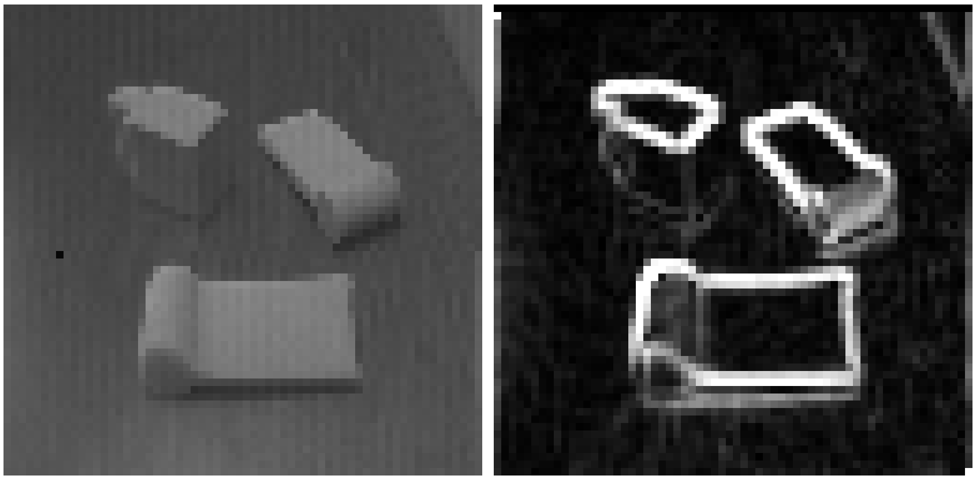

Figure 6-3: Luminance edge detection using a Sobel operator. The original image (left) is convolved with a pair of 3 $\times$ 3 Sobel filters shown in equation 6.2. The absolute values of these results are summed to produce the image at right.

A more stable luminance edge detection scheme can be implemented using a modification on the Sobel operator [Sob70]. The input image is convolved with two different filters, one designed for vertical edges and the other for horizontal edges:

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{6.2}$$

If the absolute values from the results of these two convolutions are summed, the result gives a much more stable estimation of the luminance gradient. Figure 6-3 shows the result of applying this operation to the original image from Figure 6-2. The output from the Sobel operator gives a sharper definition of the luminance gradients than the Laplacian filter does, and also produces fewer spurious results than the Laplacian (as can be seen by the amount of low-intensity background "noise" in the result images). In the final implementation of this system, the Sobel operator was used.

6.2 Texture Edge Detection: Opponent Filters

The technique that we will use for texture detection is based upon the work of Bergen and Landy [LB91]. The original version of the Bergen and Landy algorithm begins by convolving the input image with a large number of oriented local filters. Each filter must also have an opponent filter, which is the same filter rotated by ninety degrees. In our implementation, we will use one pair of oriented filters for vertical and horizontal patterns:

$$\begin{bmatrix} 0 & -1 & 2 & -1 & 0 \\ 0 & -4 & 8 & -4 & 0 \\ 0 & -6 & 12 & -6 & 0 \\ 0 & -4 & 8 & -4 & 0 \\ 0 & -1 & 2 & -1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -1 & -4 & -6 & -4 & -1 \\ 2 & 8 & 12 & 8 & 2 \\ -1 & -4 & -6 & -4 & -1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6.3}$$

and a second pair of filters for diagonal patterns used by Bergen and Landy:

$$\begin{bmatrix} -.163 & -.075 & -3.610 & 1.112 & 1.700 \\ -.075 & 5.662 & -.047 & -7.612 & 1.112 \\ -3.610 & -.047 & 11.301 & -.047 & -3.610 \\ 1.112 & -7.612 & -.047 & 5.662 & -.075 \\ 1.700 & 1.112 & -3.610 & -.075 & -.163 \end{bmatrix} \tag{6.4}$$

$$\begin{bmatrix} 1.700 & 1.112 & -3.610 & -.075 & -.163 \\ 1.112 & -7.612 & -.047 & 5.662 & -.075 \\ -3.610 & -.047 & 11.301 & -.047 & -3.610 \\ -.075 & 5.662 & -.047 & -7.612 & 1.112 \\ -.163 & -.075 & -3.610 & 1.112 & 1.700 \end{bmatrix} \tag{6.5}$$

The results of these images are similar to Laplacian edge detectors oriented towards lines of the corresponding direction. At this point, we diverge slightly from the standard Bergen and Landy algorithm. Bergen and Landy square the results of each filtered image and then collect local spatial information by downsizing each image

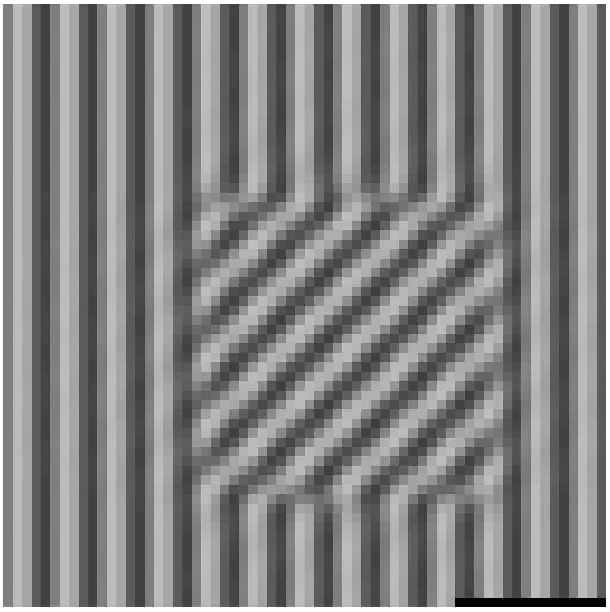

Figure 6-4: Sinusoidal texture pattern used for testing the texture detection module. A patch of texture 32 pixels by 32 pixels was rotated 45 degrees with respect to the background. The frequency of the sinusoids is $.4 \times \pi$.

using a Laplacian pyramid [1]. Opponent images are then subtracted to yield an opponency image that shows the areas of highest conflict. For our purposes, we would like the results to remain at the same level of resolution as the originals, so that they can later be combined with processed images from other modalities. In the implementation used for contour detection, the results from the four directional filters shown above are computed. The opponency images are then computed by taking the absolute value of the difference in the filtered images. This avoids both the downsizing introduced by the Laplacian pyramid and the inflation introduced by squaring each pixel before downsizing. Finally, the opponency images are normalized by dividing each pixel by the sum of the values of the corresponding pixels from each filtered image. The normalized opponency images can then be considered probabilities of textured contours along the respective line directions. The two normalized opponency images are then used as the output from the texture detection module.

[1]The Laplacian Pyramid is a powerful representation for image compression. Instead of simply averaging local pixels to downsize the image, local groups are weighted by a Laplacian function before begin summed. As an example, a Gaussian is a specialized type of Laplacian function. For more details, see [BA83]

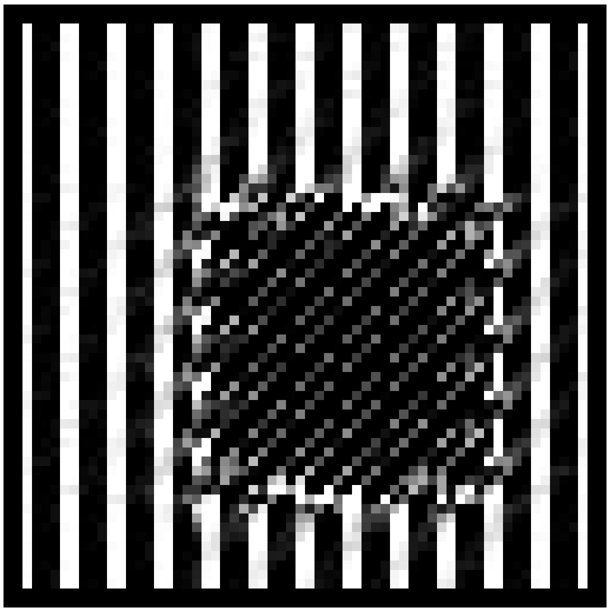

Figure 6-5: Normalized horizontal-vertical texture opponency image. The sinusoidal texture pattern in Figure 6-4 was processed by both horizontal oriented and vertical oriented filters. This opponency image is the absolute value of the difference in the filtered images.

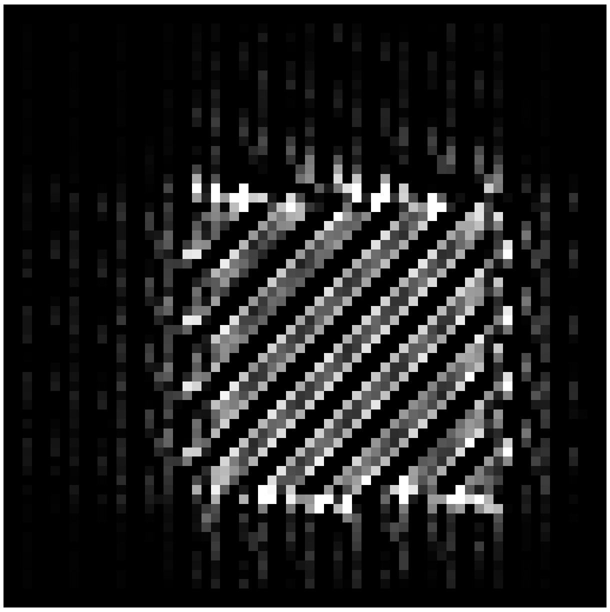

Figure 6-6: Normalized diagonal texture opponency image. The sinusoidal texture pattern in Figure 6-4 was processed by two filters oriented at +45 degrees and −45 degrees from vertical. This opponency image is the absolute value of the difference in the filtered images.

As was described in section 3.2.3, natural examples of pure texture are very rare. To demonstrate the results of this processing, and as part of the testing for this stage of processing, a patch of sinusoidal texture at 45 degrees was placed on a background sinusoidal texture at 0 degrees (see Figure 6-4). The sinusoids had a frequency of $.4 \times \pi$ and varied in intensity between 64 and 192 in 8-bit grayscale. The normalized opponency images are shown in Figure 6-5 for the horizontal-vertical opponency and in Figure 6-6 for the diagonal opponency. Notice that the diagonal opponency image draws out only the sections of the image that contain the shifted texture pattern, and the horizontal-vertical opponency image selects the background texture.

6.3 Gestalt Effects: Point Fields

While there are a variety of well-studied computational strategies for extracting texture and luminance information from an image, computing the effects of the Gestalt properties on an image are less well known. For our experimental implementation, we will use the basic format outlined by Guy and Medioni for resolving the Gestalt properties of continuation and closure [GM93]. The basic concept of the algorithm is to use local fields around salient features to extend into global configurations. The algorithm works by letting each pixel vote for edges within a local neighborhood, finding the most likely edges for each pixel location, and extending the likelihood of other nearby pixels of having lines in that specific orientation. The algorithm that is presented below is a slight simplification of the complete extension field methodology explored by Guy and Medioni, but provides nearly identical results for the class of images that our hardware can produce.

The algorithm begins by allowing each pixel in the input image to cast "votes" for edges at locations in the final image. Each pixel from the input image registers a vote for lines of specific orientations and intensities with the pixels in a local field in the final image. Votes can be mentally pictured as vectors extending from each pixel in the final image that indicate an edge direction (by the direction of the vector) and an edge intensity (by the length of the vector). To determine the exact vote that is

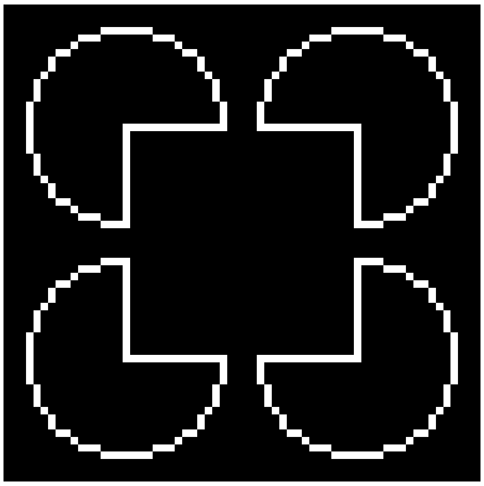

Figure 6-7: Kanizsa square test image for the Gestalt effects processing. The image is a 64×64 bitmap.

cast, you need know only the location of the input pixel, the location of the pixel in the final image, and the intensity of the input pixel. The orientation of the vote is determined by the orientation between the input and final pixels; pixels in the same row result in a vote for a horizontal line and pixels in the same column result in a vote for a vertical line. The strength of the vote is equal to the intensity of the input pixel weighted by a function of the distance between the pixels. The weighting factor is:

$$W_x = x * \frac{e^{-k(x^2+y^2)}}{\sqrt{x^2+y^2}} \qquad\qquad W_y = y * \frac{e^{-k(x^2+y^2)}}{\sqrt{x^2+y^2}} \tag{6.6}$$

where W_x and W_y are the row and column components of the weighting vector, x and y are the differences in row and column positions between the pixels and k is a scale factor that limits the size of the field. For the integrated contour experiments, $k = .2$ and the field size is limited to 3 pixels. The size was chosen to maintain the same size as the Sobel operator, and the constant k was chosen to minimize the actual values of the weighting field outside that range.

After the voting has been completed, each pixel in the final image has a list of votes for lines of specific orientations and strengths. If we view each of the votes at a specific point in the final image as a vector, that is, an $[x\ y]$ pairing, the list of votes can be made into an $n \times 2$ matrix where n is the number of votes. Guy and Medioni have

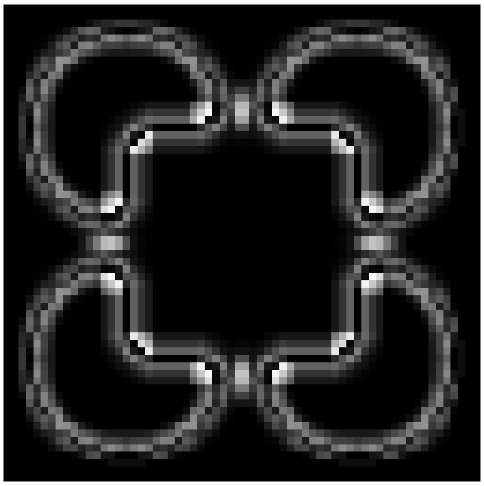

Figure 6-8: Effects of the Gestalt module on the Kanizsa square test image. Notice that the algorithm produces high probabilities of edges in the missing portions of the perceptual square.

shown that the minimum eigenvalue of the covariance matrix of this vote matrix is the saliency of the edge map accounting for the Gestalt effects of continuation and closure [GM93, p. 7]. For our implementation, this saliency map is computed by finding the singular value decomposition of the covariance matrix. Another way of viewing this computation is that the covariance matrix gives the most likely edge directions and intensities, and the singular value decomposition finds a single likelihood of this edge direction.

Consider the example of the Kanizsa square (see Figure 6-7). We expect the voting strategy to allow the straight line segments to extend across the filled gaps and complete a square. The pixels in the middle of each side of the perceptual square will receive votes for a line of that orientation from both sides, and if the field is large enough to collect enough votes the extension will be formed. The image in Figure 6-8 is the result of applying this algorithm to the Kanizsa square of Figure 6-7 with a field size of 7 and $k = .2$. The larger field size was chosen to highlight the workings of this process, since the results for a smaller field are much more subtle. Notice that the Gestalt processing is not the same as luminance detection; the original luminance edges in the input image are not detected by the Gestalt processing stage. The picture shows the probability of lines extending into adjacent squares to provide completeness or good continuation. The areas of high activity are near junction points (such as the center of the circles) or where gaps occur in what are perceived as straight lines (along the edges of the square).

6.4 Integration

Once the results from the luminance module, the texture module, and the Gestalt effects module have been collected, the integration step is almost trivial. The weights will be chosen so that each module has an equal effect on the final output. Because texture gives two results, we assign a weight of 1 to each texture result and a weight of 2 to both the luminance and Gestalt effects results. The weighted images are summed and then normalized by dividing by the sum of the weights (6). The final integration image is then ready for use by other processing stages, behavioral routines, or just to display. The next chapter will be devoted to demonstrations of the integration results for this implementation of the integrated contour model.

Chapter 7

Experimental Results

This chapter demonstrates the results of the integrated contour implementation developed in the preceding chapter. The first example will present the basic results produced by the algorithm and demonstrate how the algorithm detects features that the individual modalities lack. The second example demonstrates the operation of the integrated contour implementation on a more complex image and demonstrates how each of the individual modalities provide different information on the same image. The third example illustrates the resolving capabilities provided by integrated contours over individual modalities. The final example will show how integrated contours allow for failures in single modalities.

For all of the examples presented in this chapter, a test object was placed on a lab table for viewing by the camera system on Charlotte. The center 64×64 pixels of the image were used for the processing stages. The four test objects presented here were chosen from a variety of objects that were viewed by the system. These four test images were chosen because they provided good demonstrations of the types of effects that were observed to varying degrees in the other images.

7.1 Example 1: Basic Results

The first test object used was a normal paper plate. The plate has a flat central region surrounded by a raised edge. The edge contains many small radial ridges.

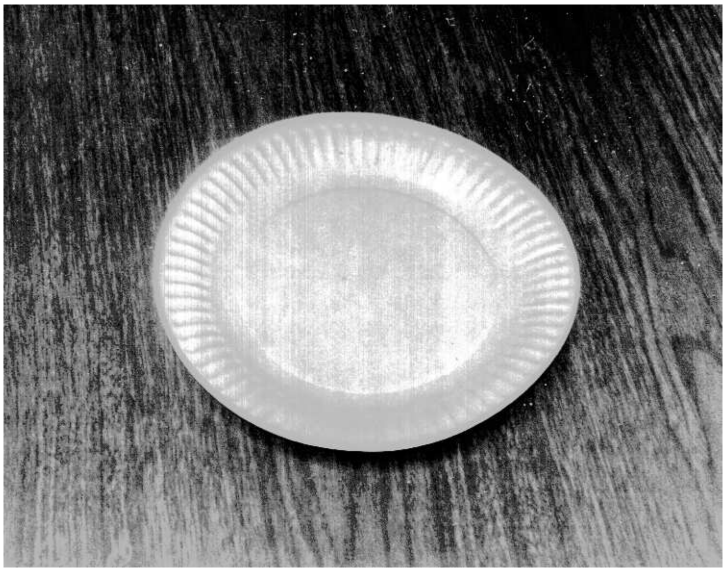

Figure 7-1: A paper plate was used as the first test object.

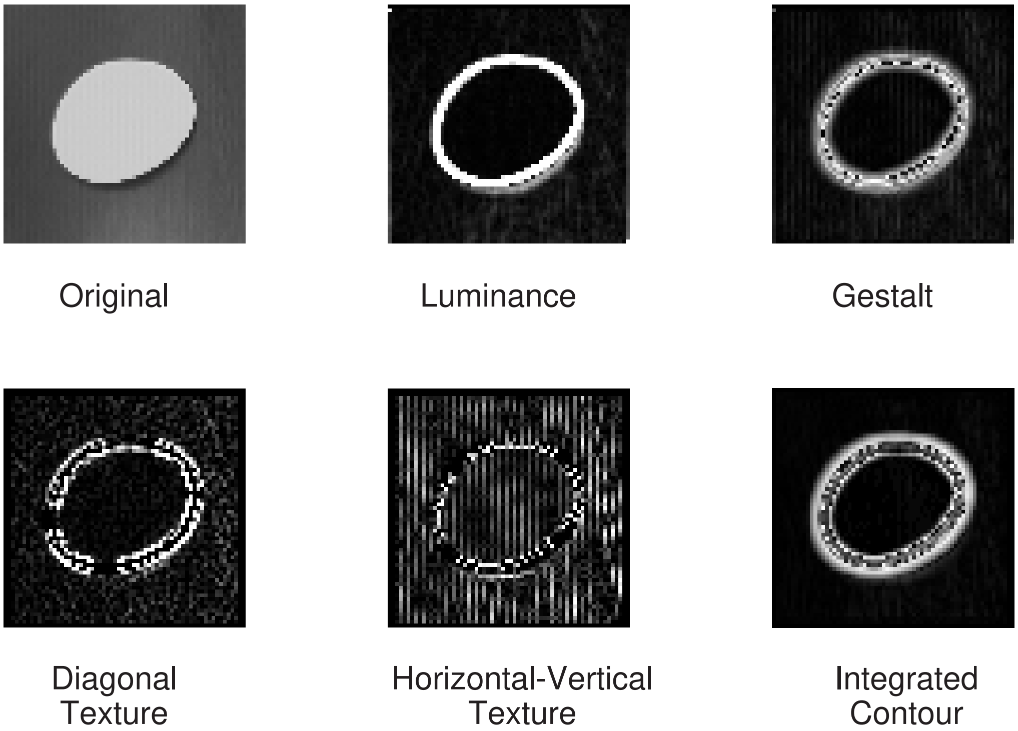

Figure 7-2: Processing results for a paper plate test image. The original image for processing is shown along with the results from each of the modules. The final result, labeled "Integrated Contour," shows some features that are not clearly defined in the intermediate images.

A photograph of the plate is shown in Figure 7-1. Figure 7-2 shows the results of the integrated contour processing. The first image in Figure 7-2, labeled "Original," shows the 64×64 image used for input to the luminance, Gestalt effects, and texture processing modules. The second image, labeled "luminance," shows the results of applying the Sobel operator on the input image. The luminance image accurately shows the outside edge of the plate, but misses any internal details. The "Gestalt" image shows the results of the point-field voting scheme described in the previous chapter. The Gestalt image shows both the outer edge of the plate, the internal edge between the flat portion of the plate and the raised edges, and some of the ridges along the outer rim. The two texture images, "Diagonal Texture" and "Horizontal-Vertical Texture" are the normalized opponency images produced by the Bergen and Landy texture algorithm. The diagonal texture image is most sensitive to the edges of the plate that are $+45$ or -45 degrees from vertical. The horizontal-vertical texture image selects out the horizontal and vertical edges of the plate, as well as the simulated-wood grain of the table.

The final image, labeled "Integrated Contour," shows the result of averaging the four previous images as described in the previous chapter. The integrated contours method shows the outer edge of the plate, the inner edge of the flat portion of the plate, and some of the radial ridges that cover the exterior ring. The integrated contour representation combines the different methods in an effective and simple representation. The details that are selected by only one modality, like the texture of the grain of the table, are drowned out by some of the other methods if there is no supporting evidence. This also has the effect of removing the noise involved in each stage of processing from the final image. Features like the radial ridges that appear in only one modality, but have supporting influences from other modalities, are heightened in the final result. The integrated contour image shows many of the features of the original images in a simple, compact representation.

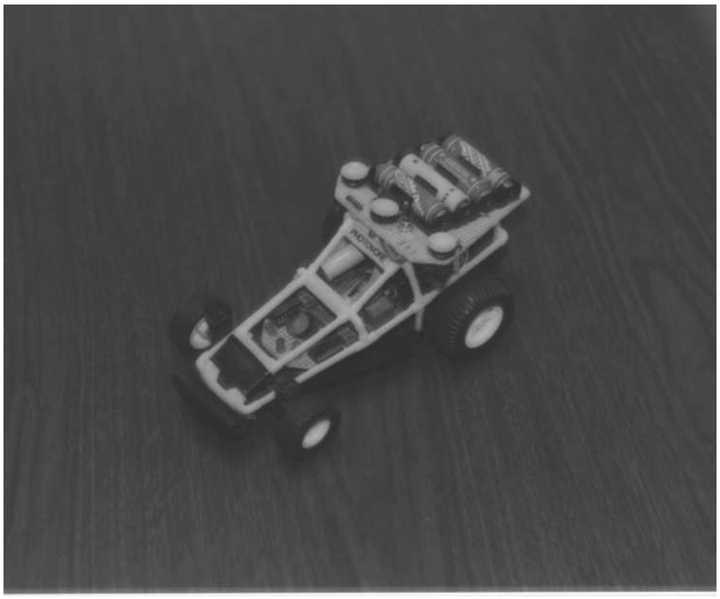

Figure 7-3: A robot car used as the second test object.

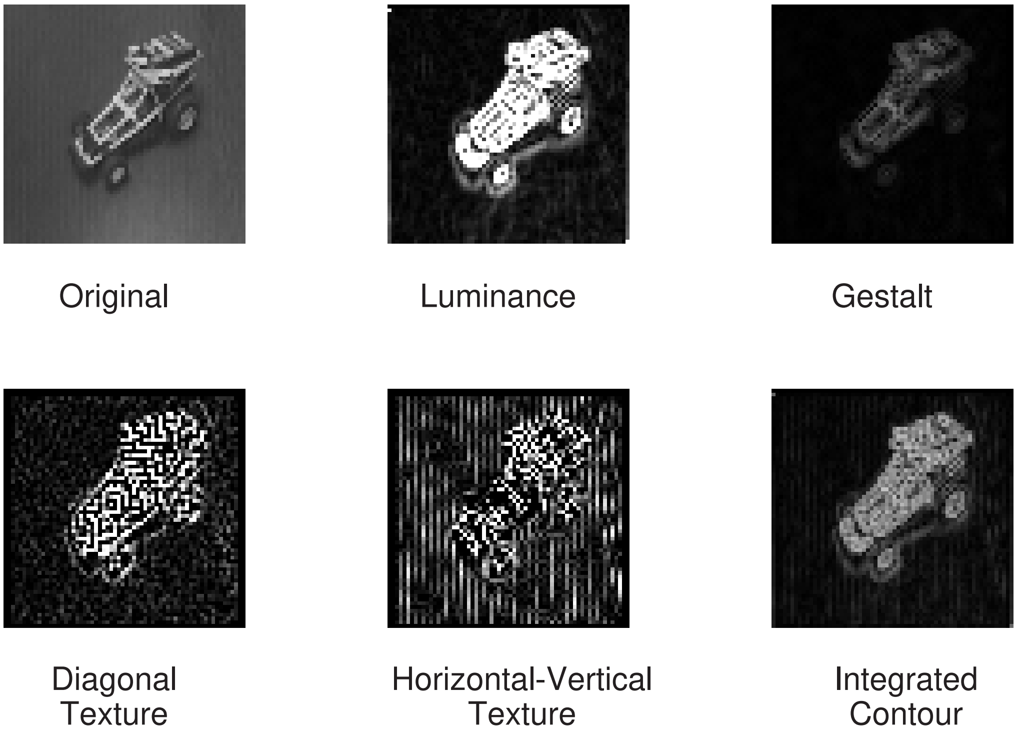

Figure 7-4: Processing results for the robot car test image. This example illustrates the operation of integrated contours on a more visually complex object, and serves to show the different contributions made by the luminance, Gestalt effects, and texture modules.

7.2 Example 2: Complex Images

The first example gave a good general introduction to the kinds of processing that integrated contours perform. The second example is designed to show the results on a visually more complex object. The test object used was a robotic car, named the "Photovore." A photograph of the test object appears in Figure 7-3. The car was selected as a test object because of the many small features contained inside the roll-bars. The results of the integrated contours algorithm are shown in Figure 7-4. The luminance results accurately portray the exterior edges of the object, but become confused near the roll-bars where there are many small details. The Gestalt image, however, draws out only the details that contain supporting influences from elsewhere in the image. The exterior of the car and the roll-bars are very clear, but much of the small detail inside the image is suppressed. The texture images also show different details from the original image. The diagonal texture shows almost constant low-levels of activity inside the outline of the car, but very little activity outside. The horizontal-vertical texture image again picks out the grain of the table as well as high levels of activity near the spoiler.

The final results of the integrated contour approach show a composite image of the car. The final image lacks some of the strong definition along the exterior of the car that the luminance image maintained, but also contains a more accurate picture of the details inside the car. The luminance image contributes the strong outlines of the car itself, but lacks the internal definitions. The texture images contribute to some of the details of the internal structure, and also some of the grain of the table. The Gestalt image helps to highlight the presence of the roll-bars, and brings out the more salient edges along the wheels. The final result of the processing shows that integrated contours can handle even complex visual scenes.

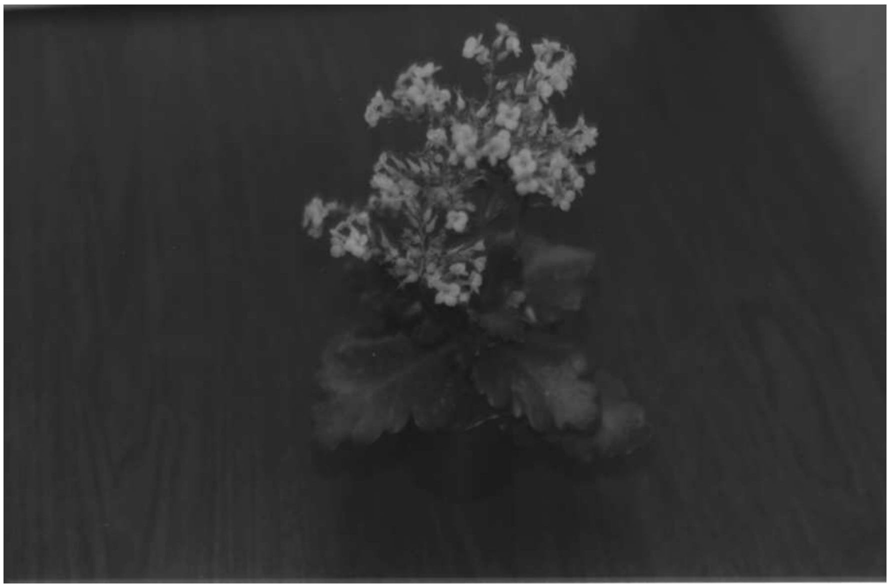

Figure 7-5: Potted flower used as the third test object.

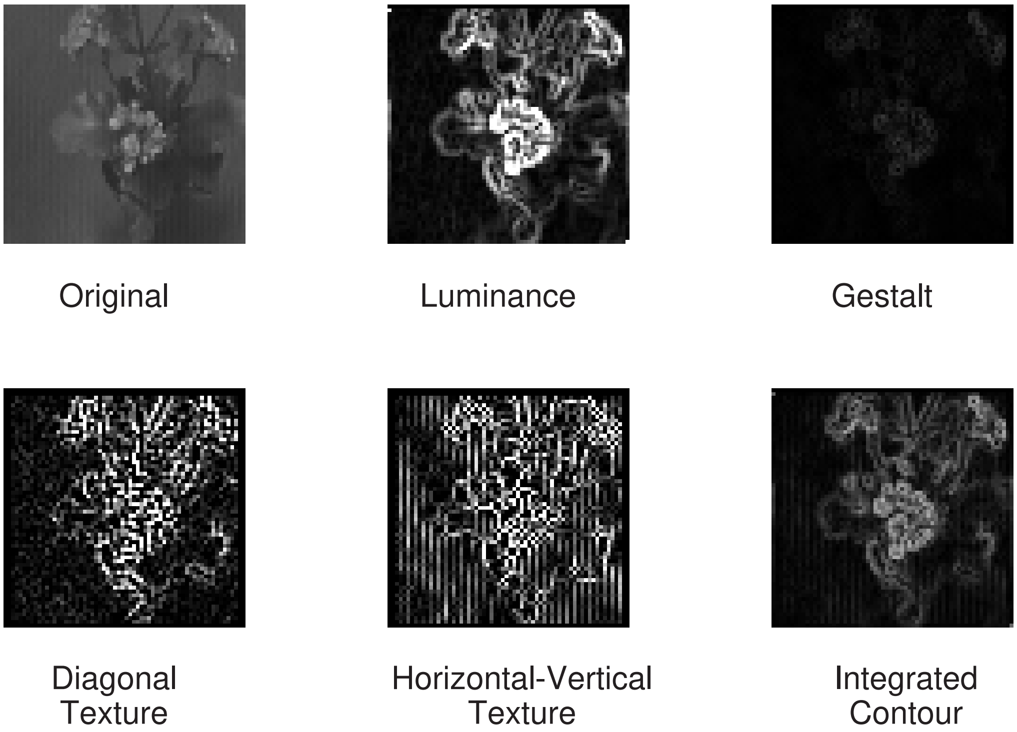

Figure 7-6: Processing results for the potted flower test image. This integrated contour image shows better resolution of detail in the group of flowers near the center of the image than any of the individual modalities.

7.3 Example 3: Resolving Power

The second example demonstrated the uses of integrated contours on complex images, and the influences of each of the modules on the final result. The third example is designed to show some of the resolution enhancing properties of the integrated contour approach. The test object used in this example was the potted plant shown in the photograph in Figure 7-5. The results of the integrated contour processing are shown in Figure 7-6. The luminance image shows good definition of the background leaves and stems, but lacks fine definition of the petals near the center of the image. The Gestalt effects picks out these petals quite well, but lacks the definition of the stems and leaves. The diagonal texture image shows high activity near the central group of petals, and some of the details around the stems. The horizontal-vertical texture image shows great activity from the table grain, and some light activity around the central flower group.

The final integrated result shows some details of the flower petals near the center of the image that are not found in any of the individual modalities. This increase in resolution power is possible because each modality allows for a slightly different view of the data present. Individual modalities, like the 3×3 Sobel operator introduce a smoothing into the image that can be reduced by using multiple techniques with different receptive fields.

7.4 Example 4: Failures in Single Modalities

The third example demonstrated the unique resolving capabilities of the integrated contour technique. This final example will show the uses of integrated contours on areas where some of the modalities fail. The test object used was a toy slinky, shown in the photograph in Figure 7-7. The slinky was chosen as a test object because it packs many edges into a very small space. The results of the processing are shown in Figure 7-8. The luminance technique performs well on the exterior outline of the slinky, but fails when the loops of the slinky become too close. The Gestalt processing

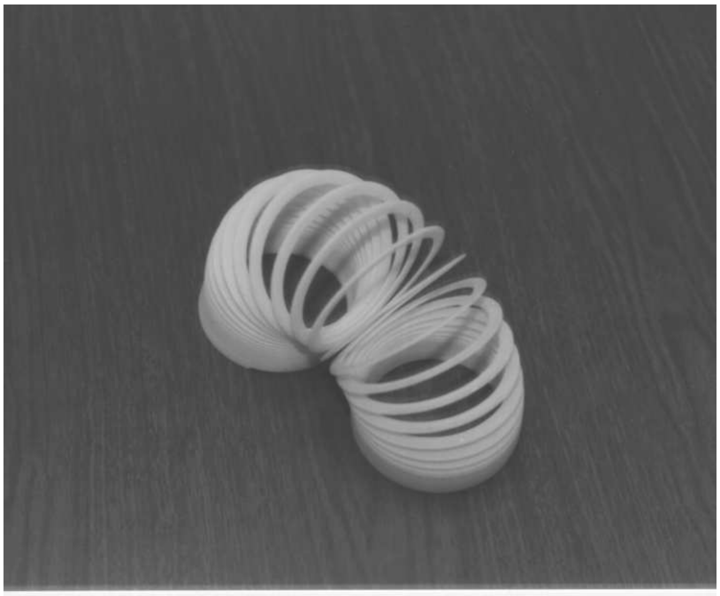

Figure 7-7: The final test object was a toy slinky.

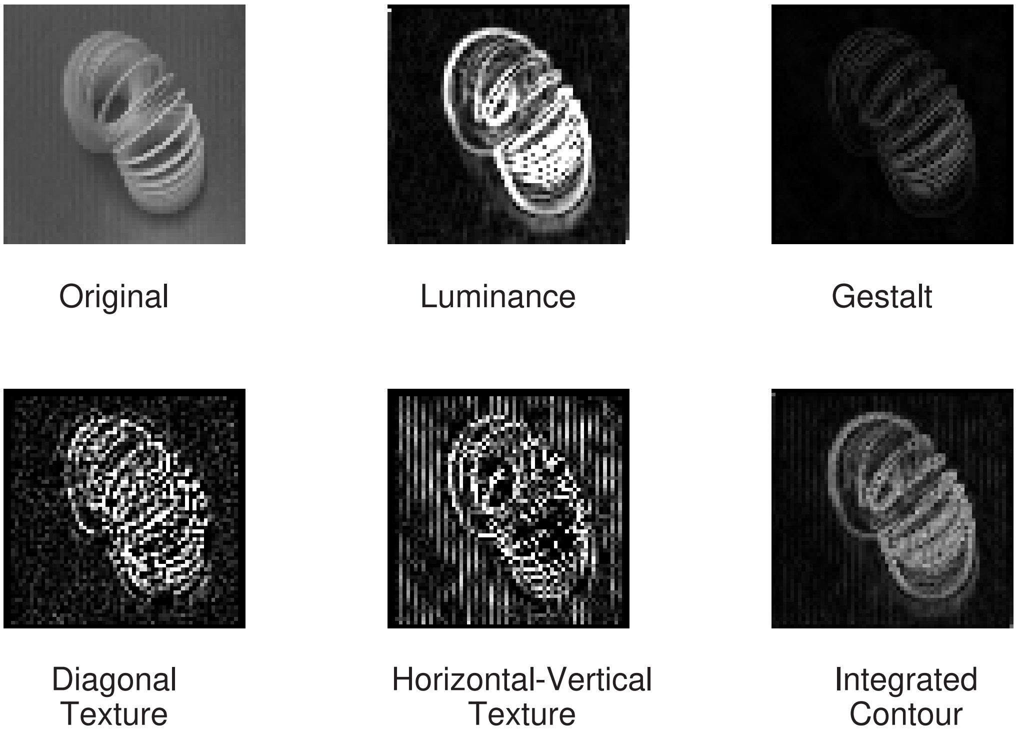

Figure 7-8: Processing results on the toy slinky test image. The final result shows the way in which strengths of the different modalities can be combined to cover over faults in a single modality. For example, while luminance has a difficult time resolving the edges of the slinky as they approach the table top, the contributions from other modalities allows for good resolution to be achieved in those areas.

produces strong curves along the tops of the individual loops at a better resolution than luminance. The diagonal texture and horizontal-vertical texture images both show high activity throughout the object. The diagonal texture image shows average detail throughout the image, while the horizontal-vertical texture shows very fine resolution near the base of the slinky.

The integrated contour image combines the different modalities in such a way that areas where one technique fails are covered by the others. At the apex of the slinky, the horizontal-vertical texture fails to resolve any detail, but the Gestalt and luminance images contribute a strong enough result to detect the contours in this area. Similarly, the luminance image fails near the base of the object where many line segments are blurred together. The Gestalt processing and the texture results help to offer finer resolution and accuracy in this region. Where one technique fails to account for details, the other techniques are able to recover. The individual modality images can also be tampered with by noise or human intervention, and these results will still hold.

The examples from this chapter have shown some of the advantages that integrated contour processing offer over individual techniques. By combining information from multiple modalities, the results that we achieve are more resistant to noise, less likely to contain modality-specific faults, and in some cases offer finer resolution.

Chapter 8

Conclusions

This final chapter will be dedicated to a review of the contributions of this work and a discussion of the future avenues of research that this work has proposed.

8.1 Summary of Contributions

Our investigation of contour perception began with two objections to the perfect slate model of machine vision. The first objection stated that vision should take accurate account of the types of stimuli and sensors that are available; vision can no more be viewed as a rectangular grid of pixels than memory can be viewed as a disk drive. The second objection stated that the processing of visual stimuli cannot be viewed as a single pipelined process that accounts only for the pixel intensity values; the influence of higher-level processing must be included, feedback must be allowed, and continuous processing must be the norm. This thesis certainly has not taken the leap to implement all of these qualifications, but it has taken the first step in that direction.

This thesis has introduced a new method for viewing the problem of contour perception. The contributions of this work to the field of machine vision can be summarized as follows:

- A classification of contours as high-level, perceptual features

- A method for perceiving contours: The integrated contours model
- A novel active vision platform: Charlotte
- An implementation that shows the viability of the integrated contour model

In Chapter 3, the problems associated with contour perception were reviewed. New definitions for the words "contour" and "edge" were introduced to re-focus this problem. From the examples of the Gestalt psychologists, demonstrations of context effects, and the examples from many individual modalities, contours were established as high-level perceptual features that can be composed and many different low-level features and high-level influences. Chapter 4 began the construction of a model of contour processing, the integrated contour model, that incorporated the many phenomena of contour perception.

The construction of this model required a new evaluation of standard vision hardware. This evaluation resulted in the construction of Charlotte, a novel visual architecture that allows for binocular, active, foveated vision. With the ability to explore new visual architectures that Charlotte provides, a subset of the integrated contour model was implemented. This implementation demonstrated the viability of the model and also served to highlight the noise reduction, error recovery, and enhanced resolution that this methodology can provide.

8.2 Future Avenues of Research

This thesis has opened more new lines of research than it has provided with solutions. The future avenues of research can be split into three categories: additional components to our implementation of contour integration, additional high-level behavioral goals to utilize this system, and evaluation of this methodology on other aspects of vision.

There are many additions that should be made to the subset of the integrated contour model that has been implemented. The addition of depth will provide the first real utilization of the unique qualities of the active vision platform. Depth also provides a strong means of contour information that has not yet been well exploited. Another interesting addition would be a module to implement the filling-in effects demonstrated by the blind spot. The filling-in effects may be helpful in providing information for occluded objects, or even in integrating information between the wide-angle and foveated cameras. Motion would be an excellent addition for rapid discrimination, and also aid in the development of processing that was not pipeline-based. Additional experiments should also focus on the use of multiple C40 boards for parallel computation of the low-level modules. The use of feedback in the integrated contour model would allow for investigations of real-time processing and of the interactions between modalities. Additional high-level procedures, like a model of shape recognition, would allow the study of how expectations can influence the low-level modules. Finally, the entire integrated contour software should be ported to run on Cog to make use of the richer variety of sensory inputs and behavioral responses.

Additional behavioral goals should also be investigated. The contour segmentation system outlined in this work is only a partial test of the integrated contour model. A high-level behavioral goal, like object tracking or recognition, or learning to follow human faces, would give a better sense of completeness to this project. By specifying other unique behavioral goals, the relative importance and influence of each modality can be more clearly studied. Learning algorithms for the weightings between modalities would need to be researched and developed. Future behaviors would also allow the use of the active vision platform as more than a static camera base.

Finally, it would be interesting to apply the methodology used in constructing a model of perceptual contours to other areas of machine vision and artificial intelligence. This type of modeling could be useful for other machine vision tasks, such as navigation or visual planning for object manipulation. The methodology of differentiating carefully between physical and perceptual stimuli could also be applied to other senses, such as audition or tactile sensation. The technique of using many simple modules for a common problem may also be useful to the study of other high-level mental tasks such as memory retrieval or path planning, but the direct extension of this technique is not clear at this time.

Bibliography

[And90] John R. Anderson. *Cognitive Psychology and Its Implications.* W.H. Freeman and Company, 1990.

[AP93] A. Azarbayejani and A. Pentland. Recursive estimation of motion, structure, and focal length. Memo 243, Massachusetts Institute of Technology Media Laboratory, Cambridge, Massachusetts, October 1993.

[BA83] Peter J. Burt and Edward H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4), April 1983.

[Bro91] Rodney A. Brooks. Intelligence without reason. Memo 1293, Massachusetts Institute of Technology Artificial Intelligence Laboratory, Cambridge, Massachusetts, April 1991.

[BS93] Rodney A. Brooks and Lynn Andrea Stein. Building brains for bodies. Memo 1439, Massachusetts Institute of Technology Artificial Intelligence Laboratory, Cambridge, Massachusetts, August 1993.

[BW93] Dana H. Ballard and Lambert E. Wixson. Object recognition using steerable filters at multiple scales. In *IEEE Workshop on Qualitative Vision.* IEEE, 1993.

[BY92] Andrew Blake and Alan Yuille, editors. *Active Vision.* MIT Press, 1992.

[Can83] John F. Canny. Finding edges and lines in images. Memo 720, Massachusetts Institute of Technology Artificial Intelligence Laboratory, Cambridge, Massachusetts, June 1983.

[CRS94] Patricia S. Churchland, V.S. Ramachandran, and Terrence J. Sejnowski. A Critique of Pure Vision. In *Large-scale neuronal theories of the brain*, Christof Koch and Joel L. Davis, eds. MIT Press, 1994.

[GGSF59] E. J. Gibson, J. J. Gibson, O. W. Smith, and H. Flock. Motion parallax as a determinant of perceived depth. *Journal of Experimental Psychology*, 8(1), 1959.

[Gle91] Henry Gleitman. *Psychology*. W. W. Norton and Company, 1991.

[GM93] Gideon Guy and Gerard Medioni. Inferring global perceptual contours from local features. *CVPR*, 1993.

[Gol89] E. Bruce Goldstein. *Sensation and Perception*. Wadsworth Publishing Company, 1989.

[Gra65] Clarence H. Graham. *Vision and Visual Perception*. John Wiley and Sons, Inc., 1965.

[Gri81] W.E.L. Grimson. *From Images to Surfaces*. MIT Press, 1981.

[HI84] B. Horn and K. Ikeuchi. The mechanical manipulation of randomly oriented parts. *Scientific American*, 251(2):100–111, August 1984.

[Hor86] B. Horn. *Robot Vision*. MIT Press, 1986.

[Hor93] Ian Horswill. Polly: A vision-based artificial agent. In *Proceedings AAAI*. AAAI, 1993.

[HS81] B. Horn and B. Schunk. Determining optical flow. *Artificial Intelligence*, 17, 1981.

[HvdHK94] Friedrich Heitger, Rudiger von der Heydt, and Olaf Kubler. A computational model of neural contour processing: Figure-ground segregation and illusory contours. In *PerAc 94' Conference: From Perception to Action*, pages 181–192. IEEE, 1994.

[Jul75] Bela Julesz. Experiments in the visual perception of texture. *Scientific American*, 232:34–43, April 1975.

[Kan79] Gaetano Kanizsa. *Organization in Vision: Essays on Gestalt Perception.* Praeger Publishers, 1979.

[KWT87] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:21–31, 1987.

[LB91] M. S. Landy and J. R. Bergen. Texture segregation and orientation gradient. *Vision Research*, 31, 1991.

[MMT76] J. J. McCann, S. P. McKee, and T. H. Taylor. Quantitative studies in retinex theory: A comparison between theoretical predictions and observer responses to the "color Mondrian" experiments. *Vision Research*, 16:445–458, 1976.

[MP90] Jitendra Malik and Pietro Perona. Preattentive texture discrimination with early vision mechanisms. *Journal of the Optical Society of America*, 7, 1990.

[MS83] J. D. Mollon and L. T. Sharpe, editors. *Color Vision: Physiology and Psychophysics*. Academic Press, 1983.

[Mum91] David Mumford. Mathematical theories of shape: Do they model perception? *Geometric Methods in Computer Vision*, 1570:2–10, 1991.

[Pin90] John Pinel. *Biopsychology.* Allyn and Bacon, 1990.

[PM76] T. Poggio and D. Marr. Cooperative computation of stereo disparity. *Science*, 194:283–287, 1976.

[PM87] Susan Petry and Glenn E. Meyer, editors. *The Perception of Illusory Contours*. Springer-Verlag, 1987.

[Ric70] W. A. Richards. Stereopsis and stereoblindness. *Experimental Brain Research*, 10, 1970.

[Sob70] I. Sobel. Camera models and machine perception. Stanford AI Memo 121, Stanford University, Stanford, California, May 1970.

[SW90] Lothar Spillman and John S. Werner. *Visual Perception: The Neurophysiological Foundations*. Academic Press, 1990.

[VP88] H. Voorhees and T. Poggio. Computing texture boundaries from images. *Nature*, 333, 1988.

[YS87] Yehezkel Yeshurun and Eric L. Schwartz. Shape description with a space variant sensor: algorithms for scan-path, fusion, and convergence over multiple scans. Courant Institute of Mathematical Sciences Technical Report 295, New York University, New York, April 1987.