REAL-TIME ATTENTION FOR ROBOTIC VISION

Gal Sela

Department of Electrical Engineering
McGill University

March 1995

A Thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfilment of the requirements of the degree of
Master of Engineering

ISBN 0-612-05474-8

Abstract

This thesis is concerned with the development and real-time implementation of an algorithm to determine interest points in a scene. These interest points will be used by a robot-mounted camera to focus its attention. The camera output has a nonuniform sampling resolution, modeled after the primate visual system. It provides a central high resolution foveal region surrounded by a much coarser peripheral region. The objective is to continuously position the camera so that the interesting areas in the scene lie within the foveal region.

To this end, a computational model of visual attention has been developed and implemented in this thesis. The algorithm is based on psychophysical experiments of human gaze fixation. Using context-free edge information as input, interest points are defined as centres of regions surrounded by edges. This is shown to be equivalent to defining interest points as the points of intersection of lines of symmetry between edges in an image. By adopting a symmetry measure based on the loci of centres of cocircular edges, a novel, real-time method for computing these interest points is proposed.

The algorithm has been implemented on a parallel network of Texas Instruments TMS320C40 (C40) processors. With this configuration, processing rates can exceed ten frames per second, depending on the algorithm parameters. The thesis also shows results of the algorithm applied to a wide range of real-world images, both foveal and peripheral, as well as an analysis of parameter sensitivity, and system throughput and latency.

Résumé

Cette thèse présente le développement et la réalisation d'un algorithme "temps réel" de sélection de points d'intérêts dans une image. Ces points d'attentions serviront à déterminer l'orientation d'une caméra montée sur un robot mobile. Cette caméra possède une résolution non-uniforme basée sur la modélisation de la rétine des primates. Elle produit une région fovéale centrale de haute résolution entourée d'une région périphérique de résolution moindre et variable. L'objectif est de centrer continuellement la caméra pour que les régions d'intérêt détectées par notre opérateur soient situés à l'intérieur de la fovéa.

Pour parvenir à cette fin, un modèle d'attention visuelle est présenté. L'algorithme est basé sur des expériences psychophysiques sur l'orientation du regard humain. L'utilisation des contours d'une image, nous permet de définir les points d'intérêts comme étant les centres de régions entourées par ces contours. Nous démontrons que cette définition est équivalente à définir les points d'intérêts comme étant l'intersection de lignes de symétrie des contours de l'image. La mesure des points de symétrie est alors obtenue par le regroupement des centres de co-circularité des paires de contours. Nous proposons alors une technique originale du calcul "temps réel" de ces points d'intérêts.

L'algorithme est réalisé sur une architecture parallèle de microprocesseurs TMS320C40 (C40) de Texas Instruments. Cette configuration permet d'obtenir une fréquence de traitement de plus de dix images par secondes, en fonction des paramètres de l'algorithme. Les résultats de l'algorithme sur une large gamme de scènes réelles sont ensuite présentés sur des images fovéales et périphériques. En conclusion, une analyse de la sensibilité de la méthode, de sa vitesse et de son temps de latence est présentée.

Acknowledgements

There are many people who have contributed to the success of this work. First, I must thank my supervisor, Prof. M. D. Levine, for his guidance and support. I wish to also thank Michael Kelly and Marc Bolduc, whose ideas and input made this thesis possible. I must also thank all the people at CIM who made this a most enjoyable place to work, my family for their support and encouragement, and Elisabeth Galarneau, without whom I could not have completed this thesis. Finally, I would like to express my gratitude to NSERC for their financial support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

This thesis discusses the development of a fast and computationally efficient mechanism for guiding visually-based attention in robotic vision systems. Furthermore, this system is biologically motivated, modelled on experimental evidence of attention in the primate visual system. The details of the design of the model and its implementation on a network of digital signal processors is presented in the following chapters. The remainder of this chapter addresses the need for visual attention mechanisms in both biological and machine vision systems. This leads to a description of the goals of the thesis and how they are achieved through a visual attention system. Finally, an overview of the progression of the thesis is given, presented as a brief outline of each of the remaining chapters.

1. The Need for Visual Attention

Recent advances in computer technology have made real-time machine vision systems a possibility, and have lead to a great deal of work in active vision [1] [5] [10]. However, despite enormous progress in recent years, machine vision systems still have a long way to progress before being able to replicate human visual performance. Many of the problems encountered in developing such a system are due to the enormous amount of data that must be processed in a short amount of time. To this end, a means of reducing both the data and computational requirements for image processing is desired.

One method employed by the primate visual system to reduce the data in a visual image is through the use of a nonuniformly sampled retina. Such a configuration consists of a central, high resolution foveal region surrounded by a much coarser peripheral region whose resolution decreases with the distance from the centre of the image. Furthermore, the primate visual system achieves a reduction in the computational requirements for analyzing these images by concentrating computational resources on salient, interesting regions of the image.

Both of the biological processes just described require an *attentional* mechanism in order to select those regions of the image that are worthy of more detailed examination. For the data reduction obtained by the use of a foveated sensor, the attentional system determines where to position the eye so that interesting parts of the image project onto the fovea. This allows these interesting regions to be analyzed at the highest possible resolution. For the computational reduction achieved by focusing computational resources on interesting parts of the scene, the attentional system is employed to determine the regions in the image that warrant allocation of these resources. However, in contrast to the attention required for foveated sensors, the mechanism controlling the allocation of computational resources does not necessarily involve eye movements since these analytical resources can be applied anywhere in the visual field.

Based on the biological evidence, some recent machine vision systems have incorporated a nonuniformly sampled sensor for data reduction [7] [68]. As well, some systems have tried to concentrate their limited computing resources on appropriate areas of an image [63] [16], thus reducing total computational requirements. For the same reasons that attention is required to guide these processes in biological vision systems, these machine vision systems also require efficient attentional mechanisms.

2. Motivation

The goal of this thesis is the development and real-time[1] implementation of an attentional algorithm for a robot-mounted, foveated sensor system. The objective is to determine interesting points in a scene so that the camera can be continuously positioned to allow the interesting regions to lie within the fovea. Furthermore, it is desired that the system be general purpose and able to function in a wide variety of real-world settings without specialized knowledge of the environment or task. To this end, it is also necessary for the algorithm to not rely on prior image segmentation or other context-dependent preprocessing.

To accomplish these goals, a biologically motivated, computationally efficient model of visual attention has been developed. This model is primarily based on the context-free (bottom-up), fixation guiding (overt) paradigm of visual attention. To achieve a real-time implementation, the algorithm is run in parallel on a network of Texas Instruments TMS320C40 (C40) digital signal processors.

[1]In the context of this thesis, the term *real-time* is used to mean as close to video rates as possible.

3. Organization of the Thesis

The thesis progresses through the motivation, development, and implementation of a real-time attentional system. This begins in chapter 2 with a review of the biological literature on visual attention, which is mainly composed of psychophysical and neurophysiological studies. Evidence from this literature shows that there are likely two distinct visual processing mechanisms in the primate visual system. The first of these, *covert attention*, is responsible for focusing the processing resources of the cortex and does not facilitate actual eye movements. The second type, *overt attention*, involves a reorientation of the eye so that the fovea is centred on different objects in the environment. This chapter justifies the selection of a model of overt attention to meet the specified objectives of the thesis.

Chapter 3 begins with a discussion of some of the models of visual attention that have been developed for machine vision applications. The lack of a fast operator to compute overt attention justifies the development of a new model, which is described in the remainder of the chapter.

The details of implementing the model are discussed in chapter 4. This chapter includes discussions on the effects of pixelization and the implementation of the model on nonuniformly mapped images. The end of chapter 4 presents a variety of results of the model applied to real-world images. This includes a comparison of these results with another system modelling overt attention [**63**], as well as an examination of the selection of various model parameters.

The first section of chapter 5 presents modifications to the implementation of chapter 4 required to achieve real-time performance. This discussion presents simplifications to the algorithm and changes to the data structures, as well as the distribution of the algorithm on a parallel network of processors. The second section of this chapter shows that the results obtained with these modifications are in agreement with the results of chapter 4. Finally, the performance of the real-time algorithm is examined, both by measuring run-times under a variety of input conditions and by examining the outputs of the algorithm through a real-time tracking situation. Both these evaluations show that the algorithm produces valid interest points at rates that are compatible with the requirements of active vision systems.

Finally, chapter 6 concludes the thesis with a discussion of the contributions of this work, its relation to previous work, and the future direction of the research.

CHAPTER 2

Visual Attention in Biological Systems

On what parts of a scene do humans focus their attention? This is a very difficult question to answer and it seems unlikely that there is a single explanation in all situations. On the one hand, when a person is searching for a particular object while performing a specific task, their attention is controlled in a top-down, context-dependent manner. On the other hand, in order to accomplish this search task, there must still be a mechanism that directs recognition processing to the relevant points in the scene. Furthermore, in order to survive, biological systems must be able to detect and respond to unexpected stimuli. This sort of attention must be driven directly from the input stimulus and therefore requires a context-free, bottom-up process. Attention is accomplished through a close coupling of these two types of processes, allowing the brain to quickly switch from one attentional cue to another in order to survive in a complex, changing environment.

Though it is acknowledged that both top-down and bottom-up processing are important aspects of any visual attention system, this work concentrates on the bottom-up, context-independent mechanisms for a number of reasons. First, most of what is known of the human visual system is based on low-level processing, making it very difficult to design a top-down attentional model that is biologically motivated. Second, top-down processing is likely performed around anchor points found during an initial, context-free attentional stage. In fact, it has been argued that context-free attentional modules form the building blocks for top-down attention [63]. Finally, the goal of this thesis is the development of a guidance algorithm for a robot-mounted foveated sensor. The objective is to position the camera so that interesting regions fall within the fovea, ready for further high resolution processing. This goal lends itself to the bottom-up, context-free paradigm. Once an object is positioned within the fovea, recognition algorithms can be used to determine what the object is and guide context-dependent attention. As well, by stressing bottom-up processing, the system will be applicable to a wide variety of situations, and will be able to perform in the absence of any specialized knowledge. However, the system should still be capable of incorporating top-down constraints in its search, and the integration of such components will be discussed briefly in chapter 6.

1. Psychophysical Studies

Many of the mechanisms of human visual attention have been discovered through psychophysical experiments. In these, human performance is examined during an assigned visuomotor task. Two basic models of visual attention have resulted from this research, the *zoom-lens* and the *spotlight* models. The first of these, popularized by Erikson and associates [21] [20], building on work by Jonides [34], proposes that attention is analogous to a zoom-lens system. At low zoom powers, attentional resources are concentrated evenly across the entire visual field. When a stimulus is flashed, the attentional system zooms in to that area, allocating a disproportionate amount of processing resources to it. However, regardless of zoom power, visual processing is always conducted in parallel across the entire visual field, with more processing concentrated in the zoomed regions.

In contrast, the spotlight model, first introduced by Neisser [53], then developed by Treisman [79], proposes two distinct stages of visual attention. According to this model, the first or *preattentive* stage is performed in parallel across the entire visual field. This stage would be responsible for figural and textural segregation. The second or *attentive* stage is then performed sequentially, applying more sophisticated analysis to the elements segregated by the first stage. The spotlight metaphor comes from the sequential scanning of the attentive stage. This is by far the most accepted paradigm of visual attention and many models have been derived from it.

1.1. What features catch the eye ? Though the models described above try to explain *how* attention is processed in the brain, they do not address the more fundamental question of *what* features the brain chooses to attend to. For context-dependent attention, this question can be answered, rather simplistically, by stating that attention moves towards objects that match the parameters of what is being searched for. However, as was observed in the experiments that gave rise to the spotlight model of attention [35] [36] [53] [78] [79] [80] [81], a preliminary, low-level segregation stage must occur even for context-dependent attention. As well, examining the interaction of stimulus and higher level feedback from the brain in a context-dependent attention task is extremely difficult. This is compounded by the fact that few details are known about the higher level processing occurring in the brain. For these reasons, most of the research in visual attention has concentrated on low level, context-free stimuli.

In analyzing the results of these experiments, a distinction between two alternate forms of attention, *overt* and *covert*, must be made. Overt attention involves an actual shift in the gaze fixation point to the point being attended to. In contrast, covert attention involves a shift of attentional processing to a new location in the visual field, without a corresponding fixation shift. Early theories made little distinction between these two attentional forms. For example, Hebb [**31**] proposed that perception is based on the sequential scanning of edges and corners, which are of significant importance to recognition. This view was supported by Attneave [4] who showed that image points of maximum curvature are good candidates for shape segmentation. In addition, he showed that connecting these points with straight lines produces an image which can be easily recognized. Hebb further reasoned [**31**] that since the visual system is processing corner and edge information, it would be plausible that gaze fixation would also be directed to these points. Experiments on newborn babies [**39**] [**29**] [**69**], which found that their gaze tends to be attracted to contours, seemed to further support this point. However experiments on adult fixations have shown that there can be a difference of as much as two degrees of visual angle between the point of attention and the point of fixation [**37**] [**58**]. Though this is not a large discrepancy, since with a two degree difference the attention point still falls within the foveal region, it has led to a proposal that covert and overt attention may be directed by separate processing channels. The covert channel determines fixation points and guides the oculomotor system, while the overt channel guides recognition processing to salient points in an image. This separation was anticipated by Richards and Kaufman [**66**] who said:

> "It is not at all necessary that the feature detectors which mediate pattern perception also drive the oculomotor orienting system. Different parts of the brain might process the same retinal activity in quite different ways. By doing so, the system as a whole might improve its performance."

Another interpretation is that overt eye movements are much more expensive, in terms of speed and mechanical costs, than covert attention shifts. Because of this, covert attention can be shifted very quickly, allowing the examination of a number of candidate fixation points before overt eye movement occurs to the most important one [**47**].

To highlight the similarities, differences, and interactions between these two attentional paradigms, some results of selected experiments on covert and overt attention are presented below. However, since the goal of this thesis is the development of a system to guide the fixation of a mobile, space variant sensor, more emphasis will be placed on experiments involving overt, or fixation guiding attention.

1.1.1. *Covert attention.* Covert attention is involved in focusing the processing resources of the recognition system in the brain. To this end, it can be reasoned that the features that are attractors of covert attention are features that are salient in object recognition. Experiments by Treisman [80] and Julesz [36] have demonstrated that humans are extremely adept at detecting a part of an image which differs from all other parts in a single aspect. This was accomplished through pop-out experiments, in which a human subject is presented with a display containing multiple objects and is asked to distinguish a *target* object from the other *distractors*. The object shapes are usually quite simple, such as bars, angles, or letters, and are characterized by a set of features such as colour, orientation, size, and texture.

The question of what features attract attention still remains. Treisman [79] found that in some pop-out experiments, reaction times increased linearly with increasing numbers of distractors, while in others, reaction times remained constant regardless of the number of distractors. She found that the particular behaviour depended on the feature differences between the target and distractors. For example, if a target has one basic feature that differs from all the distractors, such as a red dot in a field of yellow dots, then that target can be found in constant time. If on the other hand a target has no single feature which differs from all the distractors, but is defined by a unique *combination* of features, then the search time for the target increases with the number of distractors. An example of this would be searching for a red **X** in a field of yellow **X**'s and red **T**'s. Treisman explained these results by reasoning that some features are detectable in parallel across the whole image (constant reaction times), while others require a serial search of the image (increasing reaction times). By determining which unique features are detectable in parallel, an inference can be made as to which basic features attract early visual attention. In this way, Treisman found evidence that these features include line ends, closure, orientation and curvature, computed over luminance, colour, motion and depth maps.

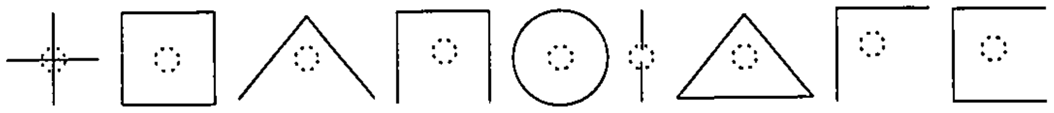
It should be pointed out that an alternate explanation of Treisman's results exists. This interpretation states that increased reaction times are not due to the effects of a serial process, but rather, are caused by a parallel process running at a slower rate due to an increase in the number of stimuli. This would occur, for example, if the brain had a limited amount of processing resources and had to divide them between all the stimuli [76] [77]. Indeed it is difficult to distinguish between these two effects. However, it is interesting to note that Treisman's interpretation of parallel and serial processing modes mirrors the preattentive (parallel) and attentive (serial) stages of the spotlight model of visual attention.

In further experiments, Julesz [36] [35] found the basic preattentive attractors, which he called *textons*, to correspond to bars, ellipses, line segments, line terminations and line crossings, along with attributes such as colour and orientation. He also showed that texture information, as well as textons, are important for figure/ground segregation. He theorized that texture is computed locally by calculating distribution statistics over the textons. Attention in the serial, attentive stage of visual processing would then be attracted to singularities of up to the second order in the statistical distribution of textons.

1.1.2. *Overt attention.* The results of studies of covert attention show that features are distinguished when they differ from their neighbours. However, there is evidence from studies of overt attention to indicate that an object's shape can be a strong fixation attractor on its own. As well, further experiments have shown that the spatial distribution of objects in a scene can also influence fixation tendencies. Both of these results are related by the notion of centroids. Human gaze appears to be strongly attracted towards the centres of small, bounded, and symmetrical shapes [37]. As well, when a group of objects is being fixated, the fixation point tends to fall near the centre of the group [15].

Bruell and Albee [9], and Pitts and McCulloch [55] have proposed that eye fixations in adults are drawn to centroids of luminance. This behaviour was observed by Kaufman and Richards [37], [66], who examined adult fixation positions when viewing simple, uniformly shaded, geometric shapes. They found that for small objects subtending less than five degrees of visual angle, the centres of the objects, as opposed to the vertices or corners, were generally fixated (see Figure 2.1). This was true regardless of whether the object was open or closed. For larger objects, fixation became more scattered. Even for these objects however, fixation tended to fall within the object outline rather than on the contours or vertices. Kaufman and Richards explained these results as a tendency for fixation to be attracted to the *centre of gravity* of small objects. They reasoned that if corners and edges were important constructs in analyzing image information, by fixating the centre of gravity of small patterns the eye is oriented so that these salient points can be processed simultaneously, at high foveal resolution, with no need for further motor activity.

Further evidence for the fixation attraction of object centres comes from He and Kowler [30]. Their experiments involved measuring the fixation of subjects viewing triangles. They found that fixation is directed more accurately to a target at the centre (symmetric point) of a triangle than to other locations within the shape. They further showed that when



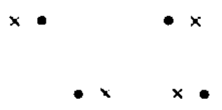
Results of a study by Kaufman and Richards [37]. Each figure subtends about two degrees of visual angle. The dotted circles outline the regions in which 86% of the fixations occurred.

FIGURE 2.1. Spontaneous fixation tendencies for various figures

subjects were asked to fixate to the whole triangle, their fixations tended to land near the symmetric point.

The perceptual processes involved in determining the centre of gravity of an object were examined by Proffitt and his associates. They found that the actual perceptual centre of an object is determined by its boundary configuration, and not by the luminance distribution of points within the object [61]. This result is important in that it indicates that the visual system relies primarily on edge information in determining the centre of a shape. In further work [60] [59] they found that the number of symmetry axes of an object contributes to how well an object's centre of gravity could be determined. This finding is further supported by Locher and Nodine [44], who found that, in the examination of abstract art, fixations are concentrated along the axes of symmetry. However, if there is no symmetry in the picture, fixations are more evenly distributed to all areas of the image. Finally, symmetry has also been linked to the separation of figure from background in images. Objects with a high degree of symmetry are more likely to be interpreted as figure than those with less symmetry [67]. In further experiments, Kaufman and Richards tested how the degree of acuteness of an angle attracted fixation. They found that the more acute the angle, the more it attracted fixation, with a maximum preference for angles of twenty degrees. As angles were made smaller than this their attraction decreased. They also found that angles are preferred over lines, and this result may explain the drop in attraction for very acute angles, since the shapes formed by these angles are so thin they begin to approximate line segments.

Another attentional phenomenon, termed the *global effect*, is observed when multiple part objects are being attended to. When such targets are presented, the orienting eye movements tend to be directed to an intermediate position between the elements. This was first demonstrated by Coren and Hoenig [15], who used the term "centre of gravity" to describe the position of fixation attraction. They instructed their subjects to shift their fixation to a specific target point, with a neighbouring distractor point positioned nearby. They found that the length of the initial saccade, or rapid eye movement, positioned the

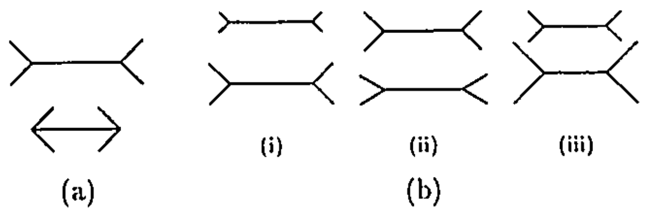


The distances between the two stimuli (•) are exactly the same in both images. The effect of the extraneous stimuli (×) causes an overestimation of length in the upper image and an underestimation in the lower image.

FIGURE 2.2. Differences in perceived distances due to extraneous stimuli

fixation point between the target and the distractor point. They further found that as the distractor was moved farther from the target point, the saccade length increased. This effect was linear up to a maximum separation of approximately two degrees. As separations were increased beyond this threshold, fixation gradually moved back towards the target point. This phenomenon has been explained by Coren [14] as an attempt by the brain to position the eyes in order to maximize the information falling on the effective fovea (2 - 4 deg. of visual angle). For points that are close together, positioning fixation between these two points allows both points to be in the fovea. For points that are farther apart, fixating to the centre of gravity of the two points no longer allows the target point to land in the fovea, and thus the target point is fixated exclusively. Subsequent tests have revealed that if one of the objects is made larger [23], more intense [17], or has a higher texture density [46], the saccade lands closer to that object. Finally, Findlay [24] showed that object boundary information is given particular emphasis by the saccadic system.

Further examination of the global effect was based on the theory that the extent of eye movements influences perception. The basic premise of this theory is that the motor energy required to move the eye from one point to another influences the perceived distance between these two points. A slight extension of this theory states that a tendency to eye movement, or efferent readiness, rather than actual eye movement, is all that is necessary for this distance perception to be noticed. The strongest evidence in support of this theory comes from Festinger, White and Allyn [22] who observed that the magnitude of the perceptual illusion matched the size of eye movement errors. Further experiments by Coren measured a subject's ability to estimate the distance between two points under the influence of nearby, extraneous stimuli [14]. In agreement with his earlier results, Coren found that the distance between the centres of gravity of the target points and the extraneous stimuli closely matched the error in distance perception (see Figure 2.2). This effect increased up to a certain distance between target and stimulus, then decreased gradually with further distance. These results have been used to explain certain visual illusions, such as the Mueller-Lyer illusion (Figure 2.3). In this illusion, the centre of gravity of the "wing"

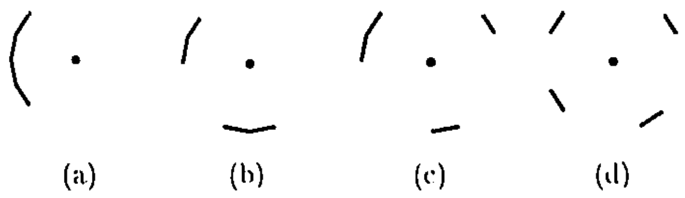


(a) The standard form of the Mueller-Lyer illusion. The two horizontal lines are the same length, though the top one appears longer than the bottom one. (b) The magnitude of the illusion can be changed by (i) changing the length of the wings or (ii) changing the wing angle. Note that the illusion weakens if the wings are made too long (iii).

FIGURE 2.3. The Mueller-Lyer illusion

sections at the ends of the line attract fixation, causing errors in the perceived length of the line. Coren [**13**] found that moving the centre of gravity of the wing, either by changing the wing angle or increasing the wing size, correlated directly with a change in illusion magnitude. As the centre of gravity was moved out, away from the line end, the perceived length of the line increased. This effect occurred up to some maximum distance between line end and wing centre of gravity. At larger distances, the wing centre of gravity no longer attracted fixation, since the two points could not both be in the fovea, and the effect of the illusion decreased. In similar experiments, Virsu [**82**] showed that illusions similar to Mueller-Lyer can be produced with any shape distractor, not just wings, placed near the line ends. These results strongly support the hypothesis that object centre of mass is an attractor for fixation.

Finally, there is also evidence that *enclosure* is perceptually significant in determining where to fixate. The importance of enclosure in figure/ground segregation was realized long ago by both artists [**3**] and psychologists [**65**]. However, it is clear that the strict mathematical definition of enclosure is too constrained to account for perceptual phenomena. Mathematical enclosure is an all or nothing property; a point is either enclosed by a contour or it is not. In contrast, experiments by Gillam [**27**] have shown that humans perceive enclosure as a continuous, graded quantity. Her results indicate that the degree of perceptual enclosure is a function of the gap sizes in a contour. For example, the sum of all the contour lengths surrounding each centre point in Figure 2.4 is the same. However, the perception of enclosure is clearly greater when there are many small breaks (Figure 2.4(d)) rather than one large break (Figure 2.4(a)). Similar results have also been shown by Elder [**19**] and Treisman [**81**] and a computational model of graded enclosure was recently proposed by Kelly and Levine [**38**]. Treisman's experiments also found that the degree of enclosure affected response times in visual search tasks. This seems to contradict the



All of the points (•) have the same total length of contour surrounding them, however, the perception of enclosure clearly increases from (a) to (d).

FIGURE 2.4. Graded enclosure

results of Kaufman and Richards described earlier, which showed that centres of gravity of closed figures attracted attention as well as open figures. However, they go on to explain that there appears to be a completion process in operation in the brain, causing many open figures to behave as if they were closed [**37**].

It should finally be noted that all of the fixation tendencies observed above apply to low-level, context-free viewing. All of these processes can easily be overridden when performing context-dependent tasks. Evidence for this was found in the experiments of He and Kowler [**30**] and, more directly, by Findlay [**23**], who discovered that the more rapidly a saccade is made, the more it is influenced by the global properties of the target configuration. Findlay argues that the slower a saccade occurs, the more time there is for voluntary influences to override the reflex tendencies.

In conclusion, the above evidence suggests that there are three basic factors which attract overt attention. First, fixation is directed to the centres of gravity of small objects in the field of view. The determination of centres of gravity is mediated by the boundary configuration and degree of symmetry of the object. Second, when multiple objects or multiple part objects are present, overt attention is directed to an intermediate position between the elements of these objects. Finally, the degree of enclosure of a point affects the degree to which it is an attractor of gaze fixation.

2. Neurophysiological Studies

There are still many unanswered questions regarding the functioning of the primate brain, and this is especially true with regard to the visual system. Furthermore, as processing progresses through the different levels of the primate visual cortex, representation of the visual scene becomes increasingly more abstract and complex, relying on global scene and contextual cues. For this reason it is extremely difficult to obtain any information about the neural processes guiding top down attention. However, it is possible to make a number of inferences regarding bottom up attention based on neurophysiological and anatomical information. Though an extensive review of all the neurological studies of attention is beyond the scope of this work, a few findings, related to the psychophysical research presented previously, will be discussed in this section.

There is some neurological evidence supporting the separation of the covert and overt processing channels. These two channels are mirrored by the *parvocellular* and *magnocellular* streams in the visual processing system. The parvocellular path appears to be involved in determining what an object is, whereas the magnocellular path seems to be responsible for determining where an object is located [48]. These two paths remain relatively well separated throughout the visual system. Both paths begin with two distinct types of cells at the retinal ganglion level, P cells for the parvocellular stream and M cells for the magnocellular stream. This separation continues, with some crosstalk, as both streams pass through the lateral geniculate nucleus (LGN) to area V1 and then area V2 of the visual cortex. After this the two streams diverge completely. The parvocellular stream proceeds to V4 and the inferior temporal cortex (IT), which is involved in stimulus identification and memorization. The magnocellular stream continues through the middle temporal cortex (MT) to the posterior parietal cortex (PP), the area responsible for object localization and spatial relations. Further evidence has shown that IT neurons respond mainly to stimuli near the fovea, while PP neurons are affected by both foveal and peripheral stimuli [48]. This evidence correlates well with the functions of these two streams since an identification system would require high resolution information from the area being fixated, whereas a localization system would require information from the entire visual field.

As for direct evidence of attentional processing centres in the brain, Moran and Desimone [49] found that receptive fields of V4 and IT neurons in monkeys could contract to include only a relevant, attended stimulus, even if two or more stimuli fell in the neurons' original receptive field. This is strong evidence that there are mechanisms which can adjust the brain's response for different attentional tasks. The question remains as to what controls these tuning processes. Some researchers have concluded that the pulvinar nucleus of the thalamus is partially responsible for this function [57]. PET scans have shown an increase in blood flow to the pulvinar when subjects were told to ignore a particular stimulus, indicating increased activity in this area [41]. As well, the thalamus is ideally positioned in the visual pathway to perform a gating function, controlling signals as they pass from the retina to the cortex. Finally, patients with pulvinar lesions exhibit a difficulty in directing and maintaining visual attention [62].

Other evidence implicates other areas in the determination of visual attention. For example, cells in the superior colliculus (SC) corresponding to the target location of a saccade have been found to have an enhanced response just before a saccade begins [28] [83] [74]. This response occurs even if eye movement is prevented. Damage to the SC significantly slows a patient's ability to shift attention from one source to another. Other research has found neurons in the frontal eye fields (FEF) that respond similarly to those in the SC. However, these cells respond only if eye movements actually occur [56] [84]. The PP also has neurons that respond in the area of an attended stimulus [83]. In particular, Mountcastle and his colleagues [51] found three types of cells in area 7 of the PP of monkeys. These cells, called fixation neurons, tracking neurons and saccade neurons, respond selectively when the monkey fixates an object of interest in its immediate vicinity, when it tracks an object of interest, and when it saccades to an object of interest, respectively. As well, lesions to the PP lead to difficulty in disengaging an existing focus of attention in order to shift to another [56].

The evidence stated above shows that attention is an extremely complicated process, involving many different areas in the brain. As well, there is evidence to show that several pathways exist which can compensate for each other if one is damaged. Some effort has gone into linking this evidence in a coherent way. For example, Posner [57] has recently proposed a three stage operation for shifting attention. First the parietal cortex *disengages* the attention from its present focus. Second the SC *moves* the focus to a new target. Finally, the pulvinar *engages* attention on the new target.

Further inferences can be made by examining the general anatomy of the brain and the connectivity of neurons. For example, as was described previously, bottom-up attention must be context-free, and must primarily be a response to certain characteristics of the visual input. Therefore, in order for an attentional operator to be context-independent, it must rely exclusively on the information available from low level, early vision cues. These correspond to features, such as tangents [33] [45] and curvature [18], that are computed at the first levels of the visual cortex. As well, most neural connections in the cortex extend just 1-2 mm [33]. This corresponds roughly to the length of one aggregate field in the visual cortex. For this reason a bottom-up attentional mechanism must also be very local, as interactions across large expanses of the visual field can only occur much further along the processing pipeline.

CHAPTER 3

Modelling Visual Attention

This chapter explores how visual attention can be modelled in a machine vision system. First, a brief review of some systems implemented by other researchers will be presented. Next the motivation for the model presented in this thesis is examined, which is based on the psychophysical and neurophysiological evidence of chapter 2. This leads finally to the development of a new mathematical model of visual attention.

1. Attention in Machine Vision Systems

Recent interest in active vision has necessitated the development of systems to guide the fixation of these active vision sensors. These systems can be grouped into two types, those designed for specific, constrained tasks, and those designed to be applicable to a wide variety of tasks and real-world environments. The task-specific systems have the advantage of a priori knowledge of what constitutes an interesting object, and can use this information to choose fixations. Therefore, these systems are mainly based on top-down attentional processing. For example, the car following system of Zielke, Brauckmann and von Seelen [88] uses information about the symmetries found in the back of a car to distinguish cars from other objects.

Some systems, including the one proposed in this thesis, attempt to implement a general purpose attention model which can be used in diverse situations. As was described in chapter 2, these must incorporate a bottom-up attentional mechanism, based only on the information in the input image, since a priori knowledge of the environment is not available. Some of the proposed methods for determining such interest regions are based on texture properties of an image. For example, gray level variance is used in [50] and a measure of local business is used by [54]. Others define interesting regions based on the optical flow computed in a scene [75]. Further approaches are based on early psychophysical data [31] [4], attributing attention to regions of high edge curvature [42] [86].

Almost all of the recent biologically motivated models of visual attention have been based on covert evidence of visual attention (see for example [40] [11] [12] [16] [47] [70] [26]). These models involve the computation of a number of *feature maps*, representing the locations of various features such as orientation, colour and direction of movement. Attention is then focused on areas of the image whose features differ from their neighbours. Most of the differences between these models involve the features that are chosen for the feature maps, the ways that the different feature maps are combined, and the effect of scale on their combination.

The goal of this thesis is to develop a system which can guide the position of a robot-mounted, foveated camera system. The objective is to position the camera so that interesting regions fall within the fovea, in preparation for further high resolution processing. As well, the determination of interesting regions should initially be accomplished in the absence of specialized knowledge about the scene. Finally, a fast, near real-time implementation is required to accommodate tracking and recognition of objects in a real-world setting. Because it involves positioning the sensor in unknown surroundings, this goal lends itself to the bottom-up, overt visual attention processes described in chapter 2.

There is only one algorithm that we know of that is based on the neurophysiological evidence of overt attention. This attentional operator, developed by Reisfeld, Wolfson and Yeshurun [63], has a number of features in common with the objectives of this thesis. First, the model is biologically motivated, modelling interest points based on symmetric properties in the input image, and correlates well with the psychophysical evidence of Kaufman and Richards [37]. Second, it requires no higher level knowledge of the scene, relying only on the contours found in the image. However, as will be seen in this chapter, the complexity of this operator scales with the square of the number of edges in the image, since it requires an operation be performed over every pair of edges in the image. This makes the operator too computationally intensive to be practical for real-time implementations. Furthermore, the application of the operator to nonuniformly mapped peripheral images is not clear. These difficulties are demonstrated by the results of an attempted real-time, foveated implementation of the algorithm by Yamamoto, Yeshurun and Levine [85]. This system required a massively parallel SIMD (Single Instruction, Multiple Data) computer architecture to implement the operator. Even so, it could only achieve frame rates on the order of seconds per frame. Furthermore, the algorithm was applied to log polar images without any modifications based on this mapping.

For the above reasons, we developed a new model of overt visual attention. It is appropriate for real-time implementation and easily adaptable to nonuniformly mapped images. This system will be presented in the following sections.

2. Defining the Model

The biological processes guiding overt visual attention will be the primary motivation behind the development of the attentional model. However, a number of further implementation requirements are imposed on the development of the algorithm to ensure the system's usefulness in an active vision environment. Primarily, the algorithm must be fast, running as close to real-time as possible. As well, the algorithm must be robust and able to function in unconstrained, real-world environments. This means that the algorithm must not be too sensitive to sensor noise, object occlusions, varying lighting conditions, and other real world hazards.

Based on the psychophysical and neurophysiological evidence presented in chapter 2, a number of criteria can be defined for developing a model of bottom-up, overt attention or gaze fixation. These are:

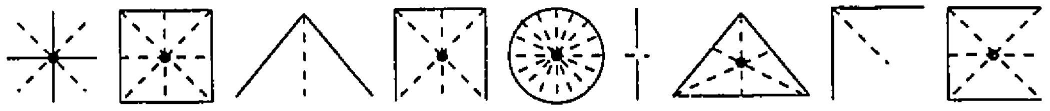
1. **Parallel processing:** Bottom-up attention, corresponding to the preattentive stage of the spotlight model, is performed in parallel across the entire visual field [53] [79].
2. **Centres of gravity:** There is a lot of evidence suggesting that eye fixations are drawn to the centres of gravity of objects in a scene [9] [55] [37] [66] [30]. This effect is most pronounced for small objects, subtending less than 5 degrees of visual angle [37]. Furthermore, the centre of gravity is determined by an object's boundary configuration, and not by the luminance distribution within an object [61]. This is supported biologically by the fact that edge information is available at an early stage of the visual processing pathway [33].
3. **Symmetry:** Symmetry is another important perceptual cue in the determination of overt attention. Evidence shows that the order of symmetry (number of symmetry axes) of an object affects how well that object's centre of gravity can be determined [60]. This is supported by studies showing that fixations tend to be concentrated along the axes of symmetry in an image [44] and that objects with a high degree of symmetry are more readily interpreted as foreground [67].
4. **Edge orientations:** The relative orientations of boundary elements affect how well they attract fixation. For example, acute angles attract fixation more than obtuse

angles, with a maximal attraction for angles of 20 degrees [37]. As well, any angle is preferred over an individual line segment.

5. **Global effect:** Multiple objects grouped close together attract attention to the centre point of the group [15]. This effect only occurs for objects separated by less than 4 degrees of visual angle, which is the maximum distance two objects can be spaced while still allowing both of them to fall within the effective fovea.
6. **Enclosure:** The degree of enclosure of a point affects its perceptual significance [19] [81]. Furthermore, enclosure is perceived as a graded quantity which is a function of the gap sizes in a contour [27].
7. **Peripheral and foveal inputs:** Neurological evidence shows that the overt attentional processing channel receives inputs from both the foveal and peripheral regions of the retina [48].
8. **Local interactions:** In order to be biologically plausible, an attentional context-free system must have an implementation which can be achieved through very local interactions. This is due to the fact that neural connections in the cortex extend just 1-2 mm [33].

In agreement with these criteria, we model fixation points as the centres of enclosed symmetric regions, based on the contours in an image (see item **2** and item **6**[1]). These points can similarly be modeled as the intersections of the lines of symmetry in an image. The validity of this approach, especially for closed figures, can be seen by comparing the points of intersection of the lines of symmetry in Figure 3.1 with the psychophysical findings of Kaufman and Richards in Figure 2.1. As well, evidence for this sort of processing in biological systems can be found (see item **3**). Each possible fixation point is also assigned a magnitude of interest. This value is dependent on two factors: the angles separating the edges contributing to the lines of symmetry (item **4**), and the degree of enclosure of each point (item **6**). Finally, points of high salience which are close together are grouped, and a single, centre point is determined for fixation (item **5**). All of these functions will be performed over the entire visual field at each frame of processing (item **1**). As well, the algorithm is adapted to compute interest points for both a foveal image, mapped in rectangular coordinates, and a peripheral image, mapped in log-polar coordinates (item **7**). It should also be noted that although the particulars of the hardware configuration for this

[1]In this section, item numbers refer to the elements in the list of model criteria in section 2 of this chapter. For example, item 2 refers to the centres of gravity entry in the list.



The lines of symmetry for a variety of shapes are marked with dashed lines (- - -) in this figure. The points marked with dots (•) are the intersections of the lines of symmetry. The validity of modeling centres of gravity as the intersection of lines of symmetry, especially for the closed shapes can be seen in these images. The agreement of this model with psychophysical findings can be seen by comparing this figure with Figure 2.1.

FIGURE 3.1. Intersections of lines of symmetry

system do not allow for a strictly local implementation (item 8), such an implementation is possible and will be discussed.

3. Developing the Algorithm

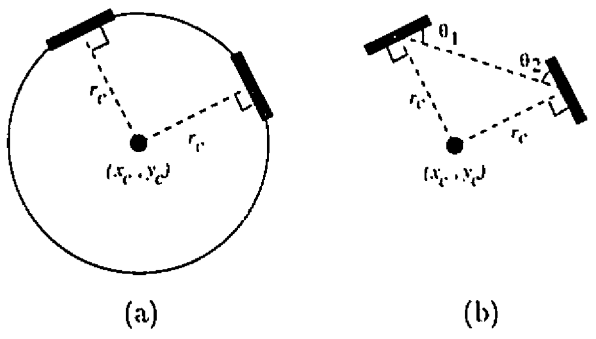
The processing for the algorithm can be divided into a number of distinct steps. Suppose an image is received from the camera and frame-grabber. First edge detection produces edge magnitude and orientation maps from which lines of symmetry are determined. Next an interest map is computed by determining where the lines of symmetry intersect. Finally, output points that are close together are grouped by smoothing the interest map and finding the local maxima. These maxima are the salient or interest points in the image.

Of the processing described above, edge detection, smoothing and local maxima determination are well known in the image processing literature (see [43] for a review) and their specific implementation is not critical. Suffice it to say that the edge detection stage computes a discrete approximation of the first derivative of the image. This results in two output images, one showing edge intensity, and the other edge orientation at each edge pixel. The smoothing stage is equivalent to lowpass filtering, and is usually accomplished by a convolution with a Gaussian mask. Local maxima are found from this image as the pixels whose values are greater than all of their nearest neighbours.

The remaining processing stages, determination of the lines of symmetry, their crossings, and the degree of enclosure of a point, are developed specifically for this thesis, and will be described in detail in the following sections.

3.1. Determining the lines of symmetry.

3.1.1. *Defining symmetry.* There are a number of definitions of symmetry found in the literature. Before describing these it is useful to introduce the notion of *cocircularity.* Two edges are said to be cocircular if a circle can be drawn to which both edges are tangent (Figure 3.2(a)). An alternate, yet equivalent way of defining cocircularity is to say that two edges are cocircular if they form angles of equal magnitude, but of opposite sign, with the

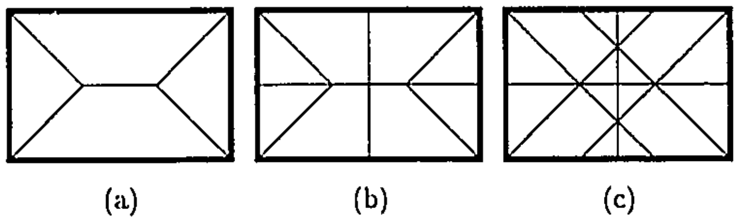


Edges are cocircular (a) if a circle can be drawn to which both edges are tangent, or (b) if the angles formed between the edges and the line joining them are of equal magnitude, but of opposite sign (if $\theta_1 = -\theta_2$).

FIGURE 3.2. Cocircularity

line joining them ($\theta_1 = -\theta_2$ in Figure 3.2(b)). The centre of cocircularity is the centre of the circle defined by two cocircular tangents (point (x_c, y_c) in Figure 3.2) and the radius of cocircularity, r_c, is the radius of that circle.

Many of the definitions of symmetry can be described in terms of cocircularity. The symmetric axis transform (SAT) [6] can be defined as the loci of all cocircular centres whose circles of cocircularity are contained entirely within a particular contour. A superset of the SAT, the symmetry set [25] is defined as the centres of all circles tangent to the curve at two or more distinct points. This is equivalent to the definition of the SAT without the restriction requiring the circles of cocircularity to be entirely within the curve. Another symmetry measure, closely related to the symmetry set, is the smooth local symmetry (SLS) [8]. The points of the SLS are the set of centre points of the lines connecting pairs of cocircular edges. A comparison of these symmetry measures is illustrated in Figure 3.3. Finally, the symmetry measure used in the system of Reisfeld, Wolfson and Yeshurun [63] should briefly be described. In this algorithm symmetry is defined as the loci of *all* points centred between two tangents, regardless of whether the tangents are cocircular or not. The magnitude of symmetry at these points is then weighted so that edges with normals pointing directly toward or away from each other produce a maximal output. A plot such as those of Figure 3.3 cannot be produced for this measure since it is not actually involved in finding symmetry axes, but rather in using the symmetric information in the image as part of an algorithm to determine interest points.

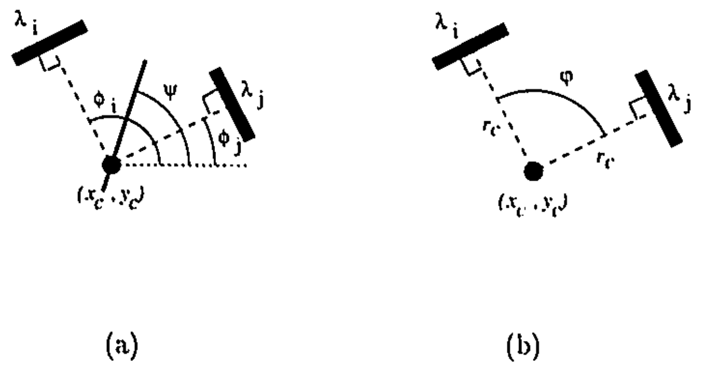


Output of different symmetry measures for a rectangle. (a) The symmetric axis transform (SAT) [6], (b) the smooth local symmetry (SLS) [8], and (c) the symmetry set [25].

FIGURE 3.3. Comparison of different symmetry measures

In comparing the symmetry measures, a few points should be emphasized. First, it should be noted that all the symmetry measures except for the SAT can be applied without any form of segmentation of the image. The SAT requires segmentation since it must distinguish between regions that are inside and outside a contour. The remaining symmetry measures are purely a function of the edges in the image. Second, it can be seen in Figure 3.3 that all the information contained in the SAT is a subset of the SLS, which is, in turn, a subset of the symmetry set. This means that the most extensive set of symmetry axes is contained in the symmetry set. It can also be seen that there are many similarities between the various symmetries, highlighting the fact that all these symmetry measures are closely related.

Selection of a particular symmetry measure for use in this algorithm is based on a number of criteria. First, the method must produce lines of symmetry that intersect in a way that is compatible with the interest model described previously. This requirement precludes the use of the SAT, since this algorithm does not output all of the required lines of symmetry. In fact, for the rectangle in Figure 3.3, the SAT completely misses the vertical symmetry axis which results in the loss of the intersection of the horizontal and vertical lines of symmetry. However, as defined by the model, this point of intersection should be the primary interest point of the rectangle. Second, the algorithm must be adaptable to a fast, parallel implementation. The algorithm proposed by Reisfeld and associates [63] suffers in this regard because it requires an operation to be performed between all edge points with all other edge points in the image. This is the reason that a different model of overt attention was pursued in the first place. For the symmetry set and the SLS these operations can be limited by taking advantage of the cocircularity requirement. In these cases only cocircular edge pairs need to be considered. As well, the computation of the symmetry set can be further optimized since the determination of the centre of cocircularity



(a) Each pair of cocircular edges defines a particular line of symmetry, with symmetry orientation ψ, bisecting the two edge normals. The orientations of the edge normals are labeled ϕ_i and ϕ_j respectively, and (b) the angle separating the edge normals is labeled φ.

FIGURE 3.4. Definition of parameters used in the model

is a by-product of the symmetry algorithm which will be described later in this chapter. In contrast, the SLS requires the midpoint of the two edges to be determined, requiring further computations. However, the reliance on contributing edge angle and the production of a weighted symmetry map in the algorithm of [63] is well correlated with psychophysical evidence, and any implementation should be modified to include these features.

3.1.2. *Computing lines of symmetry.* Based on the above arguments, the symmetry set is used as the symmetry measure for this work. However, the definition of the symmetry set is extended to allow the output to be weighted based on the contributing edge orientations and intensities as was explained previously. As well, in order to determine the intersections of lines of symmetry, a unique symmetry magnitude for a variety of symmetry orientations at each point must be determined. To this end, the *extended symmetry set* is defined as the set of all symmetry magnitudes, one for every possible symmetry orientation, at every point in an image.

We begin by defining the set $\Gamma_r(p)$ as the set of all pairs of cocircular tangents λ_i and λ_j whose centre of cocircularity is the point $p = (x_c, y_c)$ and whose radius of cocircularity is r. As well, the extent of the operator can be limited by imposing an upper bound, r_{max} on allowable radii of cocircularity. Therefore

$$(1) \qquad \Gamma_r(p) = \left\{ (i,j) \;\middle|\; \begin{array}{l} \text{edges } \lambda_i \text{ and } \lambda_j \text{ have cocircular centre at point } p \\ r_c = r < r_{max} \text{ for edges } \lambda_i \text{ and } \lambda_j \end{array} \right.$$

For every pair of cocircular tangents λ_i and λ_j in $\Gamma_r(p)$, a symmetry orientation $\psi(\lambda_i, \lambda_j)$ is computed. The symmetry orientation is defined as the orientation of the line bisecting the angle formed between the two inward pointing normals of the contributing tangents. In terms of the parameters of Figure 3.4(a), the symmetry orientation ψ is

(2) $$\psi(\lambda_i, \lambda_j) = \frac{\phi_i + \phi_j}{2}$$

Now define $\Gamma_{r,\psi}(p)$ as the subset of all cocircular tangents in $\Gamma_r(p)$ with symmetry orientation ψ. Then we can compute a symmetry magnitude, $S_{r,\psi}(p)$, for each symmetry orientation at the point p, contributed by edges at a radius r, as

(3) $$S_{r,\psi}(p) = \sum_{\lambda_i, \lambda_j \epsilon \Gamma_{r,\psi}(p)} \|\lambda_i\| \|\lambda_j\| (\sin\varphi/2)^{w_1}$$

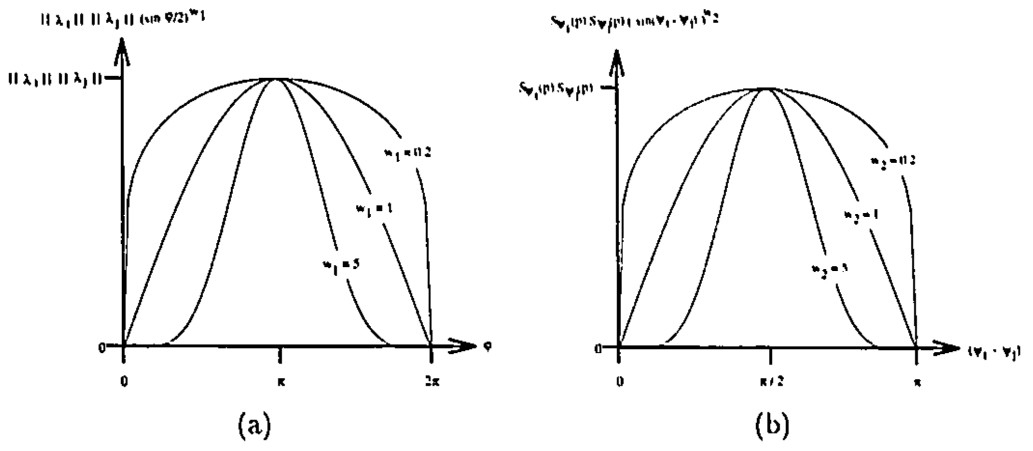
where $\|\lambda_i\|$ and $\|\lambda_i\|$ are the edge intensities of two cocircular edges and φ is the angle separating their normals (see Figure 3.4(b)), computed as

(4) $$\varphi = |\phi_i - \phi_j|$$

The reliance on φ is included to agree with the psychophysical experiments of Kaufman and Richards [**37**]. Their findings, that acute angles attract fixation more than obtuse ones, is reflected in the term $(sin\varphi/2)$. This effectively weights parallel edges (those with normals pointing toward each other) most heavily, with a decline in weight as the level of parallelism decreases. This does not completely agree with the results of Kaufman and Richards, who found a maximal fixation tendency for lines forming angles of 20 degrees. However, their experiments were only conducted on line segments that met at a point and therefore, angles smaller than 20 degrees placed the two lines very close together, and they may have begun to approximate a single line. This correlates with other experiments by Kaufman and Richards [**37**] which showed that individual lines attract fixation less than any angles do, and may explain the perceived drop in attraction for angles less than 20 degrees. As well, applying a maximal symmetry weighting for parallel edges agrees with the intuitive notion of symmetry. Finally, we note that the parameter w_1 is included to allow the relative contribution of non-parallel edge pairs to be varied. The magnitude function and its reliance on w_1 is illustrated in Figure 3.5(a).

The application of equations(1), (2), (3) and (4) results in a separate symmetry magnitude for every possible point, orientation and radius of cocircularity. To obtain the extended symmetry set, it is necessary to combine these values over all the radii. This results in a symmetry magnitude, $S_\psi(p)$, at every point, p, and orientation, ψ, and is computed as the maximum of the symmetry magnitudes over all radii.

(5) $$S_\psi(p) = \max_{r=0}^{r_{max}} S_{r,\psi}(p)$$



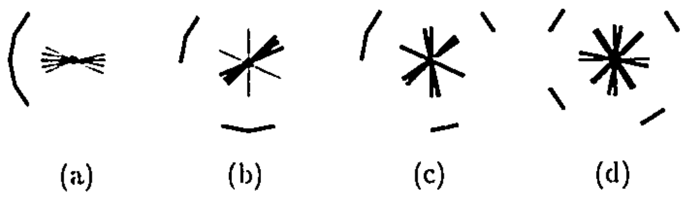
(a) The effect of the weighting parameter w_1 on the symmetry magnitude function. Notice that the maximum occurs at edge separations of π radians. (b) The effect of the weighting parameter w_2 on the interest function. Notice that this graph is identical to the magnitude function, but with a different maximum at symmetry orientation separations of $\pi/2$ radians.

FIGURE 3.5. The effect of weighting parameters w_1 and w_2

Finally, it may seem that the determination of the extended symmetry set requires interactions that span the maximal extent of the operator, r_{max}. However, there are implementations of similar algorithms [6] [71] that use a diffusion process to compute symmetry. These processes only rely on nearest neighbour communications and show that such an implementation is possible. This is a necessary requirement in order for the algorithm to be biologically plausible.

3.2. Determining intersections of lines of symmetry. As was described in the model, an interest point is defined as the intersection of lines of symmetry. The equations presented above showed how these lines of symmetry are found. This section will deal with the determination of the intersections of these lines of symmetry. Based on the definition of the extended symmetry set, lines of symmetry intersect at the points where there are two or more symmetry orientations with non-zero magnitudes in the extended symmetry set. As well, in agreement with the psychophysical data, the magnitude of the interest point is weighted by the degree of enclosure of the point. However, as was discussed in chapter 2, this measure of enclosure must be a continuous, graded quantity, rather than the strict mathematical definition of enclosure.

There have been a number of computational models of enclosure defined in the literature (see [19] for a review). However, we require a method that can be computed from the lines of symmetry determined at each point. To see how we accomplish this, the shapes of Figure 2.4



In the above images, the thickness of the symmetry lines is proportional to the magnitude of that line of symmetry. As the perception of enclosure increases from (a) to (d), the number of high intensity, orthogonal lines of symmetry also increases.

FIGURE 3.6. The contours of Figure 2.4 with their symmetries

are repeated in Figure 3.6 with their lines of symmetry added. The thickness of each line of symmetry is proportional to the symmetry magnitude, $S_\psi(p)$, of that line. Notice how, as the shapes become more enclosed, the edges contribute to strong, orthogonal lines of symmetry. In other words, parallel edges forming symmetries of high magnitude tend to contribute most to the perception of enclosure. As well, sets of these parallel edge pairs, spaced so that their resulting symmetries meet at right angles, result in a strong perception of enclosure. The lines of symmetry at a point can therefore be combined to emphasize enclosed regions by weighting the contributions from orthogonal lines of symmetry most heavily. This is accomplished by comparing all pairs of symmetry values at each point. The pairs are combined, and weighted by their symmetry magnitude and the angle between them so that orthogonal lines of symmetry have the highest weighting. Therefore, the interest value, $I(p)$, is computed as a function of all the symmetry values at the point $p = (x_c, y_c)$ as

$$I(p) = \sum_{\psi_i, \psi_j} S_{\psi_i}(p) S_{\psi_j}(p) (\sin(\psi_i - \psi_j))^{w_2} \tag{6}$$

The parameter w_2 is used to tune the sensitivity of the algorithm to non-orthogonal symmetry combinations (Figure 3.5(b)). It's effect on equation(6) is similar to the effect of parameter w_1 on equation(3). Finally, it should be noted that the determination of $I(p)$ through the application of equation(6) is dependent only on the symmetry values found at each point, and so is a completely local operation.

CHAPTER 4

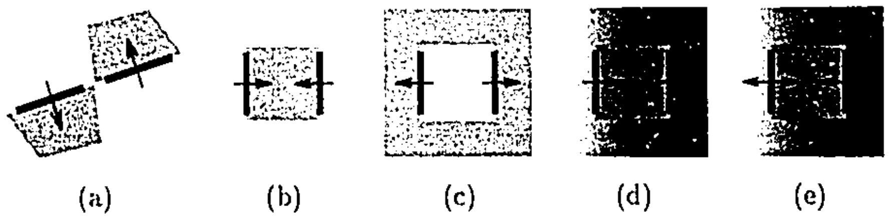
Implementation of the Visual Attention Model

Though the previous chapter described the model in detail, there still remain a number of issues which must be addressed in order to produce a working implementation. Some of these issues involve practical considerations, such as the way in which cocircular edge pairs are determined. Others are concerned with the fact that the implementation is conducted on a discrete grid of pixels. Furthermore, the spatial characteristics of this grid are quite different for foveal and peripheral inputs, and so some differences exist between the implementations for each of these two input types.

1. Implementation on Foveal Images

Foveal images are quantized to a discrete number of pixels, represented on a uniformly-mapped, rectangular grid of points. The discussion which follows describes the implementation of the model on such an input image. A number of issues must be addressed for each of the processing stages including edge detection, finding symmetries and interest points, and smoothing and finding the local maxima. These will be discussed individually in the following sections.

1.1. Edge detection. Though the operation of the edge detector is fairly straightforward, there are a few comments which should be made regarding its application in the context of this thesis. As was mentioned in chapter 3, the exact implementation of the edge detector is not critical. In practice, we use the Sobel edge operator (see [**43**]) because of its small mask size and efficiency. Like most edge detectors, this operator results in a vector quantity at every point, representing the magnitude and normal orientation of the edge at that point. As can be seen in Figure 4.1(a), there are two possible normal orientations for any edge alignment. These are distinguished by the direction of contrast change at the edge,

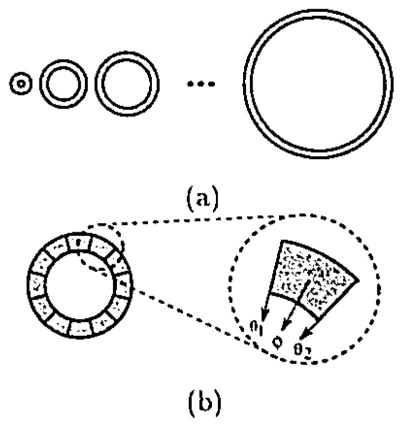


(a) Two edges with the same alignment but opposite normal orientations. (b) A shape defined by inward pointing edges, (c) outward pointing edges, and (d) both inward and outward pointing edges. (e) To find symmetries for all these cases the algorithm assumes that all edges have two normals.

FIGURE 4.1. Combining inward and outward pointing edge normals

with the normal orientation pointing, by definition, in the direction of bright to dark contrast change. There are a number of consequences of this phenomenon. First, it is clear that a pair of edges with normals pointing towards their centre of cocircularity (inward pointing) corresponds to the border of a dark object on a bright background (Figure 4.1(b)). This edge pair should clearly contribute to the symmetry set of the object. Similarly, a pair of edges with outward pointing normals can be seen as part of the border of a bright object on a dark background (Figure 4.1(c)). However, in real images, objects and background rarely have a constant intensity. In fact, it is simple to come up with a situation where a shape has a boundary with cocircular edge pairs consisting of one inward pointing and one outward pointing normal (Figure 4.1(d)). Therefore, in order to detect these objects, the algorithm handles all edges as if they had two normals, one pointing in the inward and one pointing in the outward direction (Figure 4.1(e)).

1.2. Finding interest points. The main problem involved in developing a working implementation of the algorithm described in chapter 3 is the determination of the symmetry set on a discrete grid of points. A number of methods have been proposed. Most of these involve either an iterative wave diffusion process [**6**] [**71**], or a convolution method [**71**] [**38**]. From another point of view, by noting the relationship of centres of cocircularity to the points in the symmetry set, finding the latter can be seen as being equivalent to the problem of finding centres of partial circles in an edge image. There are many circle finding methods described in the literature. Most of these are based on variations of the Hough transform [**32**] [**52**] [**2**] [**87**]. A method similar to convolution with annular operators [**38**] was chosen because it is most readily understood and also easily adaptable to computation of the extended multiscalar symmetry set. However, this method was extensively modified in the final implementation to make it more efficient and compatible with the target DSP hardware.



(a) A set of annular templates and (b) one of the templates divided into angular bins. Within each angular bin, only edges with normal orientations between θ_1 and θ_2 are considered. As well, each angular bin has an orientation ϕ associated with it.

FIGURE 4.2. Annular templates

This results in a technique that resembles wave diffusion methods used in conjunction with a series of Hough-like accumulator arrays.

The convolution approach to finding the symmetry set involves convolving the edge intensity image with a set of annular masks of differing radii. These radii are chosen to cover the full range of extent of the symmetry operator (see Figure 4.2(a)). This produces a high output when many edge elements are located within the annular regions. Adapting this approach to the computation of the extended symmetry set requires the orientation of the edge normals to be considered as well. To accomplish this, each annulus is further subdivided into a discrete number of angular bins (Figure 4.2(b)). Within each bin, only edge segments which have a normal pointing towards the centre of the annulus are considered. However, because of discretization, there is a range of allowable edge normal orientations which are considered to be inward pointing for any given angular bin. For example, any edge with a normal between θ_1 and θ_2 for the enlarged angular bin of Figure 4.2(b) is considered to be inward pointing for that bin. As well, each angular bin also has an orientation ϕ associated with it, which is defined as the median angle of all inward pointing edge normal orientations for that bin. These annular templates are centred at all points of the image. The edges that fall within the annular masks, with normals oriented toward the annular centres, are used to compute the symmetry at the central point. This leads to a computation of symmetry at every point, and over every scale covered by the annular templates.

Such an approach requires some modifications to the algorithm described by equations(1) – (5). In fact, most of these modifications do not involve changes to the actual equations themselves, but rather involve slight changes to the definitions of the parameters used in the equations. These parameter modifications are mainly concerned with treating the set of all edges which fall within a single angular bin together, rather than the previous strategy in which each edge is considered independently. As such, the label λ is now used to refer to the *set* of inward pointing edge elements which fall within a single angular bin of a particular annular template. This is a natural extension of the previous use of λ which referred to a single, inward pointing edge element. Similarly, the definition of $\|\lambda\|$ is revised to refer to the maximum magnitude of all the edges which are members of λ. Finally, the definition of cocircularity is modified to include these angular bins. Two angular bins are said to be cocircular if they belong to the same annular template, and each contains at least one edge with an inward pointing normal.

Based on these modifications, the set $\Gamma_r(p)$ can now be defined as the set of all pairs of cocircular angular bins belonging to a particular annular template with radius of cocircularity r, centred on the point p. Therefore, given the sets λ_i and λ_j corresponding to the inward pointing edges falling within two angular bins, equation(1) now becomes

$$(7) \qquad \Gamma_r(p) = \left\{ (i,j) \;\middle|\; \begin{array}{l} \lambda_i \neq \{\emptyset\} \text{ and } \lambda_j \neq \{\emptyset\} \\ \lambda_i \text{ and } \lambda_j \text{ correspond to different angular bins of the same} \\ \qquad \text{annular template, } r, \text{ centred on point } p \end{array} \right.$$

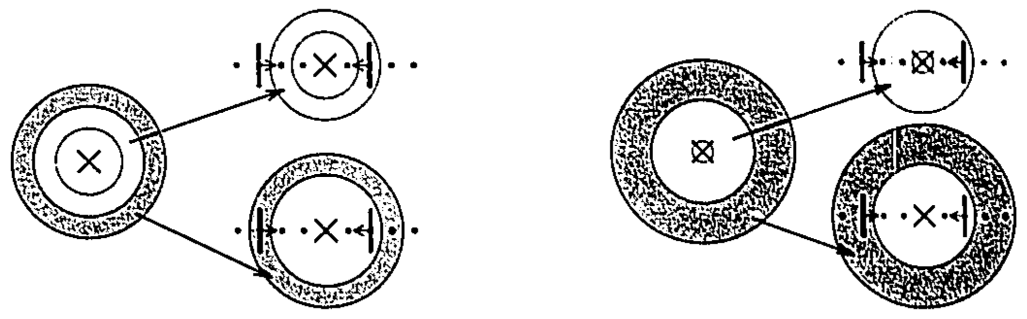
In this way, all the angular bins which are members of $\Gamma_r(p)$ contain cocircular tangents whose centre of cocircularity is the point p.

From this we determine a symmetry magnitude, $S_{r,\psi}(p)$, at each point p, for each symmetry orientation, ψ, and radius, r. These values are combined to produce the extended symmetry set, $S_\psi(p)$, which is finally used to compute an interest value, $I(p)$, for every point. The computations can be performed by direct application of equations(2) – (6) from chapter 3. However, the variables ϕ_i and ϕ_j now refer to the orientations associated with each contributing angular bin, rather than the orientations of individual edges. Also $\Gamma_{r,\psi}(p)$ now refers to the subset of all pairs of angular bins in $\Gamma_r(p)$ contributing to the symmetry orientation, ψ. The equations are repeated below for convenience:

$$(8) \qquad \psi(\lambda_i, \lambda_j) = \frac{\phi_i + \phi_j}{2}$$

$$(9) \qquad S_{r,\psi}(p) = \sum_{\lambda_i, \lambda_j \in \Gamma_{r,\psi}(p)} \|\lambda_i\| \, \|\lambda_j\| \, (\sin \varphi/2)^{w_1}$$

$$(10) \qquad \varphi = |\phi_i - \phi_j|$$



In this figure, × marks the annular centres and • marks pixel positions. (a) If the width of each annulus is too narrow the centre of two cocircular edges can be missed. (b) The centre of two cocircular edges can also be missed for wider annuli, if successive annuli do not overlap.

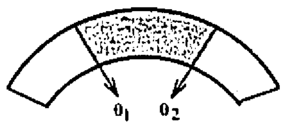
FIGURE 4.3. Effect of pixelization on annular selection

$$S_{\psi}(p) = \max_{r=0}^{r_{max}} S_{r,\psi}(p) \tag{11}$$

$$I(p) = \sum_{\psi_i,\psi_j} S_{\psi_i(p)} S_{\psi_j(p)} (\sin(\psi_i - \psi_j))^{w_2} \tag{12}$$

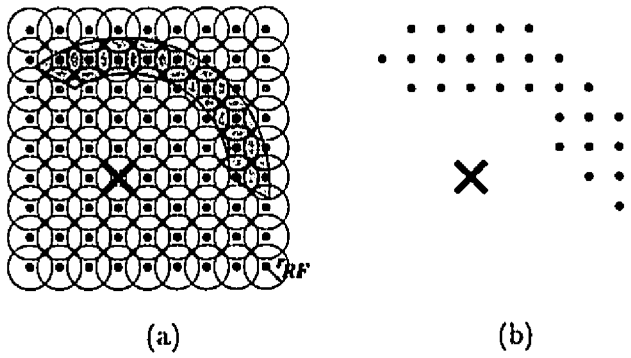
1.2.1. *Defining the annular templates.* Each annular template is defined by an annular width, which is the difference between its minimum and maximum radius. The wider each annular template is, the less localized the resulting lines of symmetry will be. However, due to pixelization, if the templates are too narrow the central point of two edges could be completely missed (Figure 4.3(a)). For this reason an annular width of two pixels is used. As well, in order to cover the full extent of the image with no gaps, the minimum radius of one annulus must be less than or equal to the maximum radius of the next smaller annulus. However, some pairs of cocircular edges will not be detected if the minimum radius of one annulus is equal to the maximum radius of the next smaller one (Figure 4.3(b)). As well, this overlap should be minimized to limit duplication of lines of symmetry at different scales. For this reason an overlap of one pixel is used between successive annular templates.

Each annular template is also divided into angular bins. The selection of the number of bins is motivated by two conflicting requirements. Setting the number of bins too small results in large angular bins, which leads to large clusters of interest points in the output. However, setting this value too large causes the algorithm to be highly selective in orientation tuning. This results in many disjointed interest regions in the output, and also increases the algorithm's sensitivity to noise in the edge orientations. This problem can be partially solved by introducing some overlap between neighbouring angular bins.



Without angular bin overlap, edges within a certain range of orientations (θ_1 to θ_2) would be searched for only in the dark shaded region. With overlap into the neighbouring bins, edges with the same range of orientations are searched for in the lightly shaded regions as well.

FIGURE 4.4. Effect of angular bin overlap



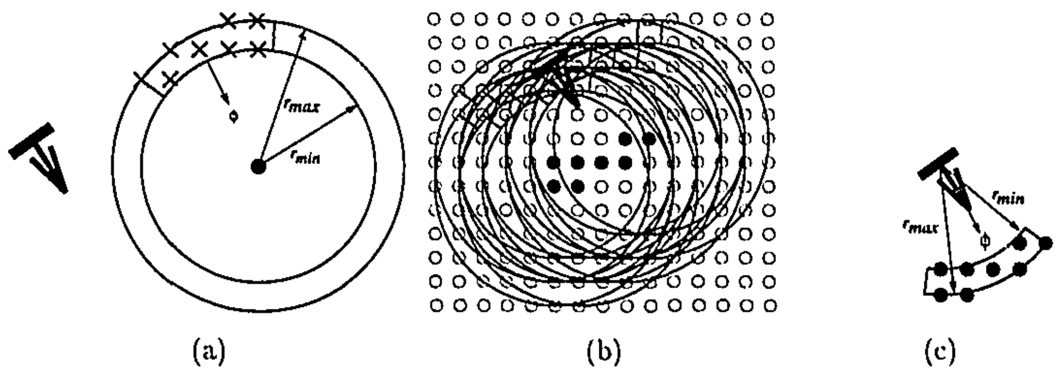
(a) Each pixel (•) affects a receptive field around it of radius r_{RF}. A pixel whose receptive field overlaps the template (shaded region) is part of the template mask. (b) The pixels that make up the template in (a). The × represents the centre of the annulus from which this mask was derived.

FIGURE 4.5. Calculating the template masks

This allows a large number of angular bins to be selected, with each bin having a wide enough extent to allow for some misalignment in edge normals (see Figure 4.4). However, the ultimate choice of the number of bins and the angular bin overlap is not critical, as will be shown in section 3.2.2.

Given a particular choice of template parameters, the shape of the annular template mask can be determined. In practice, a separate template mask is computed for each angular bin. This is accomplished by overlaying the template on a rectangular grid of pixels (Figure 4.5(a)). By associating a receptive field (RF) of radius r_{RF} around each pixel, the template mask can be obtained by calculating which pixel's receptive fields overlap the region of the angular bin (Figure 4.5(b)).

1.2.2. *Inverting the annular templates.* In actual practice, a more efficient variant of the annular templates is used. This alternative will be shown to be equivalent to annular templates, yet results in an implementation with a much smaller search space. Furthermore, this will be the basis for the performance improvements described in the real-time implementation of chapter 5. This method, though derived from arguments related to the annular



(a) For a particular edge and annular template, the angular bin for which the edge is deemed to be inward pointing can be determined (shaded region). The pixels corresponding to this region are shown as × , and the centre of the annular template is marked with a • . (b) The annular template is centred at all points that place the edge within the shaded angular bin. The points marked • correspond to all the possible centres for this edge. (c) These points make up a new *inverted* template, which maps the edge to possible centres of cocircularity. Notice that the inverted template is simply a mirrored version of the original one.

FIGURE 4.6. Inverting the annular template

implementation, turns out to be quite similar to the wave propagation implementations, which further highlights the similarity of these two methods.

The annular implementation described previously involves a search around each output point for all edges which can contribute to the symmetry at that point. This process can also be driven in reverse by determining the possible annular centres that can be affected by each edge. This is accomplished by *inverting* the annular templates, indicating the location of possible annular centres for a given edge, and storing the edge contributions in a set of Hough-like accumulator arrays.

To see how these inverted templates are computed and applied, it is best to begin by referring to a particular example. Given the edge and annular template of Figure 4.6(a), the angular bin for which the edge is considered inward-pointing can be determined (shaded region of Figure 4.6(a)). By centering the annular template at all points that place the edge within this angular bin (Figure 4.6(b)), all possible centres affected by this edge for the given annular radii ($r_c = r_{min}$ to r_{max}) can be found. This produces a new inverted annular template (Figure 4.6(c)), whose points correspond to the offsets of possible centres of cocircularity from the edge position.

This procedure is repeated for all edge orientations, resulting in a set of inverted annular templates, each one mapping a particular edge orientation to possible centres of cocircularity. Furthermore, in the same way that each angular bin has a corresponding orientation, each inverted annular template has an associated contribution direction, which describes the position of the template with respect to the contributing edge. In fact, the orientation of the inverted templates with respect to an edge is equivalent to the orientation of the original angular bin (compare ϕ in Figure 4.6(a) and (c)). Finally, it should be noted that the points corresponding to the inverted templates are simply mirrored versions of the points of the angular bin from which they were derived.

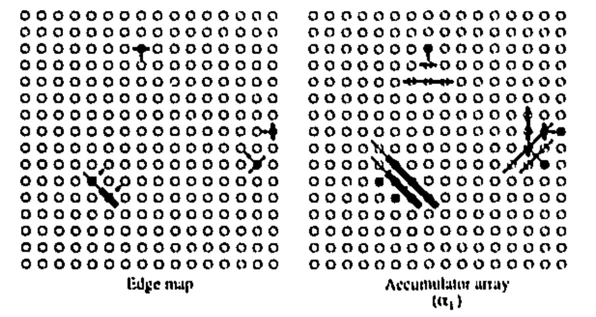
The inverted template is used to determine all the possible centres of cocircularity, within the range r_{min} to r_{max}, for each edge in the input image. This information is stored in an accumulator array, as a separate value for each contribution direction at each point. In accordance with the definition of edge magnitude used in this chapter[1], the value stored in the array is the maximal magnitude of all edges contributing to a point from a particular angular bin. This is accomplished, by traversing the entire input image, and for each edge, applying the inverted annular template corresponding to it's orientation[2]. Application of the inverted templates involves overlaying the templates on the accumulator array at a position corresponding to the edge position in the original image. At the points of the accumulator array corresponding to the template offset points (points marked • in Figure 4.6(c)), the edge magnitude is compared to the value previously stored in the accumulator array for the corresponding contribution direction. If the edge magnitude is greater, its value replaces the previous value in the accumulator array. An example of this procedure is shown in Figure 4.7(a).

In the same way, inverted templates can be determined for all other annular templates. This produces a series of inverted templates for each edge orientation, each one corresponding to the particular radius of cocircularity of the annular template from which it was derived. Each of these radial ranges is treated separately, and the templates corresponding to each range are applied sequentially to the image, affecting different accumulator arrays. Therefore, for n original annular templates, n accumulator arrays, $\alpha_1, \alpha_2, ..., \alpha_n$, are required, each corresponding to centres of cocircularity at a particular range of radii (Figure 4.7(b)).

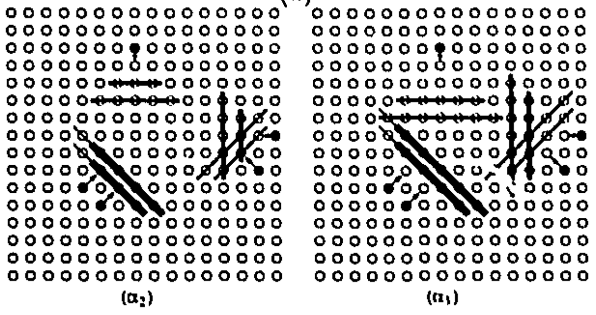
It should be noted that after the inverted templates have been applied to every edge in the image, any point in the accumulator arrays with more than one non-zero contribution direction is a centre of cocircularity. These points have contributions from exactly the same

[1]To review, edge magnitude $\|\lambda\|$ was previously defined as the maximum magnitude of all the inward pointing edges lying within a single angular bin.

[2]As was discussed previously, each edge actually has two normal orientations, one inward and one outward pointing. Because of this, two templates are actually applied at each edge location, corresponding to the two normal orientations. However, the application of one template does not affect the other, and so, in order to simplify the discussion, only the application of a single template is considered.



(a)



(b)

(a) The edge map shows all edge orientations, with the width of the edge being proportional to the edge magnitude. The accumulator array, α_1, shows the results of applying the templates at each edge position. Notice that the templates of the two edges in the bottom left of the image overlap with the same orientation. At the points of overlap, the value of the edge with the highest magnitude is stored in the accumulator array. Also notice that the two edges at the right of the image result in overlapping outputs at different orientations in the accumulator array. These overlapping points correspond to centres of cocircularity of the two edges. (b) The accumulator arrays for different template radii.

FIGURE 4.7. Applying inverted annular templates to a simple image

edges as the centres of cocircularity that would be obtained using the non-inverted annular template method. In fact, the edges which contribute to an accumulator array at a point p and orientation ϕ are exactly the same edges which would fall in the same angular bin of a particular annular template. This set of edges was labeled λ in the discussion of the application of the non-inverted templates. Furthermore, $||\lambda||$ was defined as the maximum of all these edge magnitudes, which is equivalent to the value stored in the accumulator arrays for each contributing orientation.

Based on this, we can develop a new definition of the set $\Gamma_r(p)$, which is equivalent to equation(1). We refer to the value at a particular point p and orientation ϕ of an accumulator array as $\alpha(p,\phi)$. The set $\Gamma_r(p)$ is then defined as

$$\Gamma_r(p) = \left\{ (i,j) \;\middle|\; \begin{array}{l} \alpha(p,i) \neq 0 \text{ and } \alpha(p,j) \neq 0 \\ i \neq j \end{array} \right. \tag{13}$$

From here, the application of the algorithm is identical to the one described previously, and the determination of interest values is computed by applying equations(2) – (5). As was the case for the annular implementation, ϕ_i and ϕ_j refer to the orientations of the contributing template, rather than an individual edge, and $\Gamma_{r,\psi}(p)$ refers to all the elements in $\Gamma_r(p)$ with symmetry orientation ψ.

It can be seen that the inverted template procedure is equivalent to the application of annular templates described in section 1.2. However, a number of advantages are gained by adopting this approach. Application of the annular templates requires a series of operations to be performed at every point of all the angular bins making up the annulus. In contrast, application of the inverted templates only requires a computation for the points corresponding to the angular bin associated with the contributing edge orientation. Therefore, given n angular bins, application of the annular templates involves n times more points to be processed. However, this may not be significant if the annular implementation requires significantly less operations per point. At every point in the template, application of the annular method requires a memory access to determine the value of the edge at that point, and a comparison, to determine the maximal edge magnitude for the angular bin. Application of the inverted templates requires a similar series of operations. At every point in the template a memory access is needed to determine the value stored in the accumulator array, which is compared to the magnitude of the contributing edge. Therefore, for every point it is centred on, the annular method must perform n times more processing than the inverted template method. Finally, the number of points that each template must be applied to should be examined. The annular templates need to be centred at *every* point in the image. In contrast, the inverted templates are only applied to locations with an edge. Under the worst circumstances an edge is found at every point in the image, in which case the inverted template would be applied to the same number of points as the annular template. Therefore, in the worst case, the inverted template method involves an n times decrease in computation over the annular method. As well, there is a possibility of significantly increasing this improvement by reducing the number of edges, and this will be exploited in the discussion of the real-time implementation.

1.2.3. *Processing flow.* Figure 4.8 shows the full processing path for computing interest points. Each inward pointing edge affects one of the accumulator arrays, $\alpha_1(p)$, $\alpha_2(p)$, ..., $\alpha_n(p)$, based on the distance of the edge to the point p. The values stored in these accumulator arrays is the maximum edge magnitude, at each angular bin orientation, of all the edges within the accumulator array's range of radii. Application of equation(3) then gives a symmetry magnitude at each radius and orientation. These values are combined over all radii through equation(5) to give a single symmetry magnitude for each orientation at every point. Finally, these magnitudes are combined over all the symmetry orientations using equation(6), resulting in a magnitude of interest at every point in the image.

1.3. Smoothing and determining local maxima. As discussed in the previous chapter, smoothing is accomplished by convolving the input image with a Gaussian mask. This would normally require 2D processing for most convolution masks. However, the Gaussian belongs to a special class of *separable* functions, which means that it can be separated into two 1D Gaussians, one horizontal and one vertical. Since convolution obeys the commutative law, convolving the input image with one of these 1D Gaussians, then convolving the result of this with the other 1D Gaussian gives the same result as a 2D convolution with the original 2D Gaussian mask. However, a 2D convolution of an $n \times n$ Gaussian mask with an $N \times N$ image requires n^2N^2 multiplications and additions. In contrast, the same results can be achieved using two 1D convolutions with $2nN^2$ multiplications and additions. This is clearly a computational savings, which increases as larger convolution masks are used.

The final output of the algorithm are the local maxima of the smoothed image. Points are defined as local maxima if they have a value greater than any other point within a specified distance from themselves. However, the determination of local maxima follows a smoothing operation which causes high frequency changes in the interest map to be removed. Thus maximal points which are close to each other tend to be grouped into large clusters with a single maximum. Because of this, it is only necessary to examine the nearest neighbours of a point to determine if it is a local maxima.

2. Implementation on Peripheral Images

One of the major goals in developing this algorithm is its implementation on non-uniformly mapped peripheral images. This arrangement mirrors the spatial characteristics of the primate retina, where resolution becomes coarser as visual angle increases [72] [73] [7]. Though this requires some changes to the implementation described above, these are relatively minor and the algorithm remains quite similar. As such, the discussion below describes the peripheral implementation in terms of the way that it differs from the foveal implementation described previously.

2.1. Log-polar mapping. The variable resolution of the periphery is mapped onto log-polar coordinate axes, permitting the full periphery to be represented in a rectangular

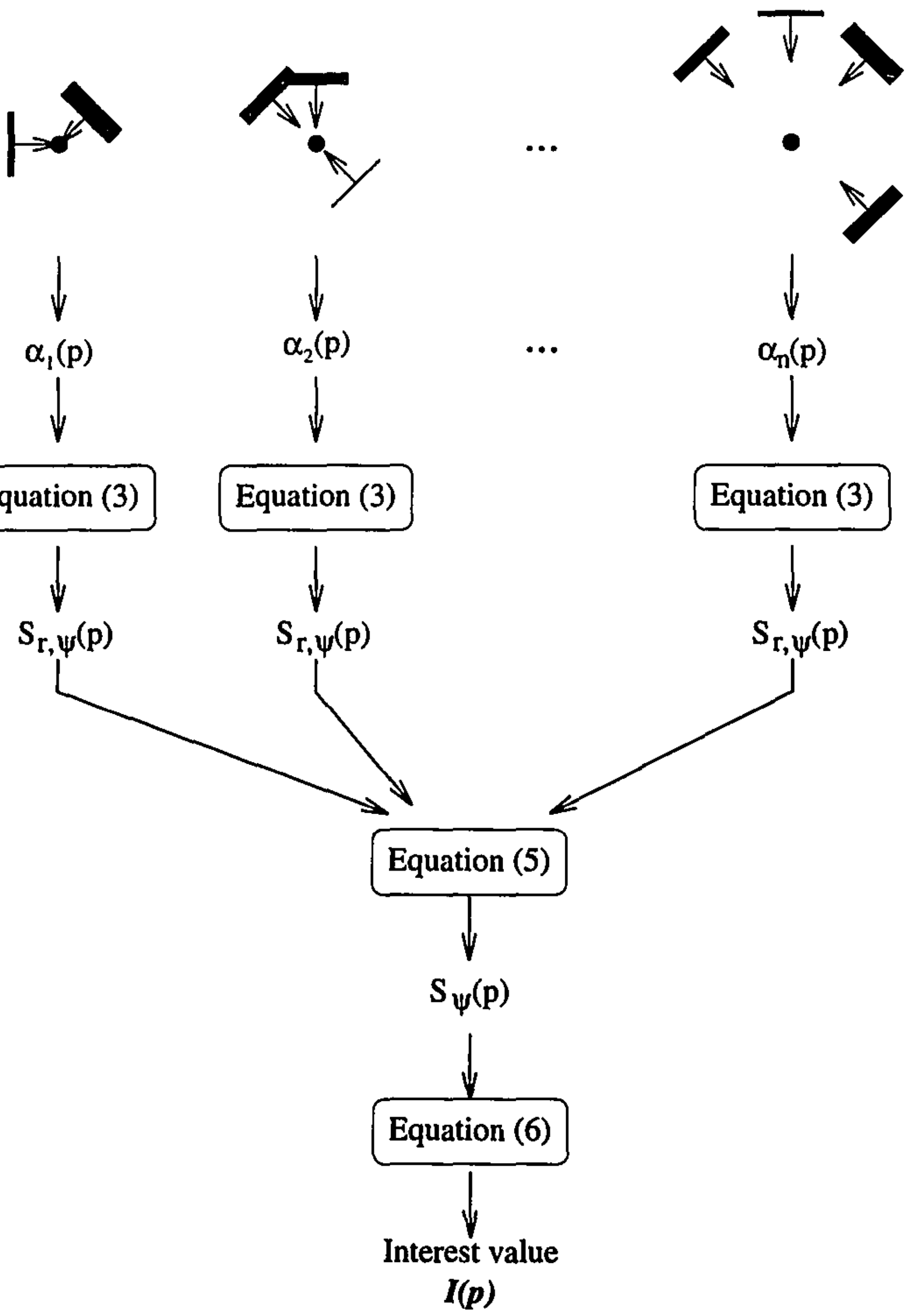
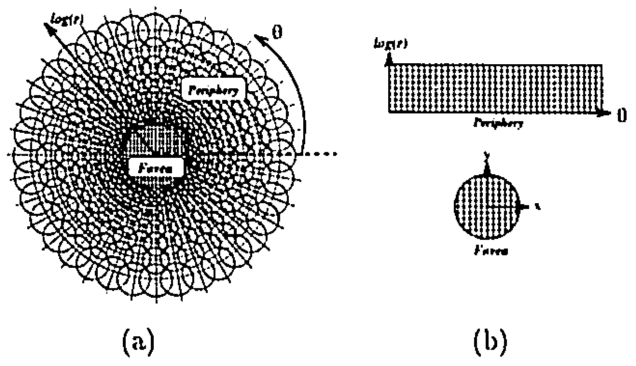


FIGURE 4.8. Processing flow for computing interest points

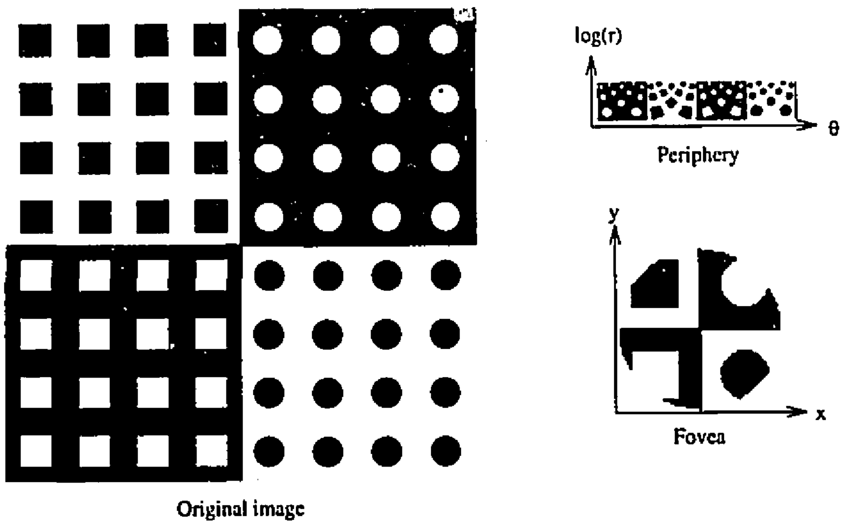


(a) The fovea, mapped onto a rectangular grid, is surrounded by the periphery, mapped onto a non-uniform, log-polar grid. The crossings of the grid lines mark the centres of the pixels. The circles delineate the receptive field (RF) of each peripheral pixel. Within each RF a weighted average of all points falling within it's receptive field is computed. (b) The fovea and periphery can be mapped to two separate, rectilinear grids.

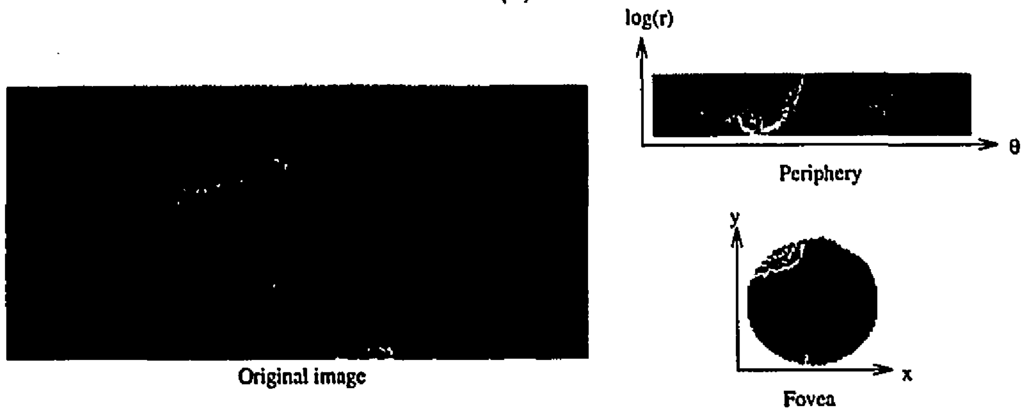
FIGURE 4.9. Log-polar mapping

image (Figure 4.9) (see [7] for a full description of this system). This mapping provides a number of advantages from the perspective of computational vision. First, the amount of data which must be stored as a result of this mapping is greatly reduced when compared to the original rectangular input image. However, important image details can still be examined at high resolution in the fovea and inner peripheral regions. Second, looming and rotating objects appear as simple translations in this mapping. Some examples of the kind of output produced by this mapping are shown in Figure 4.10. An important point which should be noted is that the log-polar peripheral image has no boundary in the θ direction. This axis wraps around, connecting the left edge of the periphery with the right edge. This wraparound must be considered when implementing the interest algorithm on this coordinate axis.

2.2. Changes to the foveal implementation. The first difference between the foveal and peripheral implementation is at the edge detection stage. Since the Sobel edge detector was originally designed for a rectangular coordinate grid, its implementation in the log-polar domain must be examined. The mapping from rectangular input image to log-polar peripheral image still maintains nearest neighbour connectivity. This means that points that are neighbours in the input image will map to points that are neighbours in the peripheral image. As well, linearity is approximately preserved in a local domain. Therefore, since the Sobel operator only relies on nearest neighbour interactions (it uses a 3x3 pixel mask) it can be applied in the log-polar domain. There are, however, a number of differences that must be considered in interpreting the output. First, since the log-polar



(a)



(b)

(a) Log-polar mapping applied to a simple test image. (b) The same mapping applied to a complex, real-world scene.

FIGURE 4.10. Images mapped in log-polar coordinates

domain is obviously non-uniform, the scale of the edges found by the Sobel operator in the log-polar domain will vary with position. However, this is exactly what is required, since the scale of salient objects also increases with their position in the log (r) direction of the periphery. Second, the orientations found by the edge detector are relative to the θ and log (r) axes. This is also desirable since the application of templates and the determination of lines of symmetry will also occur along these axes. Finally, the algorithm must be modified slightly to account for the wraparound of the θ axis. This is accomplished simply by allowing the edge detection operator to wrap around this axis, treating the image as if the points at the left boundary continued past the right boundary points.

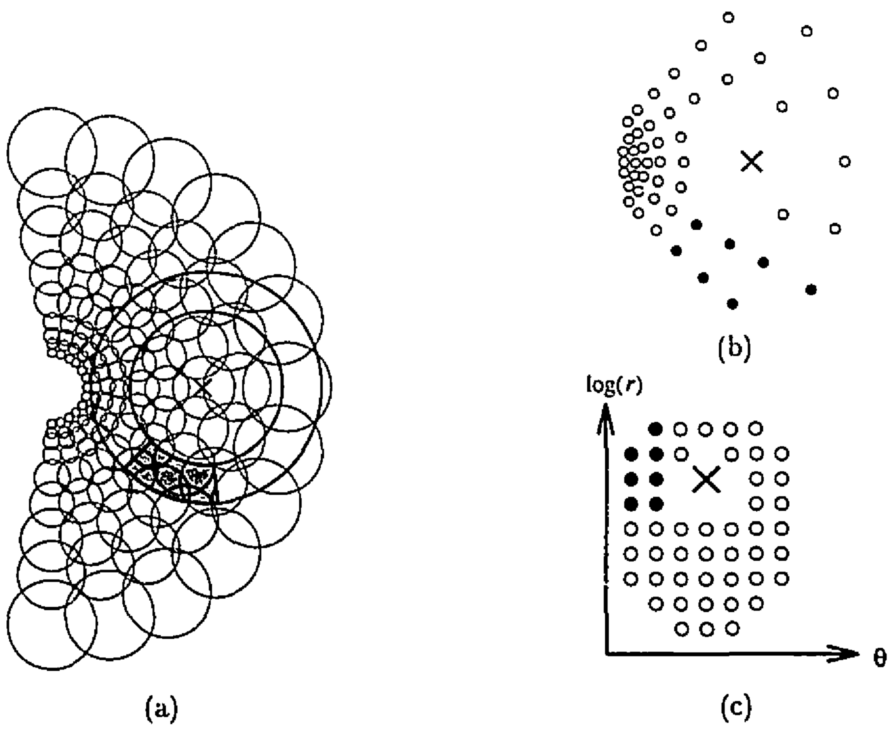
The same annular template parameters are used for the peripheral image as for the foveal image. However, because of the difference in mapping functions, the templates must be computed differently. As in the rectilinear case, the template bins are overlaid on the grid of pixels. However in this case the grid is not uniformly distributed (Figure 4.11(a)). Again, each pixel whose receptive field overlaps the template becomes part of the template mask (Figure 4.11(b)), which is then transformed into log-polar coordinates (Figure 4.11(c)). Furthermore, these templates can be inverted in a manner similar to the inversion of the rectangular coordinate, foveal templates. This is done by centering an annulus on all points that place an edge within the appropriate angular bin (Figure 4.12(a)). This gives a set of templates which map edge position and orientation to possible centres of cocircularity in the log-polar domain (Figure 4.12(c)). In this way, the inverted template algorithm can be run on the peripheral, log-polar image in the same way as was done on the foveal image, with the only difference being the specific pixels belonging to each template mask.

The second modification required to implement the algorithm in the log-polar domain involves image wrap around. In log-polar coordinates there is no real boundary at the ends of the θ-axis. Instead, this axis wraps around, connecting the left edge of the image with the right edge. This is dealt with by simply allowing the templates to wrap around the θ axis as well. For example, a template mask that crosses the right edge of an image continues from the left (Figure 4.13).

Finally, there is one additional side effect of the log-polar implementation described above which should be noted. As the same template is applied higher in the log (r) direction it's spatial extent increases (Figure 4.14). However, this is a desirable effect since spatial resolution becomes coarser in the outer periphery, requiring the spatial extent of salient objects to similarly increase in this region.

3. Evaluation and Test Results

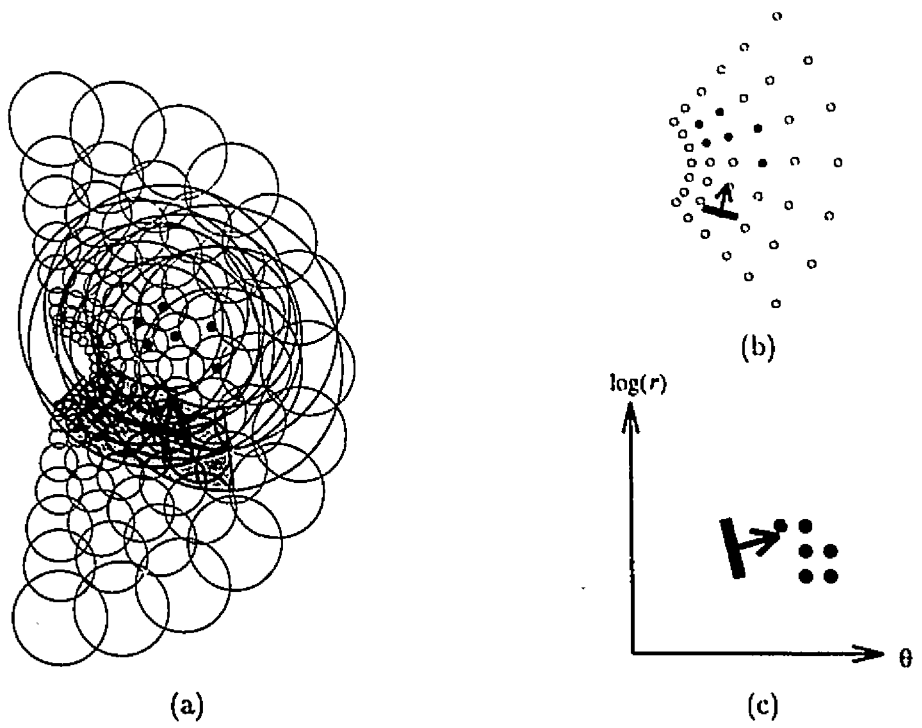
This section examines the performance of the described interest point algorithm. There are basically two parts to the analysis. First, the interest point model will be verified and results with real-world images will be examined. All the results produced in the first section will be based on the same set of parameters. The second section will examine how changes to these parameters affect the output, and will justify the choice of values used to produce the results in the first section.



The peripheral pixels corresponding to a particular annular template are found by (a) centering the annulus on the centre of a foveal receptive field pixel, (b) determining which peripheral receptive fields intersect the annular template, and finally (c) plotting these points on a log-polar coordinate axis. In the example shown, circles (○ and ●) represent the peripheral pixels which intersect the annulus, with the filled circles (●) corresponding to the specific pixels which belong to the shaded angular bin region of the annulus.

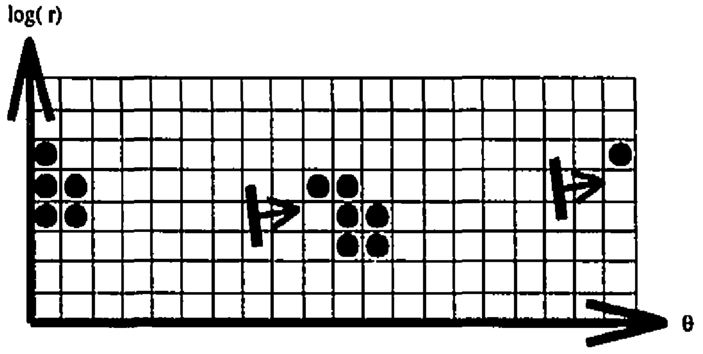
FIGURE 4.11. Determining the annulus in log-polar coordinates

3.1. Evaluating the outputs. The processing path for determining the interest points in an image is shown in Figure 4.15. As was described previously, edges in the input image are first detected, resulting in an edge magnitude and edge orientation map. These images are then processed by the interest algorithm, producing a raw interest map. The map is then smoothed, and finally the local maxima of the smoothed image are found. To evaluate the results, the points of local maxima are shown as white crosses superimposed on the original image, with the size of the cross proportional to the interest magnitude at the point. As well, to ensure that the white crosses are visible, the intensity of the original image is reduced in these output images. Finally, in all of these figures, the size of the largest annular template is shown by the circle at the bottom right corner of each figure. The same maximum template extent, r_{max}, was used for the outputs of all the real-world images in this chapter. Any variation in the size of the indicated circle from one figure to another is entirely due to different image scaling in printing.



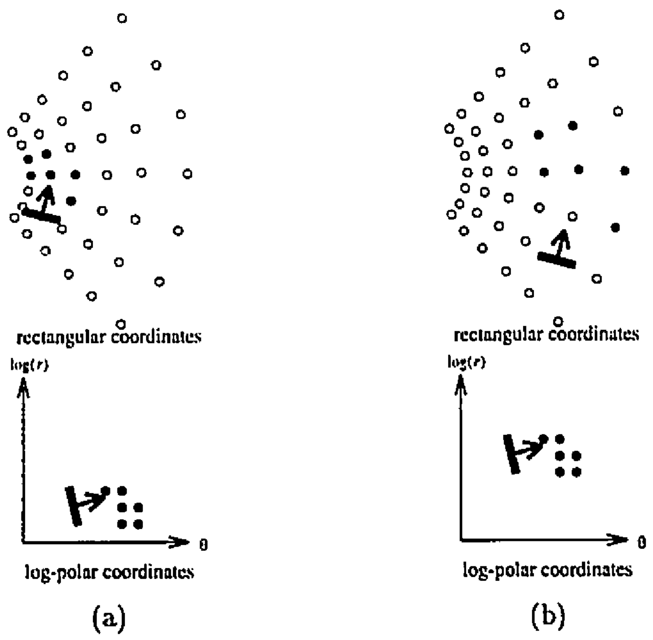
(a) An annular template is centered on all RF centres that place the edge within the desired angular bin. These RF centres, marked with a •, correspond to all possible annular centres for this edge. (b) The points are replotted with the RF's and annuli removed. The open circles (o) mark the locations of all RF centres and the filled circles (•) mark the pixels corresponding to the edge's *inverted* template. (c) The same edge, plotted in the log-polar domain. The template points (•) mark the offset, in log-polar coordinates, of the edge to it's possible centres of cocircularity.

FIGURE 4.12. Inverting templates in the log-polar domain



Because the theta axis in the log-polar coordinate system wraps around, any template applied near the edge of the theta-axis is wrapped around, affecting points at the other end of the image.

FIGURE 4.13. Template wrap around



(a) A template applied to the log-polar grid, and the associated points in rectangular coordinates. (b) The same template applied higher in the log(r) direction affects the same number of points, but these points now cover a much wider area when mapped back to rectangular coordinates.

FIGURE 4.14. Increasing template extent as a function of log(r)

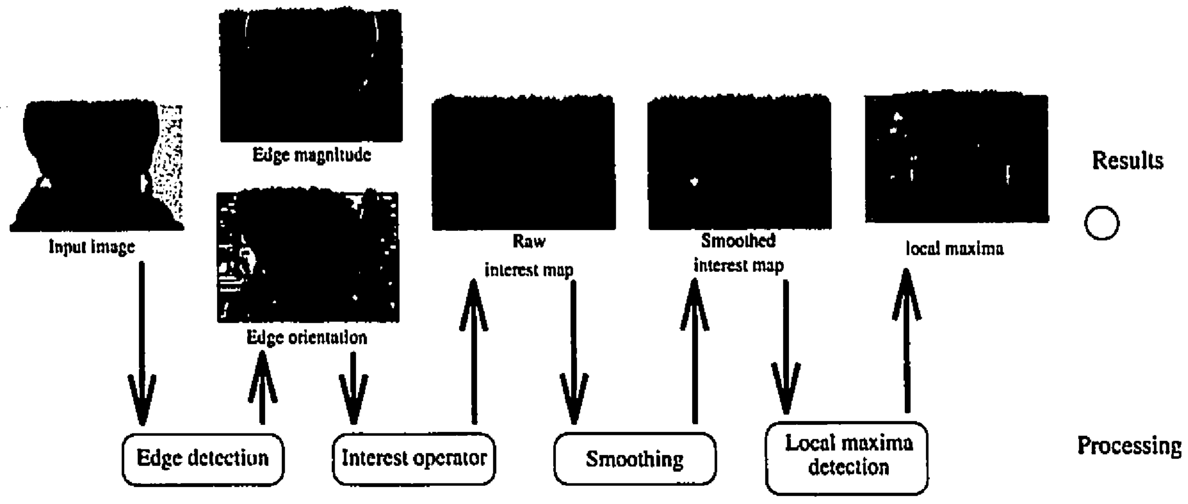


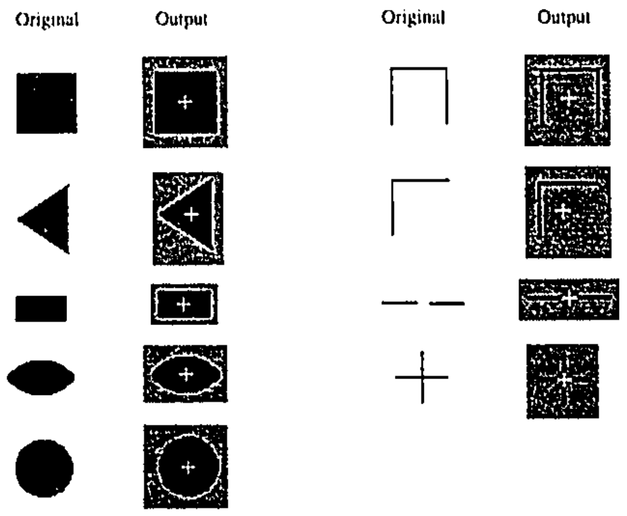
FIGURE 4.15. Processing path

As was mentioned above, all the results in this section use the same set of parameters. This provides convincing evidence that there is a general set of parameters which allows the algorithm to be effective regardless of input stimulus. The justification for the selection of these parameters will be deferred until the next section. For now, it is sufficient to present the values used. Each annular template is divided into 32 angular bins. The extent of each of these bins overlaps its two neighbouring bins by 25%. As well, each annulus is two pixels wide, overlapping the next smaller and larger annuli by 1 pixel each. The smallest annular radius is 1 pixel and the largest annulus has a radius of 10 pixels. Finally, the two weighting parameters w_1 and w_2 (see equations(3) and (6)) are each set to 5, and Gaussian smoothing is applied with $\sigma = 4$ pixels.

3.1.1. *Verifying the model.* To begin evaluating the algorithm's performance, it is useful to test it on simple geometric shapes (Figure 4.16). Comparing these results with the experiments of Kaufman and Richards [37] presented in Figure 2.1, we observe that there is strong agreement between the algorithm output and the psychophysical data. This verifies both the interest point model and the implementation. Of course, here we assume that the complete object is viewed entirely by the fovea. It is also interesting to note that the algorithm is effective at finding the centres of open as well as closed forms. This is due, in part, to quantization effects in the edge detection, which interprets corners as diagonal edges and line ends as orthogonal edges. This creates more inward pointing edge orientations and allows for intersections to occur even when the theoretical lines of symmetry for these shapes (see Figure 3.1) have no intersections. As can be seen, the resulting interest points correspond exactly to the results of Kaufman and Richards for open forms. Finally it should again be remembered that all the results of Figure 4.16 were produced with the same set of parameters, showing that special tuning for different shapes is not necessary.

3.1.2. *Results on real images.* The interest point model, and the psychophysical experiments from which it was derived, are mainly based on evidence from observers viewing simple geometric shapes. Therefore, it is not apparent how well this model will perform on natural, real-world scenes. As well, because much of the psychophysical evidence is based on simple shapes, the desired interest points for these stimuli are clearly defined. When applied to more complex images, the evaluation of what constitutes an interest point is much more difficult. In fact, it is not clear if a quantitative method for evaluating the algorithm's performance on such scenes can be developed. Because of this, the results of the algorithm will be evaluated qualitatively, judging the outputs against an intuitive notion of what is interesting in the image.

Figure 4.17 shows the results of the algorithm applied to a variety of real-world scenes in rectangular coordinates. For some images, where many interest points are found, a second

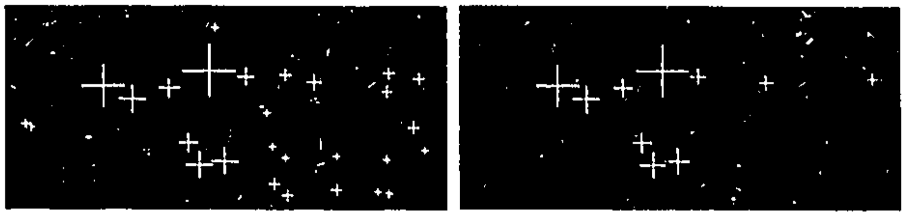


The crosses indicate the interest points obtained by applying the algorithm to a variety of shapes. These outputs can be compared directly to the psychophysical results of Kaufman and Richards [37] presented in Figure 2.1.

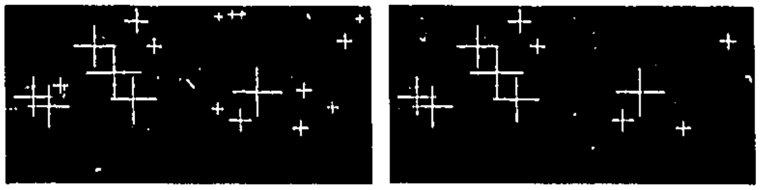
FIGURE 4.16. Algorithm output for simple shapes

image is presented, showing only the top ten interest points. Beginning with Figure 4.17(a), it can be seen that many weak interest points are obtained, due to the almost random nature of the edges found in the trees. However, the top ten interest points clearly follow the path of the wall, with the highest among these corresponding to the three towers along the wall. These are clearly the most interesting objects in the scene. Many weak interest points are also found in the next image (Figure 4.17(b)). However, the top ten interest points fall on many of the most important areas in the image. These include the license plate, the stop sign, and the wheels. As well, some other possible interest points, corresponding to windows of homes in the background, are also found. Similar results are seen in Figure 4.17(c), where the towers and the boats in the foreground are highlighted. In Figure 4.17(d) most of the points occur within the two planes, with the two highest points corresponding to the centre of each of the two planes. Similarly, the three highest points in Figure 4.17(e) correspond to the three towers in the image. Finally, Figure 4.17(f) highlights the obvious interesting region in the image, a fragment of comet Shoemaker-Levy crashing into Jupiter.

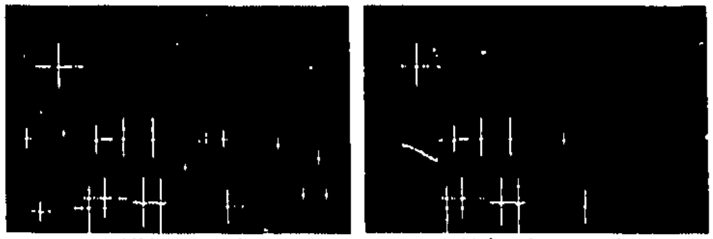
Figure 4.18 shows the result of applying the interest operator to an important subset of objects, faces, for which the algorithm is particularly well-suited (see [64] for a face recognition application of a similar algorithm). When facial features such as the eyes and nose are at the scale of the operator (Figure 4.18(a), (b) and (c)), they are correctly found. When the entire face is at the scale of the operator (Figure 4.18(d)), the centre of the


All interest points   10 highest interest points

(a)


All interest points   10 highest interest points

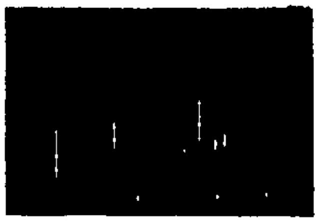
(b)

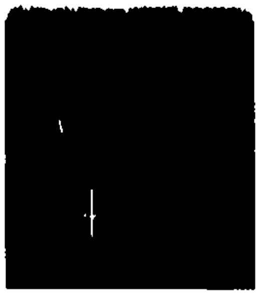

All interest points   10 highest interest points

(c)

All interest points   10 highest interest points

(d)


(e)


(f)

The output of the algorithm for a variety of real-world scenes. The points marked by a white cross indicate the locations of the interest points, with the size of the cross proportional to the interest magnitude at that point. Where a large number of interest points are found, the top ten points are shown in a separate image.

FIGURE 4.17. Algorithm output for real-world scenes

face is found. Finally, Figure 4.19 shows how the operator performs on a group of faces with a very noisy background. Again, as was the case for many of the outdoor scenes, the random nature of the background results in many weak interest points. However, the top ten interest points mostly correspond to the faces of the wolves, which are the desired points in this scene.

Figure 4.20 demonstrates an important aspect of the algorithm. Since it is only sensitive to relative edge orientation, the algorithm is invariant to rotation in an image. As well, as long as the individual features of an object are large enough not to be smoothed together, the algorithm is also invariant to changes in scale. Thus the same interest points can be tracked as an object moves and rotates. In this case, notice that the points corresponding to the eyes and chin remain stable as the head changes position. This is very important for the ultimate use of this algorithm, that is to use a foveated sensor to keep interesting, moving objects at the center of gaze.

Finally, there are a few points worth mentioning in light of the results presented above. First, it is impossible to determine precisely what regions of an image will be interesting in a given situation, without the use of higher level knowledge. For instance, in Figure 4.17(b) the cars may be of interest in some cases, .nd in other situations the windows of the houses may be of interest. As well, since the algorithm is not involved in any recognition, but rather is sensitive to particular groupings of edge orientations in an image, not all the interest points always correspond to actual interesting objects. This can be seen, for example, in all the points found in the trees in Figure 4.19 or between the wings and body of the planes in Figure 4.17(d). In fact, these points are valid interest points according to the model, and correspond to background regions which happen to have a shape that the algorithm is sensitive to, though even in these cases the spurious interest points are often much weaker than the desired ones. However, the intent of this algorithm is only to present candidate object locations, greatly reducing the search space for a subsequent recognition task. Clearly this is accomplished. The interest points found represent a majority of the interesting regions in the scene. As well, even with the few extra noise points that are produced, the search space for a recognition task is greatly reduced. For example, even when all the noisy interest points in Figure 4.17(a) are considered, the algorithm reduces the search space for a recognition process to 28 points from the 25056 points in the original image.

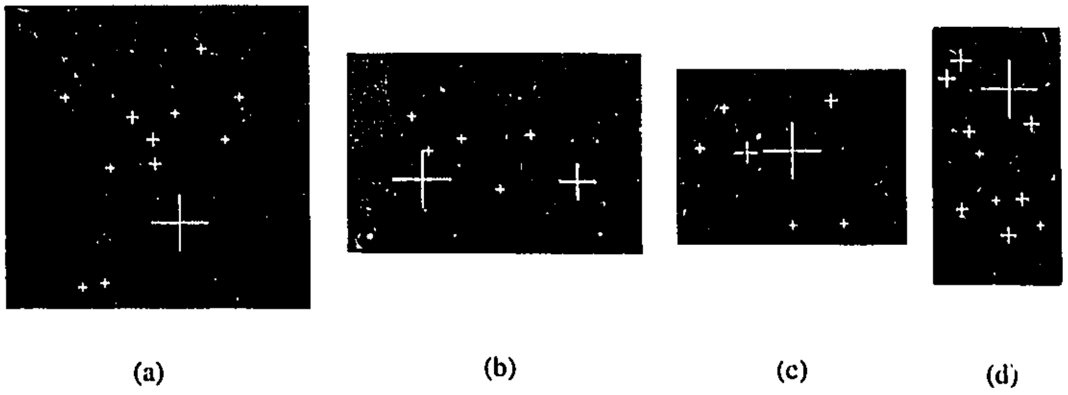
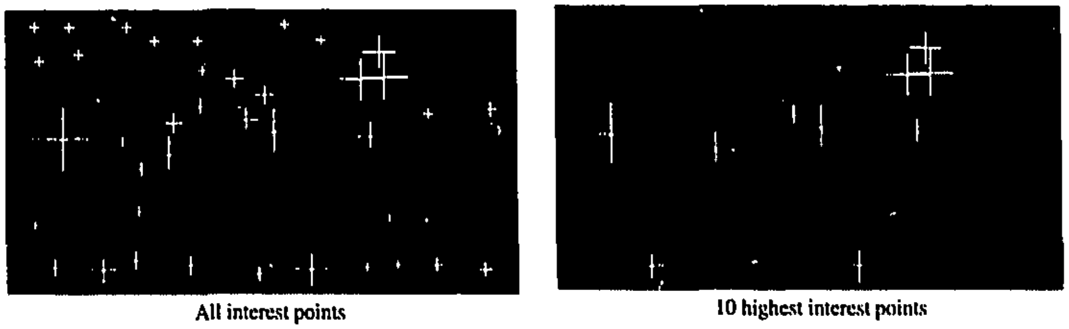
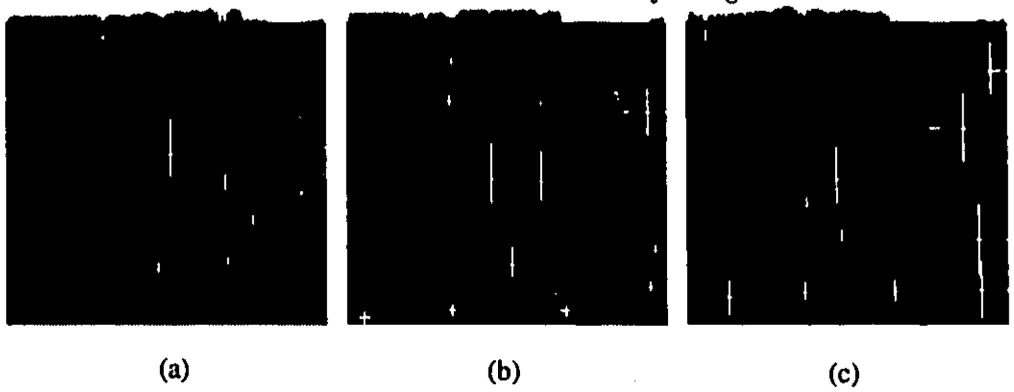


FIGURE 4.18. The algorithm applied to faces



When applied to objects on a complex background, many interest points are found. However, by examining only the major ones, the effectiveness of the algorithm can be seen to highlight the faces of each of the wolves.

FIGURE 4.19. Faces on a noisy background



The algorithm remains stable, tracking the same points, as the face is moved closer and farther, and as it rotates. These three images were obtained from the Vision and Modeling Group, MIT Media Lab.

FIGURE 4.20. Algorithm invariance to changes in orientation and scale

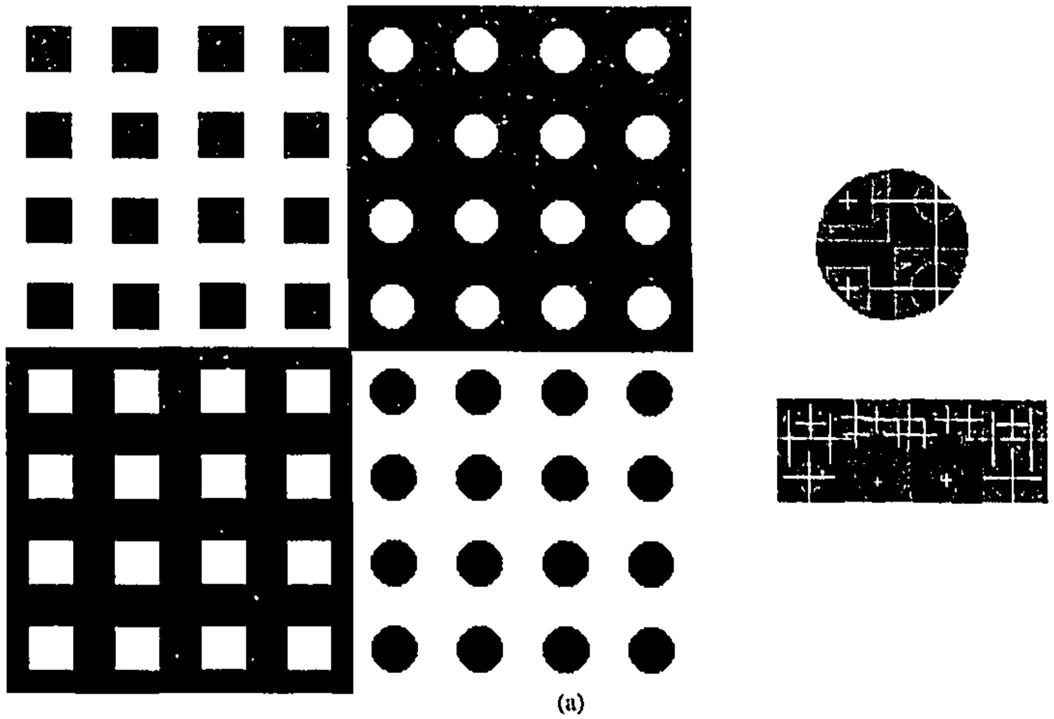
3.1.3. *Tests on log-polar images.* An important motivation in the development of this algorithm was its implementation in log-polar coordinates. The objective is to find interesting objects in the periphery, and then physically move the camera so that these objects fall within the fovea. Object recognition can then follow. As such, the ability of the algorithm to find interesting objects in the periphery must also be examined.

Figure 4.21(a) shows the output of the algorithm applied to a test image. Notice how it successfully finds the centres of both the squares and the circles in the fovea and near periphery, when $\log(r)$ is small. The shapes in the farther periphery map to much smaller forms on the log-polar axes, and so are smoothed together, forming single interest points for a group of objects.

The application of the algorithm on an outdoor scene is shown in Figure 4.21(b). It can be seen that the interest points found in this figure are in close agreement to those found in Figure 4.17(a). In fact, because of the variation in scale in the outer periphery, many of the unwanted noise points from Figure 4.17(a) are not reproduced in the log-polar implementation. As well, when both the fovea and periphery are considered together, the strongest interest points correspond to the towers along the wall.

Figure 4.21(c) shows how the algorithm performs on a face located in the periphery. Examining the peripheral image, the two points found on the left of this image correspond to the nose and chin of the face, and the single point on the right marks the top of the tie. However, the eyes, found in Figure 4.18(a) are not found in the peripheral version. This is due to the change in scale of the objects in the log-polar mapping, making them appear much smaller in the far periphery. The interest points that would correspond to the eyes occur quite close to the stronger nose point, corresponding to the centre of the entire head, and are subsequently smoothed into a single cluster. However, this is a desired effect, since only larger features, such as the head, should be identified in the periphery. These points can subsequently be centred on the fovea, permitting the identification and examination of interest points corresponding to the smaller features.

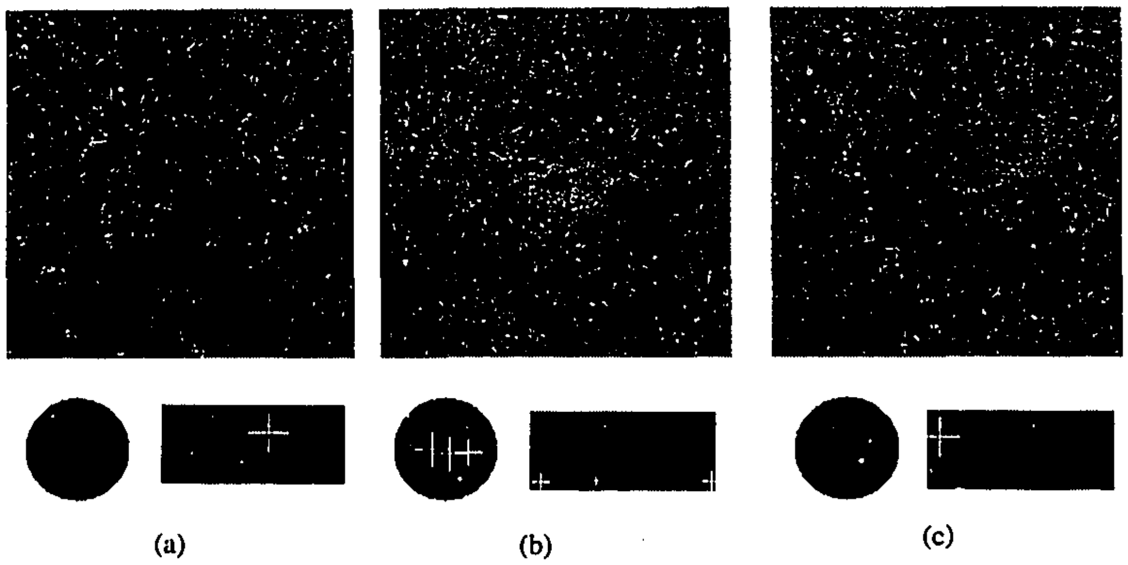
Finally, Figure 4.22 shows the algorithm's ability to track an object as it moves across the camera's field of view. The object is first found in the periphery as a single interest point (Figure 4.22(a)). As the object moves through the fovea, a number of other interest points are found, corresponding to various features within the object (Figure 4.22(b)). When the object passes back into the periphery, it is again tracked by a single interest point (Figure 4.22(c)).


(a)

(b)

(c)

FIGURE 4.21. Applying the algorithm to foveated images



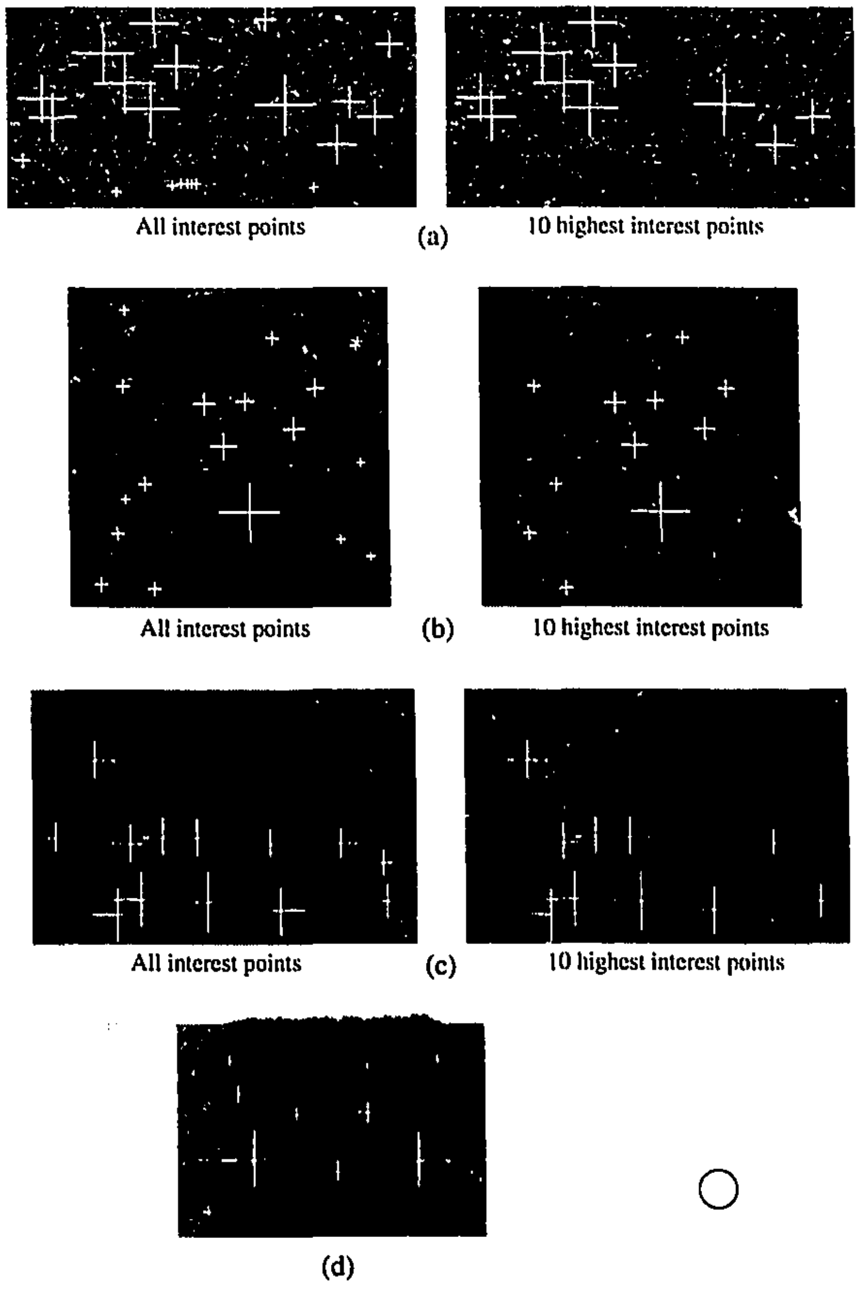
The same object is placed at different locations in the visual field. The algorithm manages to track it as it moves through the fovea. The original images, which were artificially created, are shown at half the size actually used to produce the fovea and periphery images.

FIGURE 4.22. Tracking an object through the visual field

3.1.4. *Comparison with other algorithms.* It is useful to compare the results produced by this algorithm with other algorithms described in the literature. However, as discussed in Chapter 3, there is only one other implementation known to the author which attempts to model overt visual attention. This is the algorithm of Reisfeld, Wolfson and Yeshurun [63], and so, this is the only algorithm which is suitable for direct comparison.

Figure 4.23 shows the output of the radial symmetries found by Reisfeld, Wolfson and Yeshurun's algorithm, applied at the same range of scales and with a similar smoothing and local maxima detection stage as the outputs produced in Figures 4.17 and 4.18. Where many interest points are found, a second image showing the top ten interest points is also included. It can be seen that the outputs of this algorithm are similar to those produced by the interest algorithm presented in this thesis. In some cases, such as the faces, their algorithm produces a few more noise points, while in others, such as the scene in Figure 4.23(c), fewer spurious points are produced by this algorithm. However, when the top interest points are compared the results are very similar.

The distinction between the two algorithms can be seen when comparing computational complexity. Whereas Reisfeld, Wolfson and Yeshurun's algorithm requires an operation be performed on *every* pair of edges in the image, the algorithm presented in this thesis only requires operations to be performed on pairs of *cocircular* edge elements. This leads to


All interest points
(a)
10 highest interest points
All interest points
(b)
10 highest interest points
All interest points
(c)
10 highest interest points
(d)

For comparison purposes, the interest points found by Reisfeld, Wolfson and Yeshurun's algorithm [63] are shown. Where two images are shown, the one on the left marks all the interest points found. The one on the right shows only the ten highest interest points.

FIGURE 4.23. Outputs obtained with the interest algorithm of [63]

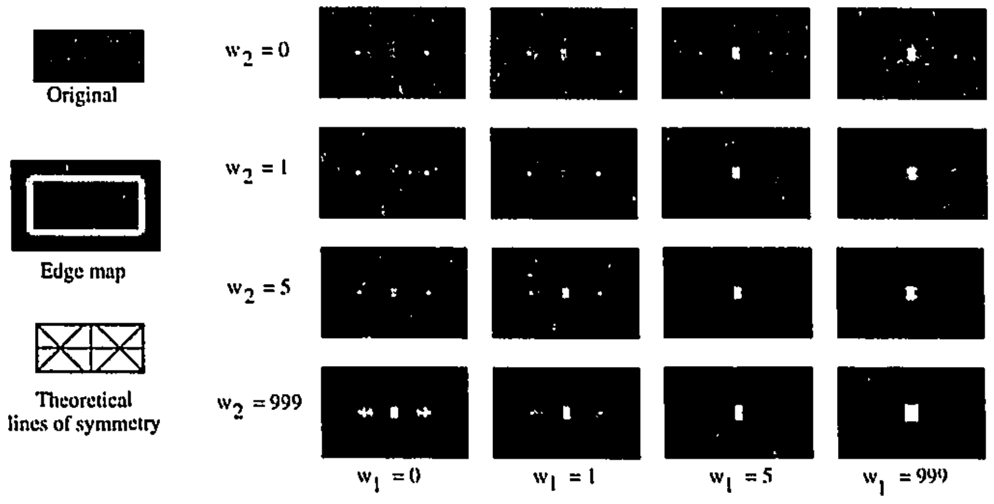
a very significant reduction in the computation required to produce the results. It also permits further simplifications, making possible a near real-time implementation which will be presented in Chapter 5.

3.2. Determining parameter values. There are four parameters that could not be determined through theoretical arguments and so, were evaluated experimentally. These are the two weighting parameters, w_1 and w_2 (see equations(3) and (6)), and two values that determine the size of each angular bin template: the number of bins the annular template is divided into and the amount of overlap between neighbouring angular bins (see sections 1.2.1).

3.2.1. *The effect of the weighting parameters.* To understand the effect of the weighting parameters, it is useful to begin by evaluating the output on a simple geometric shape. A rectangle was chosen for this purpose because it's symmetry configuration is complex enough to show all the effects of the weighting parameters, yet simple enough to understand how changes in these parameters affect the output. Figure 4.24 shows the raw interest map (before smoothing and local maxima determination) of the algorithm applied to the rectangle for a variety of combinations of the weighting parameters w_1 and w_2. By examining the crossing points of the theoretical lines of symmetry for the shape, it can be seen that there are three points of intersection of lines of symmetry: one in the centre and two near either end of the rectangle. The actual output of the algorithm differs from the theoretical lines of symmetry in two ways. First, because of the spread in the templates, the lines of symmetry found are more diffuse. Second, because of quantization effects in the edge detection, the corners of the rectangle are interpreted as diagonal edges, adding to the lines of symmetry output by the algorithm.

With both the weighting parameters w_1 and w_2 set to zero, all the lines of symmetry found by the algorithm can be seen. It is evident that this output is much noisier than the theoretical lines of symmetry, though the peaks of the raw interest map for this set of parameters do fall on the three desired intersection points. However, by setting the two weighting parameters to zero, the angle between contributing edges and the angle of intersection of the lines of symmetry are not taken into account. Because of this, the outer intersection points are more prominent than the central point since more lines of symmetry cross at these points. By increasing the value of w_1, parallel edges are weighted more in the symmetry contribution than perpendicular ones, and so the horizontal and vertical axes of symmetry are weighted more heavily than the diagonal axes, which also causes the central intersection point to be more prominent.

Increasing the value of w_2 makes the centre point more prominent, since it causes lines of symmetry crossing at 90 degrees to be weighted more heavily than crossings at other angles. However, the diagonal lines of symmetry also cross at 90 degrees at the outer



This image shows the effect of different weighting parameter values on the raw interest map of a rectangle. Brighter points in these images correspond to points of higher interest magnitude. These results were produced with the following parameters: number of angular bins = 32, overlap between neighbouring angular bins = 0%.

FIGURE 4.24. Effect of different weighting parameters on a simple shape

intersection points, giving them more weight as well. Therefore, in order to output the desired centre of gravity of the rectangle, w_1 should be set greater than zero, reducing the effects of the diagonal symmetry axes. By doing this, w_2 can be increased to enhance the relative weight of the central crossing point with respect to the outer ones (see the $w_1 = 5$ column of Figure 4.24).

The effects of these parameters on real-world objects is not so apparent from the raw interest data. Thus it is necessary to analyze the outputs after smoothing and local maxima detection. Figure 4.25 shows the interest points found for two images under a variety of weighting parameter settings. We will begin the analysis with the simpler image, the drill in Figure 4.25(a). In this example, it can be seen that when a small amount of smoothing is applied, the output becomes noisy for both weighting parameters set to zero, producing interest points that are not even within the object. As the smoothing is increased, these noisy points are absorbed into the dominant interest points of the drill, making the output less susceptible to noise. However, even with more smoothing the interest points for values of $w_1 = 0$ and $w_2 = 0$ are not well centred on the features they represent. As the weighting values are increased, regardless of the smoothing applied, the output becomes more sensitive to the central disc, and less to the elongated ends of the drill.

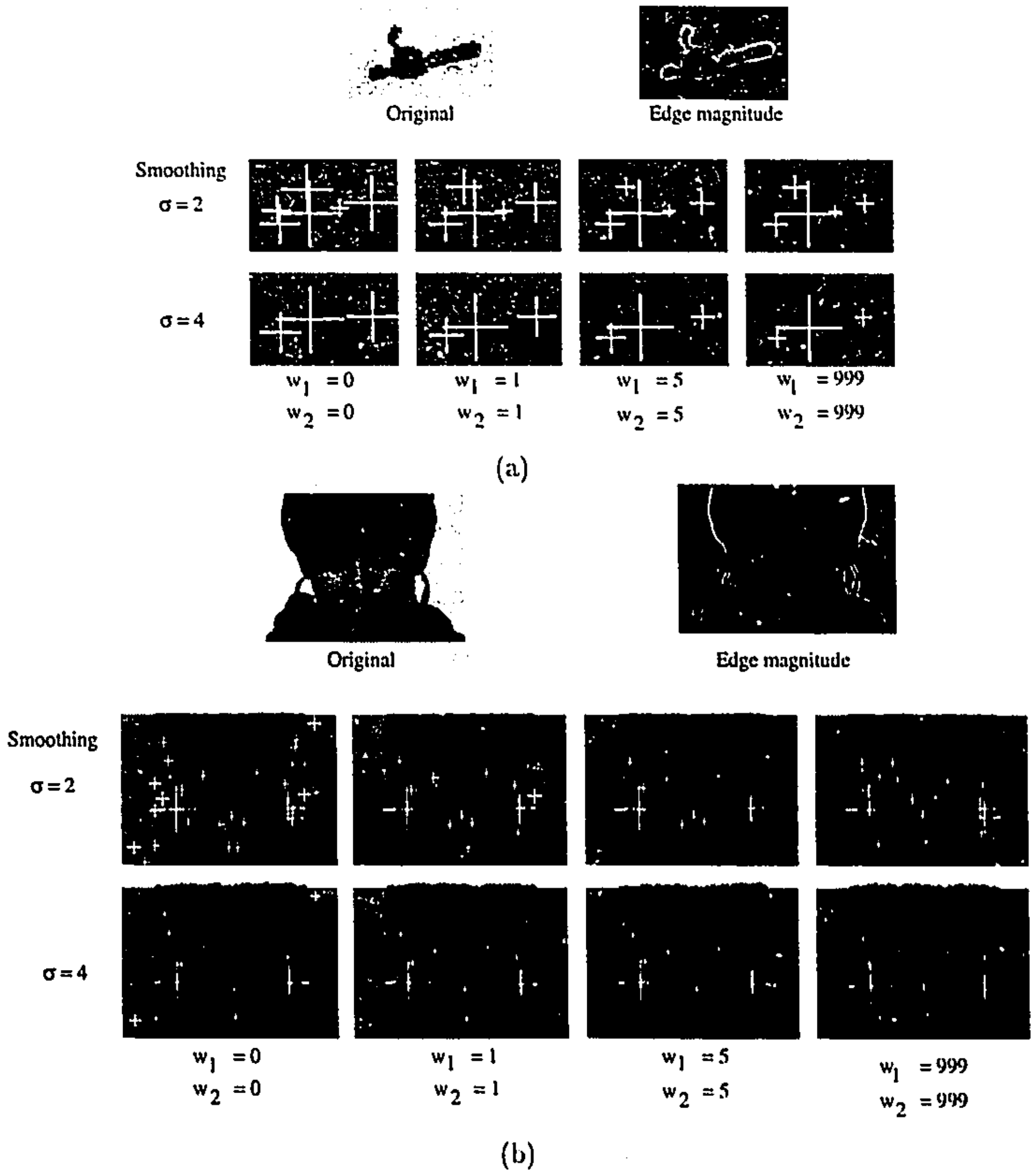
Figure 4.25(b) shows the same results for a more complex image. Again, it can be seen that small values for both σ and the weighting parameters lead to spurious interest points.

This effect can be reduced by increasing the weighting parameter values. However, this is true only up to a point. For very large values of the weighting parameters, the number of output points increases as well, giving many, closely spaced interest points. We note that this effect can be reduced by increasing the smoothing.

The above examples show some interesting consequences regarding the interplay of the weighting parameters and the amount of smoothing required. Setting the weighting parameters low produces many peaks in the interest map because all the intersections of lines of symmetry are equally weighted. Increasing the weighting parameter values causes the algorithm to be more selective in the type of intersections of lines of symmetry that are output as interest points. However, when these parameters are set too high, very few regions in a real image correspond to what the parameters are tuned to, and so most of the local maxima correspond to points of much lower interest magnitude. As well, because of the high selectivity of the algorithm with these parameters, the raw interest maps tend to consist of many very small clusters of points, rather than a smooth continuum. Such images require a much higher degree of smoothing to blend these clusters into a single point. It can therefore be reasoned that there is a trade-off between sensitivity to the weighting parameters and the amount of smoothing required. However, an intermediate value of weighting parameters is insensitive to variations in smoothing and produces acceptable results. Therefore, values of $w_1 = 5$ and $w_2 = 5$ are chosen as typical values for the weighting parameters.

3.2.2. *Effect of the angular bin parameters.* Two parameters are used to define the extent of each angular bin in the annular templates. These are the number of angular bins each annulus is divided into and the amount of overlap between neighbouring angular bins. These parameters limit the extent of each angular template. By increasing the template's extent, the output becomes more dispersed, acting as a kind of smoothing process.

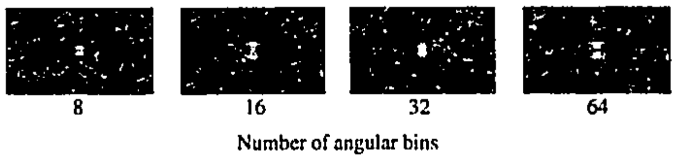
Figure 4.26 shows how this affects the raw interest map of the rectangle for 8, 16, 32, and 64 angular bins. These values were selected to take advantage of the symmetry of the annular templates about both the x and y axes. This symmetry permits the templates to be stored more efficiently in memory. The figure demonstrates that the output becomes more spread out as the number of bins is decreased. However, when actually applied to real-world images, this parameter has only a minor effect on the output, though some issues do warrant attention. For example, the results for the drill (Figure 4.27(a)) show that when the number of bins is set too small (8 bins), the output is quite sensitive to the amount of smoothing. This is because with so few angular bins, each angular template affects a large portion of the image. This creates a more homogeneous raw intensity image, which causes



The effect of different weighting parameter values on real-world objects under various levels of smoothing. These results were produced with the following parameters: number of angular bins = 32, overlap between neighbouring angular bins = 25%.

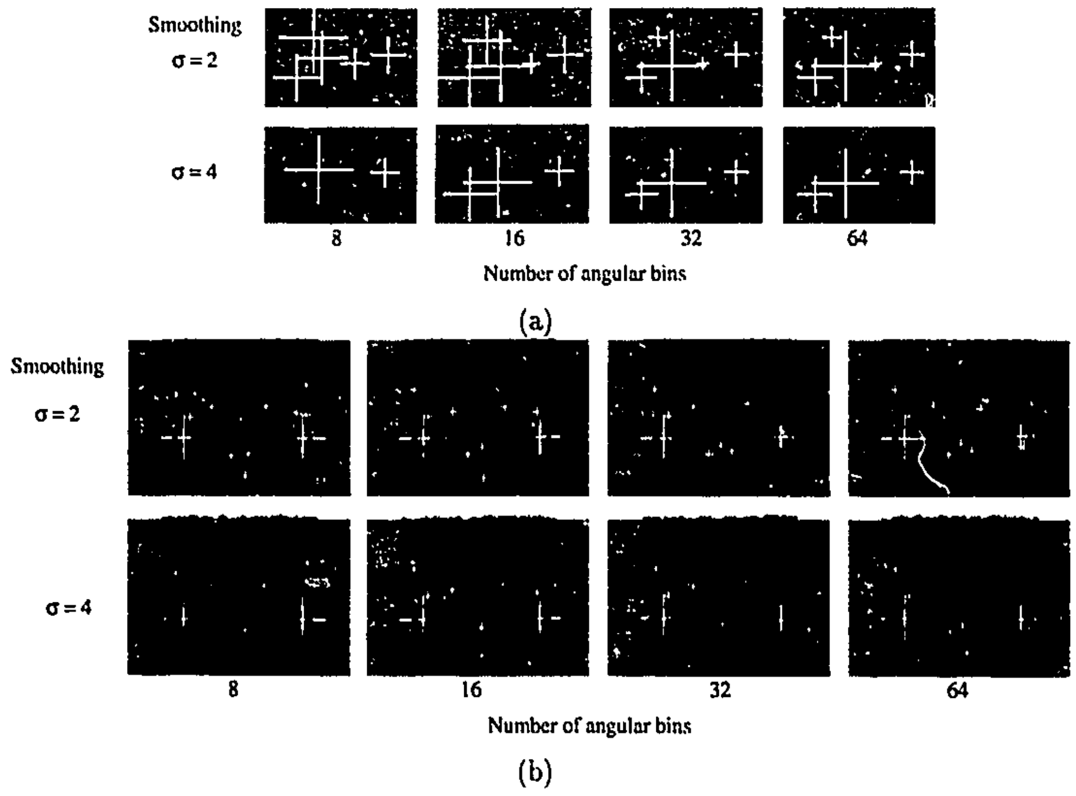
FIGURE 4.25. Effect of weighting parameters and smoothing on real-world objects

the points of local maximum to have interest magnitudes very close to the background interest magnitude. This results in spurious interest points when the amount of smoothing is low, and missing interest points when the amount of smoothing is increased.



The raw interest map for the rectangle of Figure 4.24 is shown for different numbers of angular bin divisions. Brighter points in these images correspond to points of higher interest magnitude. These results were produced with the following parameters: $w_1 = 5, w_2 = 5$, overlap between neighbouring angular bins = 0%.

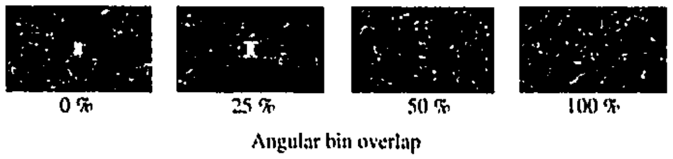
FIGURE 4.26. Effect of number of angular bins on a simple shape



The interest points found for varying numbers of angular bins. These images were produced with the following parameters: $w_1 = 5$, $w_2 = 5$, overlap between neighbouring angular bins = 25%.

FIGURE 4.27. Effect of number of angular bins on real-world images

The effects of the same set of parameters on the face image is shown in Figure 4.27(b). Again, the shortcomings of setting the number of bins too small can be seen to result in some spurious interest points for small smoothing values. As well, setting the number of bins too large (64 bins) causes too much segmentation in the raw interest maps, requiring more smoothing to bring nearby points together.



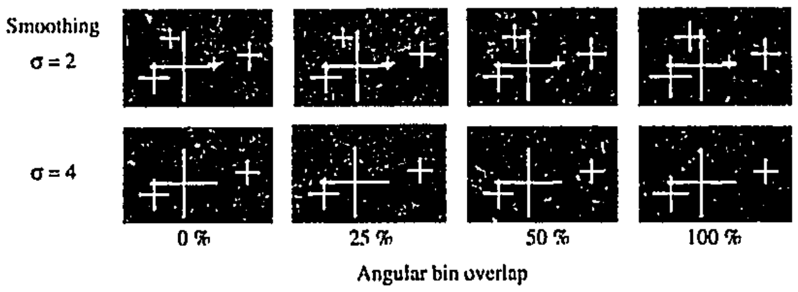
The raw interest map of the rectangle for different amounts of angular bin overlap. These images were produced with the following parameters: $w_1 = 5$, $w_2 = 5$, number of angular bins = 32.

FIGURE 4.28. Effect of angular bin overlap on a simple shape

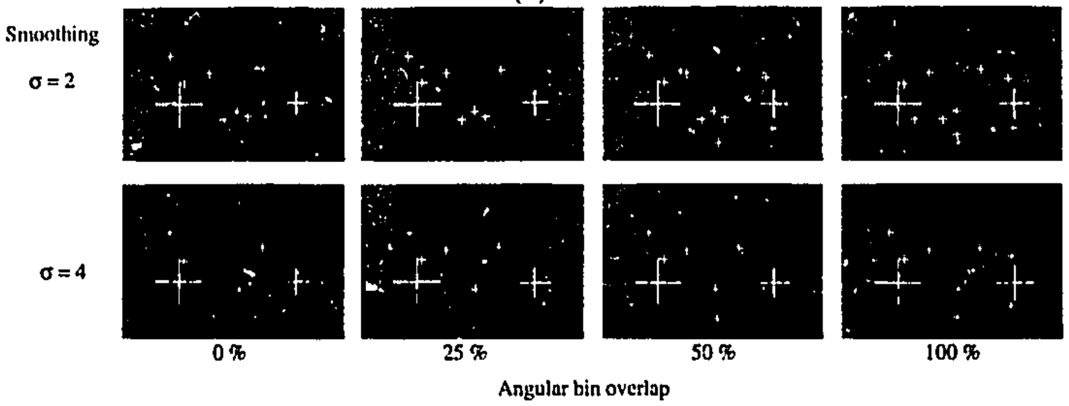
Similar, though even less dramatic effects can be observed with changes in bin overlap. Figure 4.28 demonstrates this on the raw interest map of the rectangle. As expected, larger values of overlap lead to templates of larger extent, and therefore produce a more dispersed interest map. However, this effect is too slight to produce changes in the drill image (Figure 4.29(a)). However, when applied to the face image (Figure 4.29(b)), some differences can be seen. More angular bin overlap produces more interest points. However, setting this value too small can result in the ioss of some desired interest points, as can be seen by the loss of one of the eyes when $\sigma = 4$ and bin overlap = 0%.

These examples show that, outside of extremely low or high values, the algorithm is not too sensitive to the exact selection of the annular template parameters. Thus the selection of the parameters can be motivated mainly by the desire for computational efficiency. For example, the fewer the number of angular bins, the more points there are that correspond to each bin. Therefore, a small number of angular bins requires many more points in the accumulator arrays to be updated for each edge. However, too many angular bins can make the output fragmented. A similar argument holds for the choice of bin overlap. The larger the overlap, the larger each angular bin's extent, and the more points in the accumulator arrays that must be updated for each edge. However, too small a value can make the algorithm too sensitive. Therefore, 32 angular bins, each overlapping their neighbouring bins by 25%, were selected.

3.3. Summary of results. The development and analysis of the algorithm presented in this chapter proved the validity and effectiveness of the developed interest point model. Of particular importance was the algorithm's insensitivity to exact parameter settings. This allows a set of parameters to be chosen that will be effective under a wide range of real-world conditions. Notably absent from this discussion was an evaluation of the computation requirements and execution speed of the algorithm. However, the model has been set up to allow for a fast, efficient implementation, and this is the subject of the following chapter.



(a)



(b)

The interest points found for different amounts of angular bin overlap. These images were produced with the following parameters: $w_1 = 5$, $w_2 = 5$, number of angular bins = 32.

FIGURE 4.29. Effect of angular bin overlap on real-world images

CHAPTER 5

A Real-Time Implementation of the Visual Attention Model

A real-time implementation of the algorithm is presented in this chapter. This implementation is accomplished in two ways. First, through simplifications and modifications to the algorithm discussed in the previous chapter. Second, by parallelizing the algorithm and distributing the processing amongst a network of processors. Each of the first two sections of this chapter concentrates on one of these methods. The chapter concludes with a third section which examines the performance of the real-time algorithm and shows results of its application to real images. These results are compared to the ones presented in chapter 4, thereby showing that the simplifications do not significantly affect the results.

1. Changes to the Algorithm

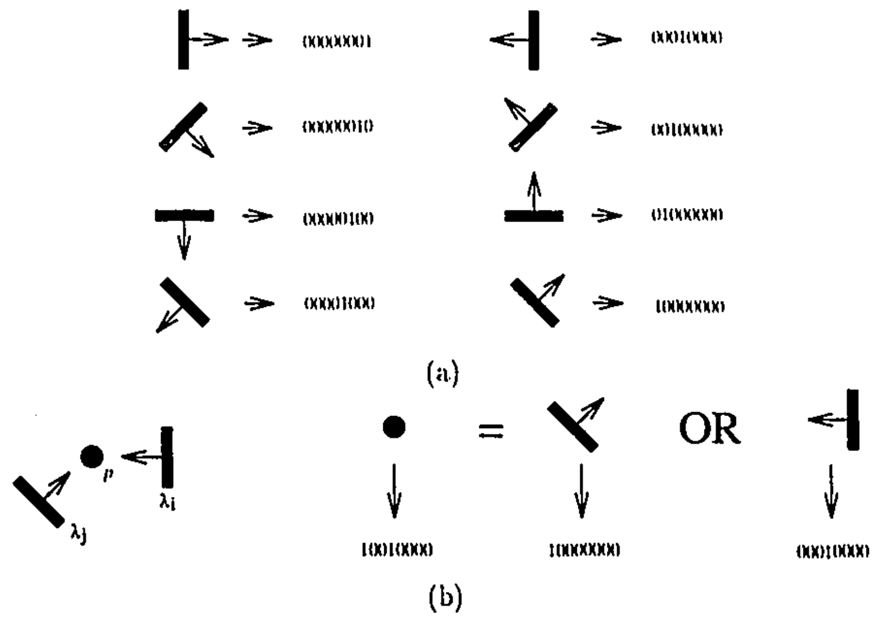
Some improvements in the execution speed of the algorithm were achieved in the previous chapter by inverting the annular templates and separating the Gaussian convolution. These changes did not affect the actual output of the algorithm. In contrast, this chapter will introduce certain simplifications to the operator which will change the output slightly. These simplifications are achieved in one of two ways. First by simplifying the calculations that must be applied to each of the data points. Second by reducing the number of data points that these calculations must be applied to. However, it will be shown that the consequences of these changes are minor, and are offset by tremendous increases in the speed of execution.

To begin, both simplification goals, reduction of computation and reduction of data, are achieved through the use of a thresholded and binarized edge map. The number of edges in the image is greatly reduced by thresholding the gradient image so that very small magnitude edges are not considered. This results in a significant reduction in the amount of data that must be processed in the image. As well, by using a binary edge map, the algorithm's reliance on edge intensity is removed since each point is either an edge ($||\lambda|| = 1$) or is not ($||\lambda|| = 0$). This simplifies the computational complexity at each point.

Even further computational savings can be achieved by taking advantage of the binary nature of the edge data. Referring to the processing flow diagram of Figure 4.8, it is seen that the first stage of the algorithm involves the application of the inverted annular templates on the edge points, resulting in contributing edge data being stored in each of the accumulator arrays. In the implementation of the previous chapter, the values stored in the accumulator arrays were the magnitudes of the strongest contributing edges at each orientation. However, since the algorithm no longer relies on edge intensity, it is not necessary to store these values in the accumulator arrays. In fact, the only information that needs to be recorded in the arrays is whether or not there is any edge above threshold which contributes to a point at a particular orientation. Therefore, at each point in the accumulator arrays, a single binary digit for each contributing orientation is sufficient to store all the information required. A binary 1 indicates that there is at least one edge contributing to the point's symmetry from the orientation represented by that bit, and a 0 signifies no edge is present.

The fact that edge magnitude information does not need to be stored in the accumulator arrays means that the application of the inverted annular templates can be simplified as well. Since only binary information needs to be stored, the application of the templates can be accomplished using a simple Boolean operation. To do this, a binary word is defined, with each binary digit associated with a particular edge orientation. This binary word is used to represent the edge information in the accumulator arrays. Similarly, a unique binary mask is assigned to each template orientation. This mask has a single bit set, corresponding to the bit position representing the template's orientation in the accumulator array (Figure 5.1(a)). Each accumulator array point affected by an inverted template is updated by logically ORing the value at that point with the template's binary mask (Figure 5.1(b)). In this way, the bits corresponding to the orientation of the contributing edges are set, and the remaining bits are unaffected.

Referring again to Figure 4.8, the next stage in the processing is the application of equation(3). This equation determines the symmetry magnitudes for every symmetry orientation based on the contributing edges stored at each point of each accumulator array. However, the fact that contributing edge data is now stored in the accumulator arrays as a single binary word at each point can be exploited for further computational simplifications. Specifically, this equation can be replaced by a series of look-up tables, each one mapping

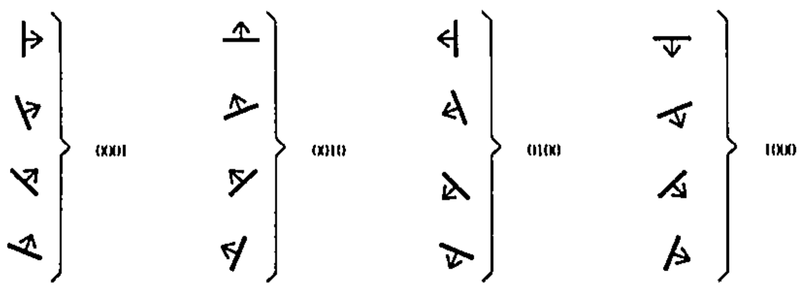


(a) An example of template masks and corresponding edge orientations for an annular template with 8 angular bins. (b) The value at point p of an accumulator array is found by logically ORing the template masks of all the edges contributing to that point. In this way, each bit set in the accumulator array represents an orientation in which at least one inward pointing, contributing edge was found.

FIGURE 5.1. Binary template masks

the binary value stored in an accumulator array to the corresponding symmetry magnitude of one symmetry orientation.

An obvious problem is that the binary word stored in the accumulator arrays has as many bits as the number of angular bins, which can lead to very large look-up tables for even modest numbers of angular quantification. However, it was shown in the previous chapter that the output of the algorithm is not very sensitive to the number of angular bins. Thus this problem can be solved by using fewer angular bins. But the fewer angular bins selected, the larger the extent of each inverted template, which consequently increases the computation required. This means that, in order to maintain practical sizes of look-up tables, the number of angular bins must be small. However, to maintain fast computation times, the number of angular bins should be large. These seemingly conflicting requirements can both be satisfied by using the full range of edge orientations to determine the extent of each angular template, and using a coarser representation of orientation to store the edge contributions in the accumulator arrays. In practice, this is accomplished by using the same binary mask for a number of template orientations (Figure 5.2). This means that any edge



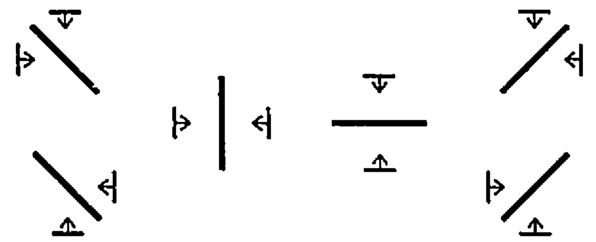
A number of angular bins can map to the same binary template mask. In this example, 16 angular bins are reduced to 4 unique orientations for storage in the accumulator arrays.

FIGURE 5.2. Reducing the accumulator array storage word length

which contributes to a particular point from any of these template orientations sets the same bit in the accumulator arrays.

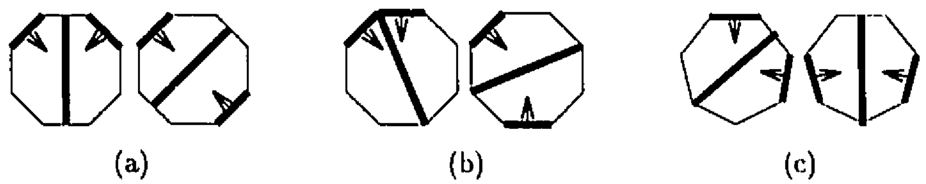
A further consequence of limiting the number of contributing edge orientations represented in the accumulator arrays is that it similarly limits the number of possible symmetry orientations. For example, Figure 5.3 shows all possible pairs of contributing edges, and therefore all possible symmetry orientations, when edge orientations are quantized to four contributing directions for accumulator array storage. Thus we note that only four symmetry orientations are possible given four contributing edge orientations. In fact, this result can be generalized for any number of edge orientations. Hence it can be seen that quantizing the contributing directions down to n values fixes the number of possible symmetry orientations to n as well.

Furthermore, since the edge magnitudes are binarized, the symmetry magnitude, $S_{r,\psi}(p)$, relies only on the angle between the contributing edges (see equation(3)). Similarly, since the number of contributing edge orientations is limited, the value of $S_{r,\psi}(p)$ can take on only a limited number of distinct values. Based on this, three types of lines of symmetry can be defined, differentiated by the number of distinct symmetry magnitudes each can have. Two of these occur when the number of contributing edge directions stored in the accumulator arrays is even. The first of these, which we call *edge-symmetry*, results in lines of symmetry that pass through the centre of a possible contributing edge orientation (Figure 5.4(a)). In the other type, called *vertex-symmetry*, the lines of symmetry pass through the vertex of two possible contributing edge orientations (Figure 5.4(b)). The third type of symmetry occurs when the number of contributing directions is odd (Figure 5.4(c)). In this case, all the lines of symmetry pass through the centre of a possible contributing edge at one end and a vertex at the other and is therefore called *edge-vertex-symmetry*. Each



Limiting the number of possible contributing edge orientations results in a similar limitation on the number of possible symmetry orientations. In this example, it can be seen that only four symmetry orientations are possible given four contributing edge orientations.

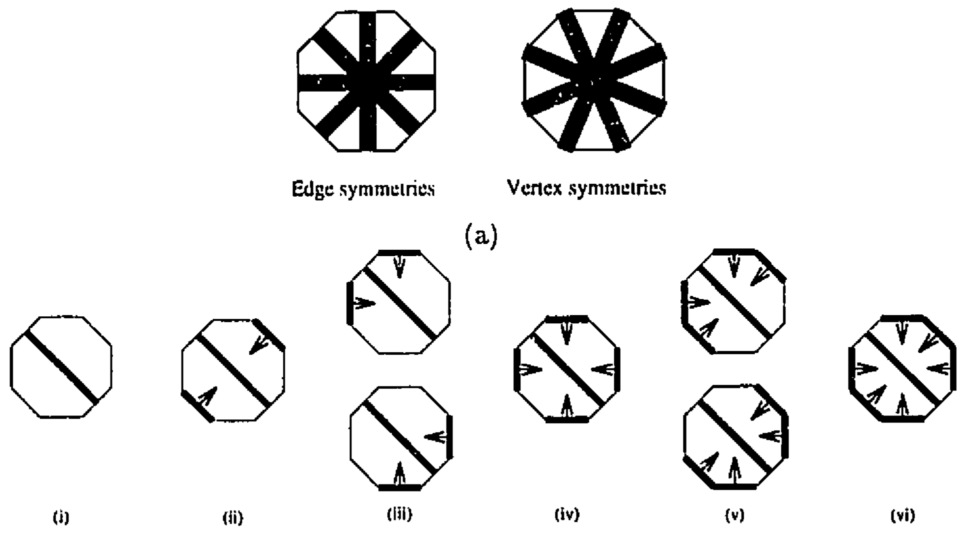
FIGURE 5.3. The relationship between the number of edge orientations and the resultant symmetry orientations



Some examples of (a) edge-symmetries, (b) vertex-symmetries, and (c) edge-vertex-symmetries.

FIGURE 5.4. Three types of symmetry orientations

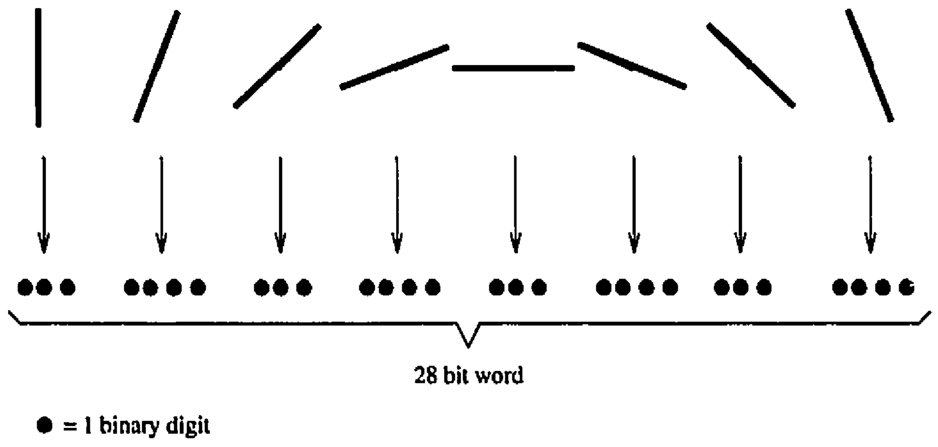
of these symmetry types produces a unique set of possible symmetry magnitudes since the combination of angles separating the edges resulting in each of these symmetries is different. The actual values of these magnitudes depends on the value of the weighting parameter, w_1 (see equation(3)), but the number of distinct values is based solely on the possible angles between edges contributing to each symmetry type. This means that these discrete values can be represented by fewer bits than required for symmetry magnitudes with continuous values. For example, eight contributing orientations result in four edge-symmetries and four vertex-symmetries (Figure 5.5(a)). Each edge-symmetry can take on one of six possible magnitudes (Figure 5.5(b)). This range can be represented uniquely in three bits, with each symmetry magnitude mapping to a particular binary value. A similar analysis shows that there are nine possible vertex-symmetry values, which requires a four bit representation. By concatenating the binary representation of these values, we can encode the symmetry magnitudes for all orientations at each point in one 28 bit binary word (Figure 5.6). This leads to a further simplification, since only a single look-up into a table of 28 bit words is required to obtain the symmetry magnitudes for all orientations from the contributing edge information stored at each point in an accumulator array.



(a) Eight edge orientations results in four possible edge-symmetries and 4 possible vertex-symmetries. (b) Each edge-symmetry can take on one of six possible magnitudes based on the edges contributing to that line of symmetry (see equation(3)). These values are:

(i) 0
(ii) $(\sin\pi/2)^{w_1}$
(iii) $(\sin\pi/4)^{w_1}$
(iv) $2(\sin\pi/4)^{w_1}$
(v) $(\sin\pi/2)^{w_1} + (\sin\pi/4)^{w_1}$
(vi) $(\sin\pi/2)^{w_1} + 2(\sin\pi/4)^{w_1}$

FIGURE 5.5. Possible symmetries for eight contributing orientations



All the possible symmetry magnitudes for eight possible symmetry orientations can be completely encoded in one 28 bit binary word using three bits to represent the six possible magnitudes of each of the four edge-symmetries and four bits for the nine possible magnitudes of each of the four vertex-symmetries.

FIGURE 5.6. Representing the symmetry set in a single binary word

The next step in the algorithm, as shown in Figure 4.8, is the combination of the individual symmetries computed for each accumulator array. This is accomplished through the application of equation(5), which takes the maximum of the symmetries of each orientation over all the accumulator arrays. This procedure requires a separate comparison over all radii for each symmetry orientation at each point. However, since the symmetry magnitudes at each point are encoded in a single binary word, the values corresponding to each orientation must be decoded from these words before the comparison can be made. Alternately, this is made much more efficient by slightly altering the way the symmetry magnitudes are stored so that a Boolean function that is equivalent to taking a maximum can be applied over all the orientations in parallel. This is accomplished by increasing the number of bits used to store the magnitude of each symmetry orientation. We refer to this as the *long-symmetry* bit format, to differentiate it from the regular binary format, which we call the *short-symmetry* bit format. The way that the long-symmetry format is computed and used is best illustrated through a specific example. Figure 5.7(a) shows the differences in the two formats in storing four symmetry magnitude values. Increasing the number of bits in the long-symmetry storage causes the mapping of symmetry magnitudes to these long-symmetry values to be nonunique. By taking advantage of the redundancies, we can encode these values so that the result of logically ORing two long-symmetry values together maps to the same symmetry magnitude as the maximum of the two. To illustrate how this is accomplished, Figure 5.7(b) shows an example of this procedure for four symmetry magnitude values at each of eight symmetry orientations.

Using the long-symmetry bit format for finding maxima requires a significant increase in the number of bits needed to store the symmetry magnitudes. For each contributing direction, $n-1$ bits are required to encode n symmetry magnitudes. For example, using eight contributing directions produces four vertex-symmetries with nine possible magnitudes each, and four edge-symmetries with six possible magnitudes each. To encode these values using the method described above requires a binary word 52 bits long $(4\cdot(9-1)+4\cdot(6-1))$. Such word sizes are too large for a practical system, and so reducing this is necessary. This is accomplished by further quantizing the number of symmetry magnitudes. For example, limiting the symmetry values to four magnitude levels for each of eight contribution directions results in a total word size of 24 bits $(8\cdot(4-1))$.

We are now left with a single binary word at each output point, encoding a magnitude of symmetry for each orientation at that point. The final processing step, as shown in Figure 4.8, involves the application of equation(6) to these values, producing a unique

	Short-symmetry		Long-symmetry
1)	00	$\Leftrightarrow$	000
2)	01	$\Leftrightarrow$	001
3)	10	$\Leftrightarrow$	01x
4)	11	$\Leftrightarrow$	1xx

(a)

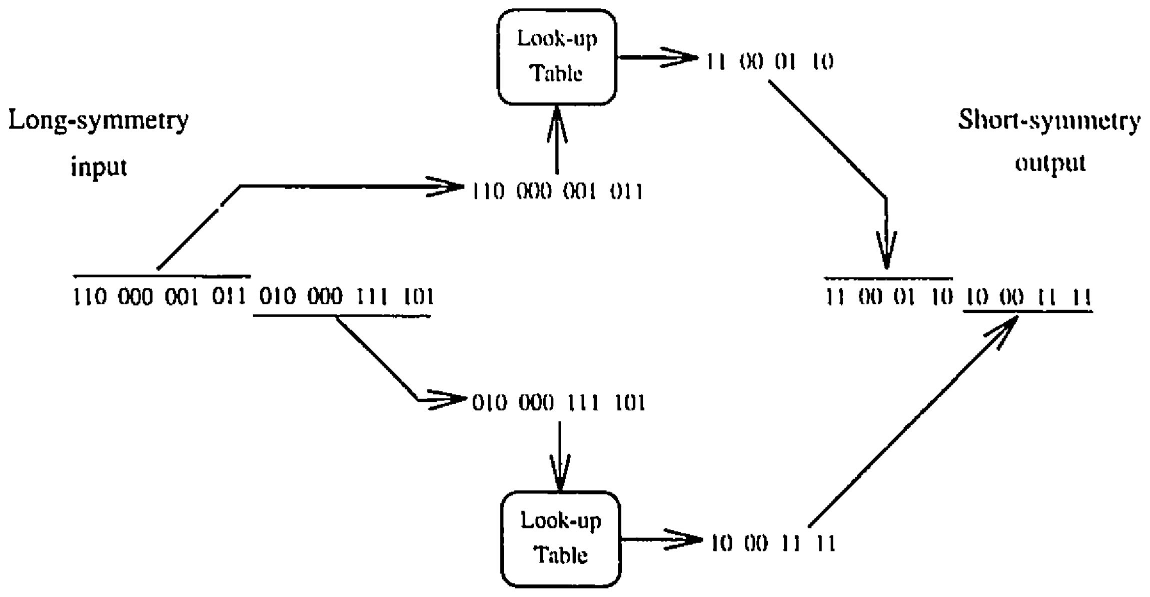
Short-symmetry set		Long-Symmetry set
00 01 11 00 00 10 10 01	$\Leftrightarrow$	000 001 100 000 000 010 010 001
MAX		OR
10 00 10 00 01 10 01 11	$\Leftrightarrow$	010 000 010 000 001 010 001 100
10 01 11 00 01 10 10 11	$\Leftrightarrow$	010 001 110 000 001 010 011 101

(b)

(a) An example of representing four symmetry magnitudes in the short-symmetry and long-symmetry bit formats. The x's in the long-symmetry values represent bits that can be either 0 or 1. (b) Finding the maximum at each orientation of two short-symmetry values requires the bits representing each orientation to be compared separately. The same procedure can be accomplished for all orientations in parallel using the long-symmetry bit format and a simple logical OR operation.

FIGURE 5.7. The long-symmetry storage format

interest value, $I(p)$, at each point. Again, this equation can be replaced with a look-up table. This table uses the binary word representing the symmetry magnitudes of each point as an index. However, the symmetry magnitudes are stored in the long-symmetry bit format, and so, the size of the look-up table can be reduced by eliminating the redundancies in this encoding of the symmetry set. In the previous example of four magnitude levels at eight contribution directions, 24 bits were required to store all the symmetries in the long-symmetry format. A 24 bit index leads to a look-up table of 16 MWords. On the other hand, each symmetry value only represents four possible magnitudes, which can be uniquely encoded in two bits. Therefore, using the short-symmetry bit format, these symmetries can be represented by a 16 bit word (two bits for each of the eight orientations), which indexes into a look-up table of only 64 kWords. This is a significant memory savings. However, it does require an efficient method for converting from the long-symmetry format to the short-symmetry format. This can be accomplished through another look-up table. Using the full long-symmetry value as an index into this look-up table would still require a 16 MWord table, negating the reason for making this conversion in the first place. However, such a procedure can be achieved. First we split the long-symmetry value into two shorter



A 24 bit long-symmetry value is split into two 12 bit words, each indexing a look-up table which outputs the equivalent eight bit short-symmetry value. The two resulting eight bit values are then concatenated to form the 16 bit short-symmetry equivalent of the original 24 bit long-symmetry word.

FIGURE 5.8. Converting from long-symmetry to short-symmetry

binary words. Then we use a look-up table to determine the short-symmetry value of each of these two words separately. This procedure only requires a look-up table large enough to be indexed by half of the long-symmetry bits.

An example of this procedure is shown in Figure 5.8. In this example, a 24 bit long-symmetry value is broken down into two 12 bit words. Each of these 12 bit words represents the symmetry magnitudes of four of the eight orientations. These words are then used as an index into a look-up table (12 bits indexes into a four kWord look-up table) which outputs the equivalent short-symmetry value in eight bits. Concatenating the outputs from the two 12 bit words gives the 16 bit short-symmetry word corresponding to the original 24 bit long-symmetry value. It is this resulting 16 bit short-symmetry value that is used as the index into a final look-up table, replacing equation(6) in Figure 4.8.

1.1. Processing flow. The previous section describes the simplifications and changes made to the algorithm to improve its run-time efficiency. In doing so, many new concepts were introduced and an overall picture of the processing flow may have been lost. As such, this section reiterates the basic computational steps implemented in the real-time algorithm. These steps are illustrated in Figure 5.9.

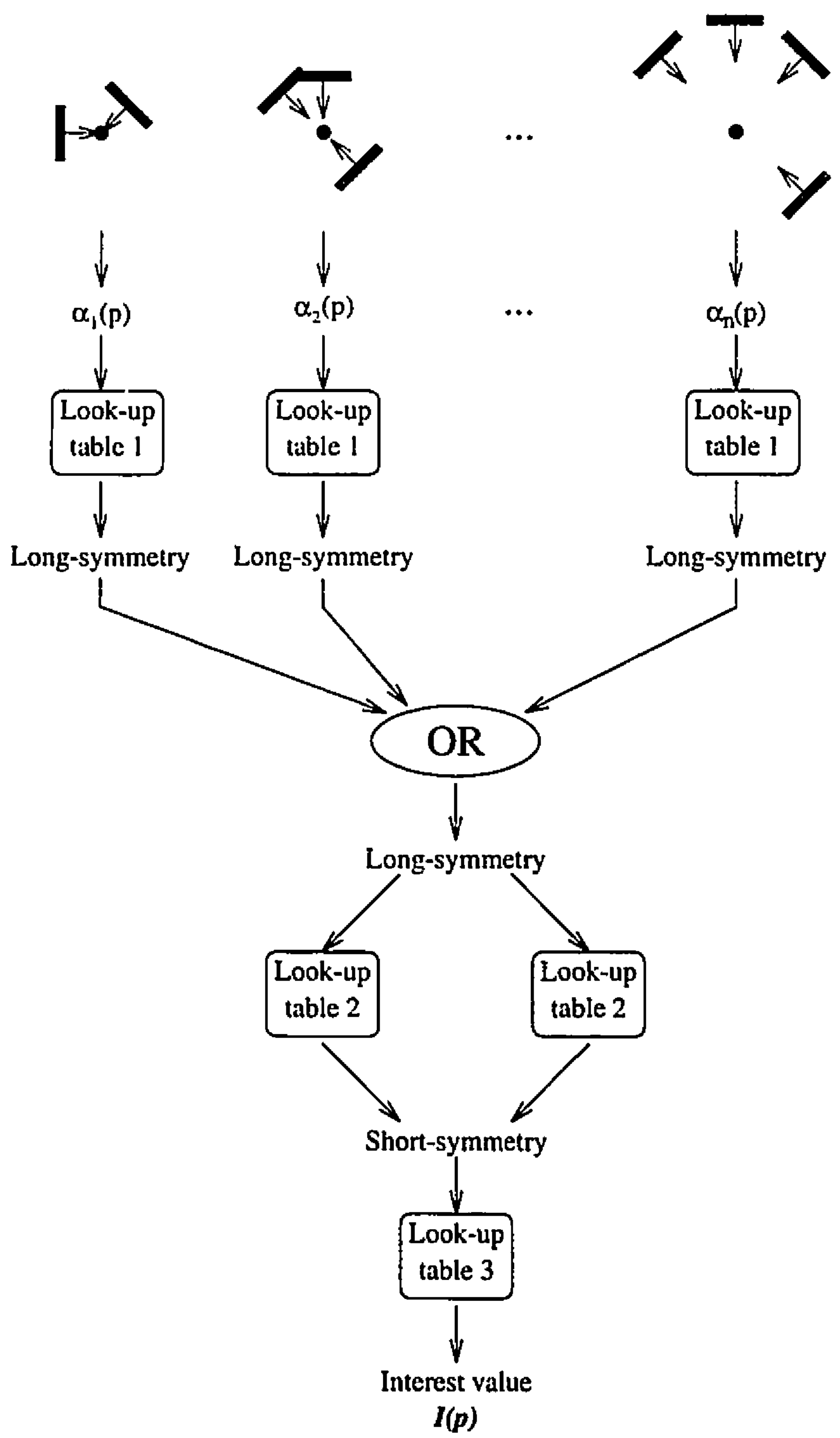


FIGURE 5.9. Real-time processing flow

All the edges, whose centres of cocircularity are at a point p, contribute to the values stored at point p of one of the accumulator arrays through the application of the inverted templates. The particular accumulator array affected is determined by the distance from the edge to the centre point p. These contributions are stored as a binary word in the accumulator arrays, indicated by $\alpha_1(p), \alpha_2(p), ..., \alpha_n(p)$ in Figure 5.9. These words are then used as indices into a look-up table, outputting the long-symmetry value resulting from the contributing edges of each accumulator array. All these long-symmetry values are then ORed together, resulting in a long-symmetry value that represents the maximal symmetry magnitude of each orientation over all the radii. This value is then converted to its short-symmetry equivalent by splitting it into two smaller binary words, each of which is used as an index into another look-up table. The outputs of these two table look-ups are joined, resulting in the short-symmetry representation of the symmetry magnitudes at each point. This value, which is equivalent to the original long-symmetry value with the redundancies in coding removed, is then used as an index into a final look-up table which determines the interest value, $I(p)$, for the point based on the particular symmetries at that point.

Finally, it is interesting to compare Figure 5.9 with Figure 4.8. Of particular note is how all the equations of the previous implementation have been replaced by either look-up tables or Boolean operations in the real-time implementation.

1.2. Determining parameters and tables. The real-time algorithm uses a number of look-up tables and quantizations in order to optimize its run-time efficiency. This section will examine how the look-up tables are precomputed, and how the number of quantization levels are chosen. Some of these values are determined by the amount of memory available for the look-up tables and the size of each machine addressable word. As such, the specifics of the hardware platform that is being used for this implementation must also be considered.

1.2.1. *Selection of quantization levels.* There are a number of simplifications applied to the algorithm which require certain quantization levels to be determined. Specifically, the number of contributing directions stored in the accumulator arrays and the number of levels of symmetry magnitudes must be chosen. These values are interdependent since the number of contributing directions directly affects the number of possible symmetry magnitudes. As well, choosing these values depends on a number of machine specific constraints, such as the amount of memory available and the size of an addressable machine word.

To maximize performance, we have implemented our system on a parallel network of Texas Instruments TMS320C40 (C40) processing nodes. These processors run at a clock rate of 40MHz[1], and are capable of executing floating point operations in a single clock cycle. As well, these processors were chosen due to their extensive communication capabilities, which will be discussed in the next section. For the purposes of this section, it is only necessary to mention two constraints pertaining to this hardware. The first is based on the amount of system memory available for the look-up tables. The largest look-up table computes the final interest value from a short-symmetry word index. The short-symmetry word is represented by an n bit binary number, which can index a look-up table of 2^n Words. The entire data memory space available is 256 kWords[2], which leads to a maximum index word size of $n = 18$. However, this leaves no room for other data, such as the accumulator arrays, other look-up tables and the annular templates which must be stored in this area as well. In order to accomplish this, the maximum value of n is limited to 17, giving a maximum table size of $2^{17} = 128$ kWords, and limiting the short-symmetry word size to 17 bits. Furthermore, it is desired that all the vertex-symmetry magnitudes be represented by the same number of bits, and similarly all the edge-symmetry magnitudes should be represented by the same number of bits. This ensures that the number of possible symmetry magnitudes for a particular symmetry type does not arbitrarily depend on the orientation of that symmetry. Table 1 shows all the possible ways these 17 bits can be divided to represent the edge magnitudes for different numbers of possible symmetry orientations. From this table it can be seen that there is a tradeoff between the number of symmetry orientations and the number of bits used to encode the magnitude of each line of symmetry. However, the number of orientations should not be made too small, since we are ultimately looking for edge contributions from a number of different orientations. Examining the table, one can see that there is a natural cutoff at eight orientations. With nine or more orientations only one bit per direction can be used to store magnitude. This representation is too coarse since some comparison of relative symmetry strengths is desirable. However, at eight orientations two bits can be used, allowing four symmetry magnitudes per orientation to be encoded. This produces a quantization of the six possible edge-symmetry magnitudes and the nine possible vertex-symmetry magnitudes to four levels each. This still provides enough differentiation to allow symmetries to be distinguished based on magnitude. Furthermore, eight symmetry

[1]A 50MHz version of these processors has recently become available.

[2]The actual memory available is 256kWords of SRAM, used to store data and tables, and 1 MWord of DRAM, used to store code. It is desired to store all the look-up tables in the SRAM because access to this memory is much faster than to the DRAM. This is especially true of accesses to non-sequential memory locations since DRAM imposes a substantial penalty each time a 64 kWord page boundary is crossed.

No. of orientations	Types of symmetries	Possible divisions of short-symmetry bits		
2	1 edge-symmetry and 1 vertex-symmetry	1 bit / edge-symmetry	and	16 bits / vertex-symmetry
		2	and	15
		3	and	14
		⋮		⋮
		16	and	1
3	3 edge-vertex-symmetries	5 bits / symmetry		
4	2 edge-symmetries and 2 vertex-symmetries	1 bit / edge-symmetry	and	7 bits / vertex-symmetry
		2	and	6
		3	and	5
		⋮		⋮
		7	and	1
5	5 edge-vertex-symmetries	3 bits / symmetry		
6	3 edge-symmetries and 3 vertex-symmetries	1 bit / edge-symmetry	and	4 bits / vertex-symmetry
		2	and	3
		3	and	2
		4	and	1
7	7 edge-vertex-symmetries	2 bits / symmetry		
8	4 edge-symmetries and 4 vertex-symmetries	1 bit / edge-symmetry	and	3 bits / vertex-symmetry
		2	and	2
		3	and	1
9	9 edge-vertex-symmetries	1 bit / symmetry		
10	5 edge-symmetries and 5 vertex-symmetries	1 bit / edge-symmetry	and	2 bits / vertex-symmetry
		2	and	1
11	11 edge-vertex-symmetries	1 bit / symmetry		
12	6 edge-symmetries and 6 vertex-symmetries	1 bit / edge-symmetry	and	1 bit / vertex-symmetry
13	13 edge-vertex-symmetries	1 bit / symmetry		
⋮				
17	17 edge-vertex-symmetries	1 bit / symmetry		

TABLE 1. Storage requirements for different orientation quantizations

orientations still allow a rich representation of the symmetries in the image. Finally, it should be remembered that the number of resultant symmetry orientations is equal to the number of possible contributing edge directions, and so choosing this value also fixes the number of possible contributing edge directions to eight. Therefore, at each point in the accumulator arrays, an eight bit binary word, corresponding to edge contributions from each of eight contributing directions must be stored. This is a further advantage, since eight bit words are a natural word size in most computer architectures, as it is in ours, and so specialized machine instructions exist to extract eight bit words efficiently and quickly.

The second constraint is the addressable machine word size for this architecture, which is 32 bits. To maximize efficiency it is desired that an entire long-symmetry value, encoding the symmetry magnitudes at all orientations, fit into one 32 bit word. Thus, this will require only one table look-up to go from the contributing edge data of the accumulator arrays to this long-symmetry value. Using the values chosen above, each of the eight symmetry orientations requires three bits in the long-symmetry word to store the four possible magnitude levels. This leads to a 24 bit long-symmetry representation of the symmetry set, which does fit within the 32 bit limit.

1.2.2. *Setting up the look-up tables.* There are three separate look-up tables used in the implementation. This section will discuss how the values for these look-up tables are computed. Though the computations involved in determining these look-up tables can be extensive, the advantage of their use is that all of these calculations can be performed on initialization. This removes the necessity of performing them at run-time, and therefore greatly improves algorithm performance.

The first look-up table outputs the the long-symmetry format of the extended symmetry set, given the binary word representing the contributing edge orientations at a point as an index. As was discussed previously, each index comes from a single accumulator array representing a particular range of radii. As well, each bit in the index represents whether or not an inward pointing edge was found at a particular orientation for that range of radii. Therefore, each bit in the index values represents a cocircular edge element at a given orientation. With this information, it is a simple matter to determine the symmetry values for all orientations. Given an index represented by the binary digits $\lambda_0, \lambda_1, ..., \lambda_n$, with corresponding contribution orientations $\phi_0, \phi_1, ..., \phi_n$, the set $\Gamma_{r,\psi}(p)$ can be defined as all bit pairs which contribute to symmetry orientation ψ at radius r of point p. From this, the magnitude for each symmetry orientation, ψ, is computed from the subset of bits belonging to $\Gamma_{r,\psi}(p)$. This is accomplished using equations(2) – (4), repeated below, and the variable definitions presented above which reflect the binary nature of the data:

(14) $$\psi(\lambda_i, \lambda_j) = \frac{\phi_i + \phi_j}{2}$$

(15) $$S_{r,\psi}(p) = \sum_{\lambda_i, \lambda_j \epsilon \Gamma_{r,\psi}(p)} \|\lambda_i\| \, \|\lambda_j\| \, (\sin \varphi/2)^{w_1}$$

(16) $$\varphi = |\phi_i - \phi_j|$$

This produces $S_{r,\psi}(p)$, a symmetry magnitude for each orientation. Each of these symmetry magnitudes is then assigned to a particular symmetry quantization level, and given the corresponding long-symmetry bit values for that level. These bit values for all the orientations are then placed end to end, resulting in a single binary word representing the symmetries for a particular arrangement of contributing edges. In this way, the long-symmetry value for each possible index is computed, and placed in a table so that the index maps to its corresponding long-symmetry value.

The second look-up table is used to convert long-symmetry values to their short-symmetry equivalents. This is accomplished by splitting the long-symmetry value in two, and using two table look-ups to find the upper and lower halves of the equivalent short-symmetry word. For each half of the split long-symmetry word, the short-symmetry half words corresponding to all the possible long-symmetry bit combinations can be determined. These values are placed in the look-up table so that the long-symmetry half word maps to its corresponding short-symmetry half word.

The final look-up table maps the short-symmetry word to an output interest value. Each possible short-symmetry word can be broken down into a symmetry magnitude quantization level for each orientation. Each symmetry orientation is then assigned the corresponding magnitude value for its quantization level. The interest value, $I(p)$, for each combination of symmetry magnitudes is computed by direct application of equation(6), repeated below, over all the contribution directions:

$$I(p) = \sum_{\psi_i,\psi_j} S_{\psi_i}(p) S_{\psi_j}(p) (\sin(\psi_i - \psi_j))^{w_2} \tag{17}$$

where $S_\psi(p)$ is the symmetry magnitude corresponding to a particular quantization level. This is computed for each possible short-symmetry value, and placed in a table so that the short-symmetry values map to their corresponding interest values.

2. Parallelizing the Algorithm

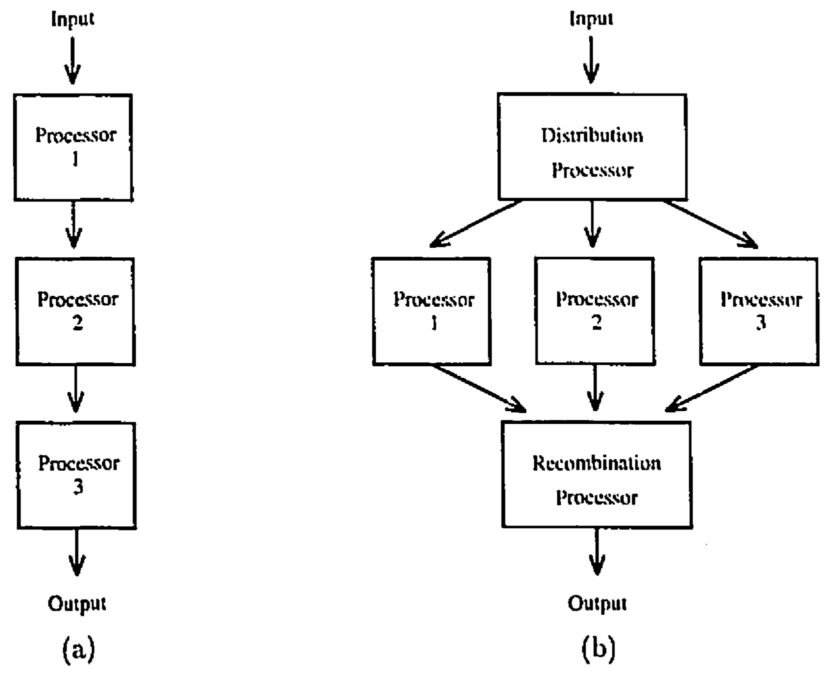
There are two major types of parallelization architectures possible for distributing an algorithm on a MIMD (Multiple Instructions, Multiple Data) network of parallel processors, pipeline processing or distributed data processing. Each of these has its own advantages and disadvantages, which must be weighed in the context of the particular implementation. Analysis of parallel architectures requires a number of factors to be considered. First, there is the *throughput* of the system, which is the rate at which data arrives at the output. Another measure, the *latency*, is the difference between the time data is input to the system and the result, based on this data, appears at the output. It is important to realize that these two measures are quite different. Specific examples of how this difference comes about will be presented shortly. A final factor which affects performance is the distribution of data.

In many cases, time critical bottlenecks occur in the distribution of data from one processor to another and not in the actual processing of the data.

Pipelined systems arrange the task to be parallelized into a number of discrete, individually computable stages, where each stage relies on the output of the previous one. Such implementations are useful when the data must pass through a linear processing stream requiring no feedback. However, pipelined networks are often associated with high latency. For example, given a three-stage pipelined network (Figure 5.10(a)), if each stage processes an image in one second the throughput of the system is one image per second. However, the output at any given time is based on the input three seconds earlier, giving a latency of three seconds. This is of significant concern for active vision systems, since the system should be able to respond to what it is seeing as quickly as possible. There are, however, a number of measures which can be adopted to minimize latency, which will be discussed shortly.

Another issue which must be considered for pipelined networks is that the throughput is determined by the speed of the *slowest* process in the pipeline. This is because all the nodes after the slowest node must wait for it to output its data before they can continue their processing. Also all the nodes before the slowest node must wait until the slowest node is ready to accept new data. This means that in order to maximize throughput and minimize processor idle times, special care must be taken to ensure that processing is divided equally amongst the pipeline nodes. As well, pipelined systems require the entire set of data to be passed from node to node along the pipeline. Depending on the algorithm and hardware, this communication may require a significant amount of the processors resources. However, we note that each individual processor need only store the tables and code required to perform it's specific function. In the case of the interest algorithm, where there is a large amount of data stored in look-up tables, this is an important consideration.

In contrast, a distributed data system achieves its performance improvement not by spreading the processing, but rather by spreading the data that must be processed (Figure 5.10(b)). In this way, each node performs the entire set of processing steps on a particular subset of data. However, such implementations require extra processing to distribute the data to the different nodes, and then recombine the outputs of those nodes. Depending on the data, this division and recombination can be quite complex, requiring significant processor time in itself. However, such networks often result in less latency than pipelined networks. As well, the data distribution requirements at each link are reduced since each node must only receive and send a subset of the total data set. However, this advantage may



(a) Pipelined processing. (b) Distributed data processing.

FIGURE 5.10. Two types of parallel processing architectures

be seriously offset at the distribution and recombination nodes, where many processes must communicate with a single node, increasing the possibility of a bottleneck. As well, since each processor performs the computations of the entire algorithm on a subset of the input data, each must also store all the code and tables required to perform these operations.

The advantages of one parallelization scheme over another must be weighed in terms of the particular algorithm being implemented, and the hardware used. For example, some algorithms may have a natural division of data, leading to trivial distribution and recombination processing. In this case, a distributed data system would be ideal. On the other hand, some algorithms may be naturally divided into a series of processing steps, thus implying a pipelined network. As well, the two methods do not have to be mutually exclusive. A system can use both pipelined and distributed elements in different places of a large network. In fact, this is precisely the approach taken for the parallelization of the interest algorithm.

Application of the interest algorithm naturally leads to two separate processing streams, one for the peripheral data and a second for the foveal data. By processing these two separately, each stream can be optimized for the particulars of its implementation. In addition, the division of these two streams is already accomplished in the retinal mapping nodes of the system [7], and so is available without any further distribution. As well, recombination of these two streams does not require any extra processing, since any implementation would require foveal and peripheral images to remain separate throughout processing.

A pipelined network is chosen within each of the foveal and peripheral streams. There are a number of advantages to be gained by using this architecture. First, the algorithm follows a natural processing pipeline. An image is input and edges are detected. The edge-detected image is then used to apply the inverted annular templates, forming a series of accumulator arrays. These arrays are used to drive the next stage, which determines an interest map based on the values in the accumulator arrays. This interest map is then input to a smoothing process and finally, the local maxima are determined. In contrast, dividing the data into a distributed architecture is not so intuitive, and results in a number of added difficulties. For example, dividing the input image into a number of independently processed subimages leads to problems when edge contributions cross the boundaries of these subimages. Second, the fact that the entire data set must be transferred from processor to processor in a pipelined network is not a problem because the architecture of the C40 processors used permits large amounts of data to be transferred without any CPU intervention. This is due to the flexibility of the on-chip DMA coprocessor, which is self-initializing. Thus it can control complex data transfers to the six communication ports while only requiring synchronization signals from the CPU. Third, by distributing the processing, each node only requires a copy of the look-up tables necessary to perform it's function. This allows more memory to be used to store image data and accumulator arrays, allowing larger images to be processed. The full network used in the parallel implementation of the algorithm is shown in Figure 5.11.

Finally, since this pipeline is four processors deep, the question of latency must be addressed. Having an output four frames behind the input is a serious liability. However, much of the processing can be performed line by line, rather than one image at a time. For example, since the maximum radius of the templates is known, the maximum distance between any output point and a possible contributing edge is also known. This means that as the application of templates progresses line by line through the input image, any lines in the accumulator arrays which can no longer be affected by an edge can be output for processing by the next processor (Figure 5.12). By doing this, the latency in the system is reduced from the time to process four full image frames. The exact latency depends on the actual templates used and the size of the input image.

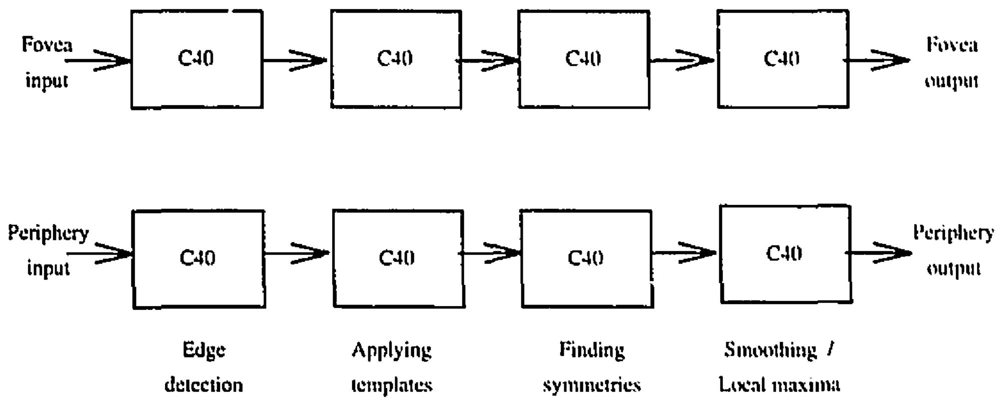
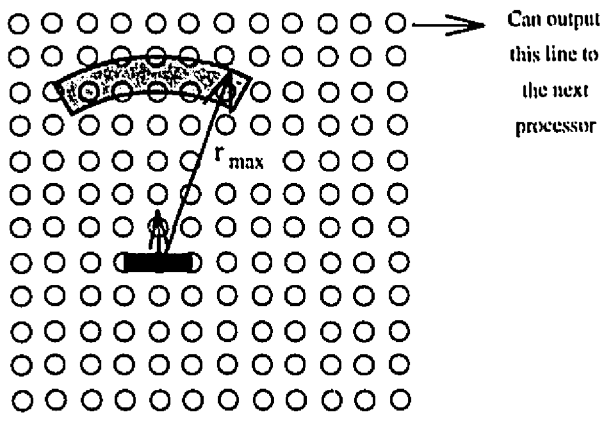


FIGURE 5.11. The parallel processing network



If processing is performed line by line, the top row of pixels can no longer be affected by any edges since the distance between these pixels and the edges yet to be processed is greater than r_{max}. Therefore these pixels can be passed on to the next processor for further processing, even though the image they belong to is not yet completely analyzed by this processor.

FIGURE 5.12. Reducing latency

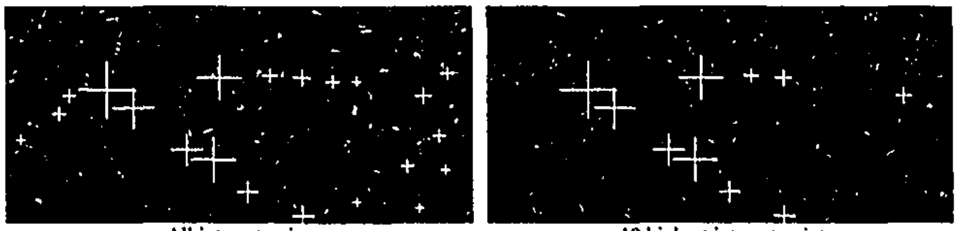
3. Evaluation of the Real-Time Algorithm

This section will examine the results and performance of the real-time algorithm. As such, it is divided into three parts. The first examines the effects of the simplifications to the algorithm made in this chapter. The second section examines the effects of changing the edge threshold and the third considers the processing speed of the network.

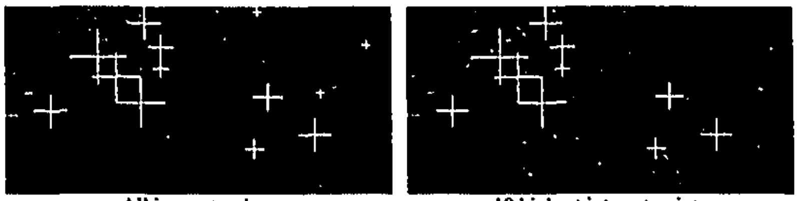
3.1. The effects of the real-time simplifications. In order to evaluate how the simplifications made in this chapter affect the output, the real-time algorithm was applied to the same input images as were used to test the algorithm in chapter 4. In doing this, the same parameters were used to produce the results. These values are $w_1 = 5$ and $w_2 = 5$ for the weighting parameters, and $\sigma = 4$ pixels for the Gaussian smoothing. As well, the same annular templates were used, with the added quantizations required for the real-time implementation. The maximum template radius, r_{max}, was set to the same 10 pixels. As in the previous chapter, the extent of this template is indicated by the circle at the bottom right of each figure. Any variation in the size of this circle from one figure to another is due to different scaling of the images for printing, since the actual size of the template used for all the images in this section is exactly same. Finally, all the results shown in this section were produced with the same edge threshold, set so that 7% of all the pixels in the image are edges. The motivation for this choice will be discussed in the next section.

Figure 5.13 shows the results of the real-time algorithm applied to the same input images as Figure 4.17. A comparison of these two figures shows that many of the resultant interest points are very similar, especially the points of highest magnitude. However, thresholding the edges does affect some aspects of the output. In Figure 5.13(a), applying a threshold produces much fewer spurious interest points in the trees. This is because the edges in this region are quite weak, and thus, below threshold. The ten highest interest points closely follow the path of the wall, with the highest response still corresponding to the towers along the wall. Similarly, for the street scene (Figure 5.13(b)), the points found with the real-time algorithm are very similar to those found in the previous chapter. One notable exception in this case is that the stop sign was not one of the top ten interest points, and in fact, is ranked 12th. This is because some of the edges defining the stop sign are below the selected threshold, resulting in fewer contributions to it's symmetry than were found in the previous chapter. In Figure 5.13(c), the towers and boats in the foreground are found, as in the previous chapter. The interest points found for the planes in Figure 5.13(d) are similar to those found previously, and in fact, less spurious points are now elicited. Similarly, in Figure 5.13(e), the highest interest points correspond to the three towers, as in the previous chapter. Finally, the points found for Figure 5.13(f) are the same as those found previously, but with a few extras corresponding to regions with weak edges that are nevertheless above threshold.

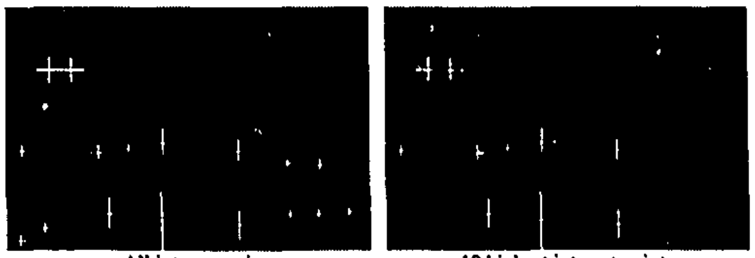
A further comparison of Figure 5.14 with Figure 4.18, and Figure 5.15 with Figure 4.19, shows similar results. The points found by the real-time algorithm, especially those of



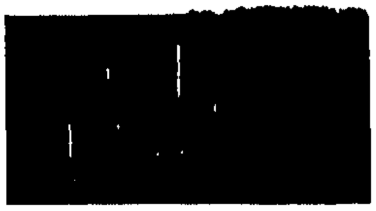
(a)



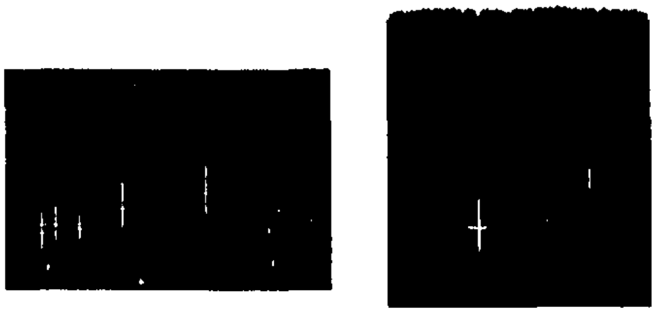
(b)



(c)



(d)



(e) (f)

FIGURE 5.13. Real-time algorithm output on the scenes of Figure 4.17

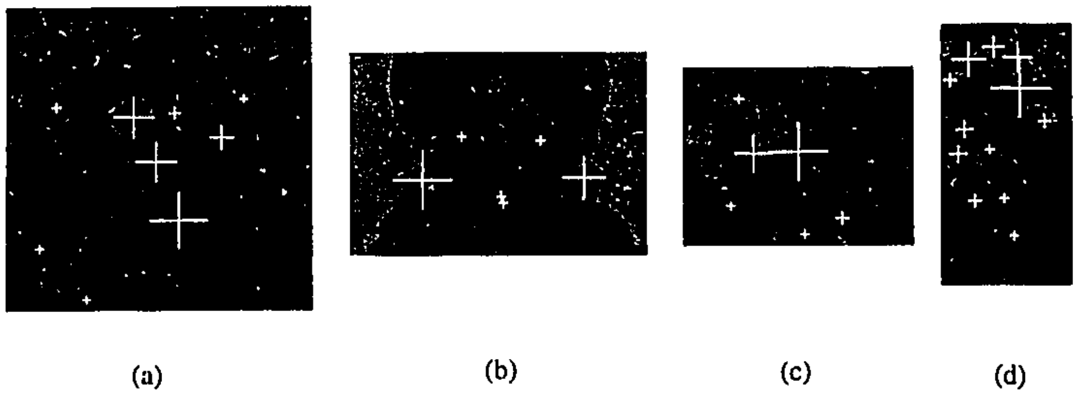


FIGURE 5.14. Real-time algorithm output on the faces of Figure 4.18

highest magnitude, still correspond to the interesting regions of the images and are very similar to the points found in the previous chapter. An exception to this are the images of the face in Figure 5.16. By comparing this to Figure 4.20, it can be seen that in all three of the real-time results the point corresponding to the chin is not found. Furthermore, although the eyes are tracked through Figure 5.16(a) and (b), they are lost in (c). The reason this occurs is because the background in these scenes is quite busy, creating many edges of high magnitude in the image. Setting the edge threshold so that 7% of the edges are above threshold causes the weaker edges of the chin and eyes to be below threshold. This underlines a perennial problem which will be examined in more detail in the next section. That is, how to choose a threshold value that is valid in all circumstances. However, Figure 5.16(a) and (b) still produce strong, valid interest points for the eyes, and even in Figure 5.16(c), where the background is most prominent, an interest point is still found within the face. Finally, valid results are obtained when the real-time algorithm is applied to peripheral images as well. This can be seen by comparing the images of Figure 5.17 with Figure 4.21 and Figure 5.18 with Figure 4.22.

Though there are some differences between the interest points found using the real-time implementation of the algorithm and those found in chapter 4, the results of this section show that the real-time algorithm still produces a valid set of interest points, with many similarities to those found previously. Some differences in the output of the two algorithms was expected, and must be accepted in order to accommodate a more efficient implementation. As well, most of the images tested in this section produced excellent results for the same edge threshold value and parameter settings. This shows that the parameters

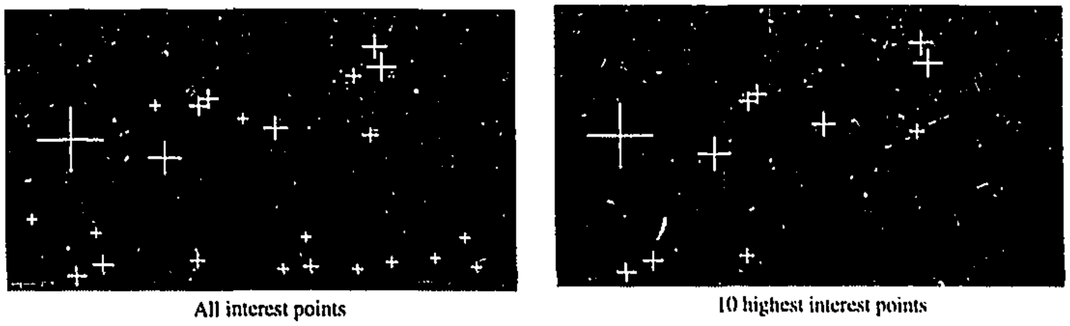


FIGURE 5.15. Real-time algorithm output on the animals of Figure 4.19

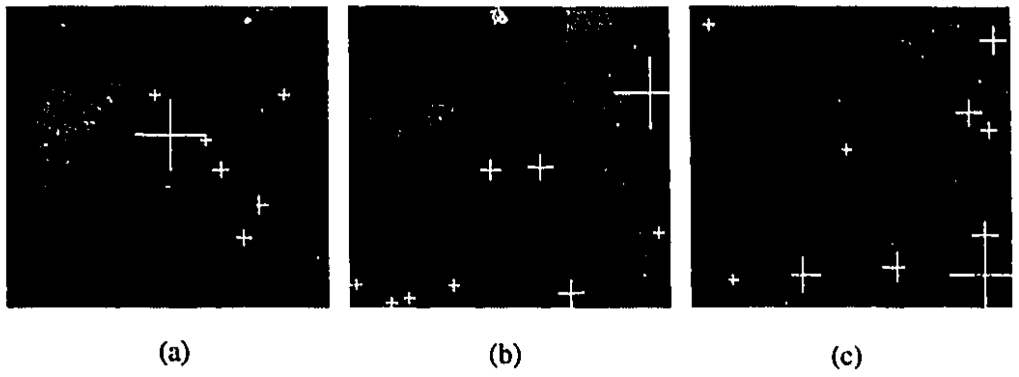


FIGURE 5.16. Real-time algorithm output on a the rotating face of Figure 4.20

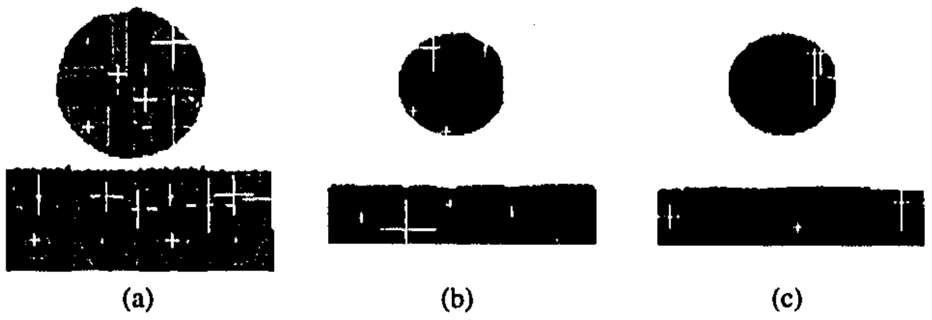


FIGURE 5.17. Real-time algorithm output on the log-polar images of Figure 4.21

can be set to produce valid results in most unconstrained settings without the need for human adjustments.

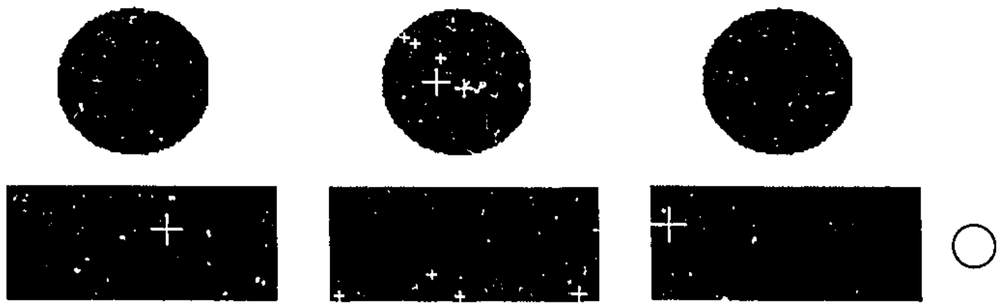


FIGURE 5.18. Tracking the bird of Figure 4.22 with the real-time algorithm

3.2. The effect of the edge threshold. One parameter which must be examined is the edge threshold. Adaptive methods for choosing appropriate edge thresholds have been presented in the literature (see [43] for a review). However, approaches which require excessive computation would not be appropriate for this implementation. As well, since the amount of processing required to implement the algorithm is a function of the number of edges in the image, it is desirable to maintain about the same number of edges above threshold from one image frame to the next. In this way, the processing time would not fluctuate greatly between frames. As such, a threshold for each frame is selected which permits a certain percentage of all the pixels in the image to be above threshold. The latter should be set as small as possible to minimize the number of edges which must be processed. However, too low a setting can eliminate important features.

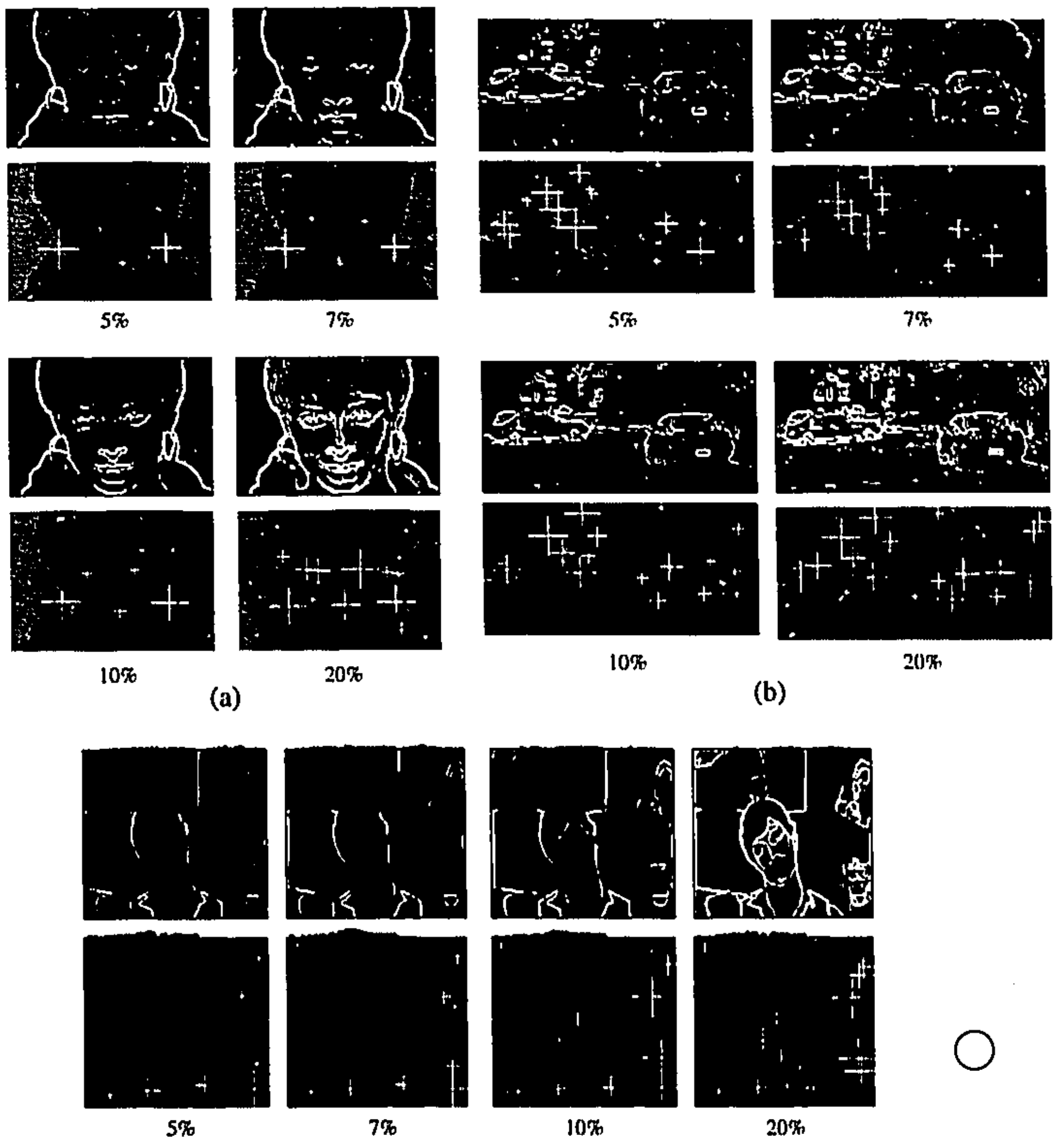
Figure 5.19 shows the output of the algorithm on a number of different images at different edge thresholds. It can be seen from these images that when very few edges are considered, many interesting regions cannot be found. For example, in Figure 5.19(a), when the threshold is at 5%, only the bottom edges of the eyes are found. Since these do not correspond to enclosed regions, they do not produce interest points. The edges at the tops of the eyes are detected with higher thresholds. Thus the interest points corresponding to the eyes are found. However, when the number of edges is set very high (for example, 20%), many spurious interest points are found. These images also explain a number of effects seen in the results presented earlier. For example, the edges around the stop sign in Figure 5.19(b) are quite weak, and really only become prominent when the threshold reaches 20%. This is the reason why the interest point corresponding to the stop sign is not one of the highest 10 points in Figure 5.13(b). Figure 5.19(c) shows why points corresponding to the chin and eyes were not found in Figure 5.14(c). For thresholds less than 10%, there are simply no edges corresponding to these features. However, as the threshold is increased to 10%, the eyes become prominent, and at 20% the chin is marked.

As discussed earlier, it is impossible to choose an edge threshold appropriate to all situations. As well, in order to maximize the algorithm's throughput, we are motivated by a further requirement to have as few edges as possible in the image. By examining the results presented in this section, it is observed that an edge threshold allowing 7% of the total number of pixels in an image to be edges is often a good compromise, yielding good results under a wide variety of scenes[3]. Finally, it should be noted that no look-up table entries depend on the value of the edge threshold. Thus the threshold can be changed "on the fly", with no need to recompute any tables. This allows adjustments to be made to the edge threshold based on real-time feedback from the interest map.

3.3. Performance of the real-time algorithm. Finally, the performance of the real-time algorithm must be examined. This is characterized by two measurements, the throughput and the latency. Figure 5.20(a) shows the average throughput (in frames/second) and latency (in seconds) of the real-time foveal processing over different edge thresholds and image sizes. Ideally, it is desirable to have the throughput as large as possible and the latency as small as possible. As expected, larger images require more processing and data transfers and so result in less throughput and higher latency. As well, the more edges that need to be processed, the slower the algorithm. For a typical case of a fovea of 75 × 75 pixels at a threshold of 7%, the algorithm's throughput is about 8.4 frames/second, with a latency of 0.17 seconds. Decreasing the foveal size to 50 × 50 pixels at 7% threshold increases the throughput to 18.4 frames/second and decreases the latency to 0.11 seconds.

Figure 5.20(b) shows the same graphs for the peripheral processing stream at typical periphery sizes. The effects of image size and edge threshold on throughput and latency are similar to those observed for the fovea. For a typical periphery size of 126 × 36 pixels at 7% edge threshold, the throughput is 7.6 frames/second with a latency of 0.40 seconds. A smaller periphery of 110 × 22 pixels with the same threshold can be processed at 15.7 frames/second, with a latency of 0.34 seconds.

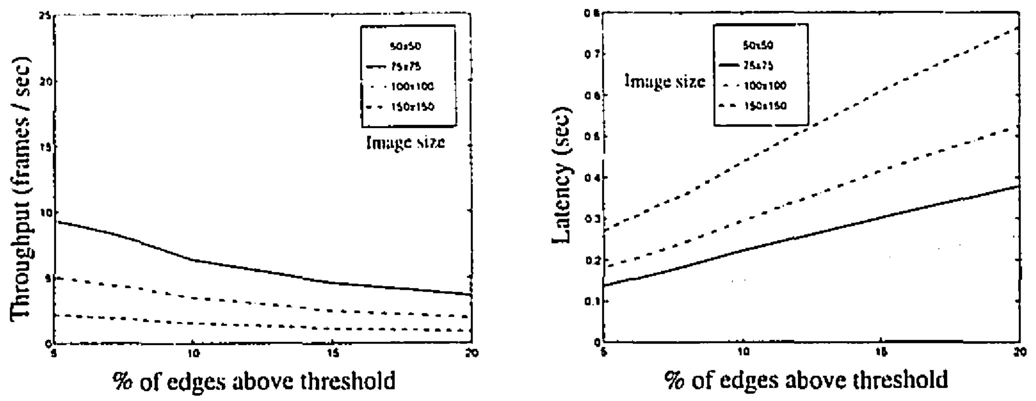
[3]It should be noted here that all the test images used in these experiments were obtained over the internet. Therefore, the sensor and illumination conditions used to produce each of these images would necessarily be different. Even under these circumstances the selected edge threshold is appropriate for most images. However, in actual use, the sensor and illumination factors would be known. Thus the threshold could be calibrated to the particular sensing situation.



These images show the real-time algorithm applied with a variety of edge thresholds. The percentage values indicate what percent of the image pixels are considered edges. The image directly above each output image is its corresponding edge map, with the white points indicating the locations of the edges above threshold.

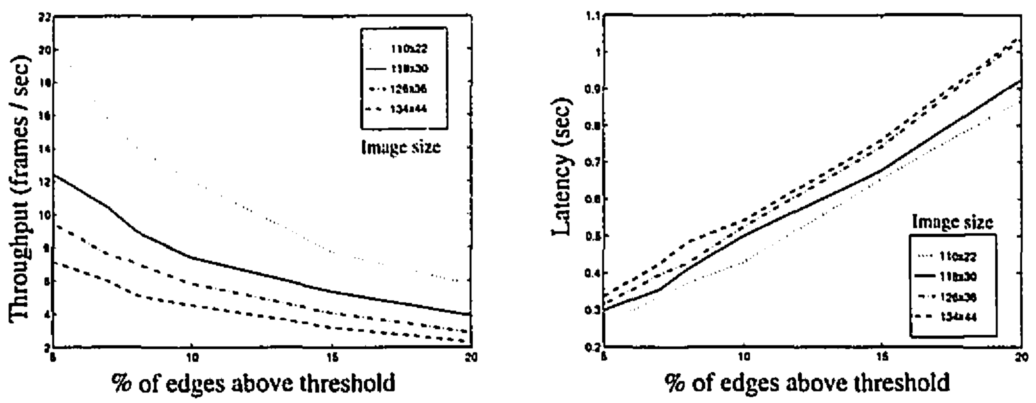
FIGURE 5.19. The effect of threshold selection on the real-time algorithm

Finally, Figure 5.21 shows how the real-time algorithm performs in a real tracking situation. Figure 5.21(a) shows the scene being processed, a person walking along the sidewalk. This original 480 × 480 image is reduced to a 70 × 70 fovea and a 126 × 54 periphery. Figure 5.21(b) shows 6 frames of processing of the real-time algorithm. These



Fovea processing

(a)



Periphery processing

(b)

FIGURE 5.20. Throughput and latency of the real-time algorithm

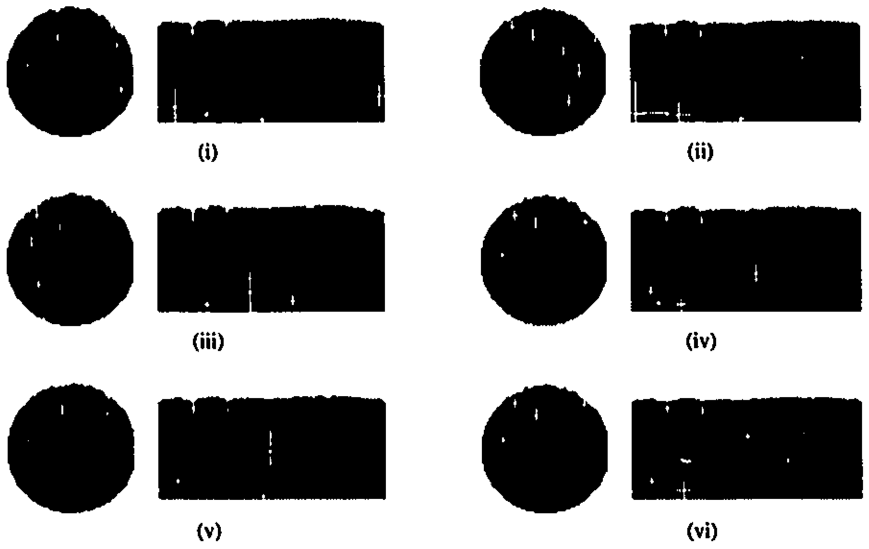
results are obtained with the same template parameters used for all the previous results, and the same edge threshold of 7%.

The first three frames of Figure 5.21(b) show the person moving through the fovea, with a number of interest points following his path. As well, parts of the person extend into the periphery in these frames and are also marked by interest points. In frames (iv), (v) and (vi), the person is no longer in the fovea, but moving up through the centre of the peripheral images. An interest point corresponding to the person is found in all of these images. The potential for tracking the person in real-time is obvious.

In obtaining all of the images in this chapter, and indeed for running the system in general, the display was set up to accept images independently from the foveal and peripheral pipelines. In this configuration, the display always shows the last fully processed frame from the fovea and periphery, with each processing stream updating its image as often as possible. This is necessary because the foveal and peripheral processing streams run independently, and the processing of these streams may occur at different rates. As such, the images of Figure 5.21(b) are representative of the output at 0.25 second increments. However, new foveal data are available every 0.11 seconds (9.1 frames/second) and new peripheral data every 0.19 seconds (5.2 frames/second).

The evidence presented above shows that the real-time algorithm performs well, both in processing time and validity of the interest points obtained. The processing speeds, though less than true video rates, are still suitable for a wide variety of real-world, active vision tasks.

(a)



(b)

(a) The first frame as seen by the camera. (b) 6 frames of the foveal and peripheral processing of the scene.

FIGURE 5.21. A real tracking situation

CHAPTER 6

Conclusions

To function in a real-world environment, an active vision system must have an attentional process to guide its limited computational resources. This thesis has described the development and implementation of such a system, directing overt visual attention in real-time. This system was based on a number of design criteria. First, the algorithm had to be biologically motivated. Second, to be of general use, it needed to rely only on context-free information. Finally, the algorithm had to be computationally efficient, with a fast, real-time implementation.

To begin, a model of overt visual attention, directed by the preceding design criteria, was developed. Based on the psychophysical evidence, this model defines interest points as the intersections of lines of symmetry in an image. To be context-free, these lines of symmetry had to be found directly from the input image, with no need for a priori knowledge or prior image segmentation. This was accomplished by defining lines of symmetry as the loci of all centres of cocircularity of edges in the image. In order to compute this, it is only necessary to obtain the edges from the original image. This can be achieved without any knowledge of the scene. In further agreement with psychophysical findings, the magnitude or saliency of the interest points is a function of both the angle separating the edges contributing to the lines of symmetry and the degree of enclosure of the interest point. Finally, points of interest that are closely spaced are smoothed together, and the final interest points are determined as the local maxima of this smoothed interest map.

Next, the details of implementation of this model were examined. First the discussion addressed the development of annular templates to determine cocircularity. Then the inversion of these templates was considered. This provides a more efficient, Hough-like accumulation method for determining centres of cocircularity. As well, the use of the algorithm for log-polar peripheral images was explored. It was found that the algorithm could be applied directly to these images if an appropriately computed annular template is used and if provisions for wrapping around one of the axes is provided.

Finally, the implementation was adapted to run in real-time. This was accomplished through modifications to the algorithm and by performing computations in parallel over a series of processing nodes. The algorithm modifications stemmed from the use of a binarized, thresholded edge map, which allowed the centres of cocircularity to be found very efficiently. Further simplifications involved reducing the number of edge orientations contributing to the symmetries. This permitted most of the computations of the algorithm to be replaced by a series of look-up tables. As well, changes to the storage structure of the symmetries allowed some computations to be performed through simple Boolean operations.

The results obtained, both by the direct implementation of the model and with the real-time simplifications, highlighted many of the system's capabilities. First, these results were in excellent agreement with the psychophysical evidence on overt attention. As well, the points found by application of the algorithm on real-world scenes showed that the output of the algorithm is in close correspondence to an intuitive notion of what constitutes an interesting region in an image. These results also showed that the algorithm is stable with respect to object rotations, translations and changes in scale. This makes it capable of tracking a moving object in real-world situations. Furthermore, the applicability of this operator to log-polar input images allows this tracking to function over a very wide field of view with excellent processing times. An examination of these processing times showed that, though true video rates are not always achieved, the rates are commensurate with the requirements of active vision systems. For example a typical 70×70 foveal image can be processed at 9.1 frames/second, while, at the same time, a 126×54 peripheral image is processed at 5.2 frames/second. Finally, the results showed that the same algorithm parameters are applicable to a wide range of input images. This means that the algorithm can function in varying real-world environments with the same parameter settings, requiring no human intervention.

Though the algorithm was designed to model overt visual attention with an application to guiding camera positioning, an examination of the results shows that the interest points found can be useful in other applications as well. For example, the algorithm is adept at finding the location of facial features over a large range of scales and poses, making it useful as a context-free front end processor for face recognition tasks. As well, though the algorithm is not able, nor is it intended, to reproduce all the psychophysical effects of covert visual attention, the points found by application of the algorithm on real-world scenes show that the output of the algorithm is in close correspondence to an intuitive notion of what constitutes an interesting region in an image. This means that the algorithm can be used to guide higher level recognition processes, making it appropriate as an algorithm for this type of attention as well. By using the algorithm in this way, a significant reduction (about three orders of magnitude) in the search space can be achieved for a recognition task.

1. Relation to Previous Work

There is only one other algorithm known to the author, the one developed by Reisfeld, Wolfson and Yeshurun [63], which attempts to model overt visual attention. The work presented in this thesis has a number of similarities with their algorithm. Both are motivated by the same psychophysical evidence, which can be observed by comparing results of the two algorithms. As well, both model points of interest based on symmetries in the image. However, there are a number of significant differences. First, the models are based on very different definitions of symmetry. The adoption of centres of cocircularity as the symmetry measure in this thesis produces a much smaller search space for the determination of the symmetries. This results in significant computational savings and the ability to obtain a real-time implementation of the algorithm. Furthermore, the use of this symmetry measure permits the algorithm to be applied directly to log-polar images, since the calculation of centres of cocircularity in these types of images can be accomplished through minor changes to the annular templates used to determine them. In contrast, application of Reisfeld, Wolfson and Yeshurun's algorithm to log-polar images would require modifications to the algorithm itself.

The significant contributions of this thesis are the development of an overt attention model based on centres of cocircularities as symmetries, the adaptation of this model to the log-polar domain, and a real-time implementation of the model through appropriate simplifications, specialized data structures, and parallelization.

2. Direction of Future Work

The next logical step in the research is the completion of the control structure by closing the loop between interest points and the motor control of the camera position. This is currently being studied in our laboratory.

Another area deserving further attention is the incorporation of top-down, context-dependent attentional cues. This would be particularly useful when the algorithm is used to guide camera position, since a single interest point must be selected as the next point of fixation. One possible method of achieving this is to search for objects with particular characteristics. A simple template matching procedure could be employed in a neighbourhood surrounding each interest point. However, examination of the interest algorithm shows that a rich shape description is available through the symmetry and cocircular edge information stored at each point. In fact, recent work on shape representation by Kelly and Levine [38] is dependent on very similar information, and has shown that such information provides a useful representation of shape with many desirable properties.

REFERENCES

[1] J.Y. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active vision. *International Journal of Computer Vision*, pages 334–356, 1987.

[2] I. Amir. Algorithm for finding the centre of circular fudicials. *Computer Vision, Graphics and Image Processing*, 49:398–406, 1990.

[3] R. Arnheim. *Art and Visual Perception*. University of California Press (Berkley), 1974.

[4] F. Attneave. Informational aspects of visual perception. *Psychological Review*, 61:183–193, 1954.

[5] R. Bajscy. Active peceptions vs. passive perception. In *Proc. 3rd IEEE Workshop on Computer Vision*, pages 55–59, 1985.

[6] H. Blum. Biological shape and visual science (part 1). *Journal of Theoretical Biology*, 38:205–287, 1973.

[7] M. Bolduc. A foveated sensor for robotic vision. Master's thesis, McGill University, 1994.

[8] J.M. Brady and H. Asada. Smoothed local symmetries and their implementation. *IJRR*, 3(3):36–61, 1984.

[9] J.H. Bruell and G.W. Albee. Notes toward a motor theory of visual egocentric localization. *Psychological Review*, 62:391–400, 1955.

[10] K. Brunnstrome, T. Lindberg, and J.O. Eklundh. Active detection and classification of junctions by foveation with a head eye system guided by the scale-space primal sketch. pages 701–709. Proc. 2nd European Conf. on Computer Vision, May 1992.

[11] D. Chapman. Vision, instruction and action. Technical Report AI-TR-1204, MIT, April 1990.

[12] J.J. Clark and N.J. Ferrier. Modal control of an attentive vision system. In *2nd International Conference on Computer Vision*, pages 514–523. IEEE, December 1988.

[13] S. Coren. The interaction between eye movements and visual illusions. In D.F. Fisher, R.A. Monty, and J.W. Senders, editors, *Eye Movements: Cognition and Visual Perception*, pages 67–81. 1981.

[14] S. Coren. An efferent component in the visual perception of direction and extent. *Psychological Review*, 93(4):391–410, 1986.

[15] S. Coren and P. Hoenig. Effect of non-targer stimuli upon length of voluntary saccades. *Perceptual and Motor Skills*, 34:499–508, 1972.

[16] S.M. Culhane and J.K. Tsotsos. An attentional prototype for early vision. In G. Sandini, editor, *2nd European Conf. on Computer Vision*, volume 558, pages 551–560, 1992.

[17] H. Deubel, W. Wolf, and G. Hauske. The evaluation of the occulomotor error signal. In A.G. Gale and F.W. Johnson, editors, *Theoretical & Applied Aspects of Eye Movement Research*. Amsterdam (North-Holland), 1984.

[18] A. Dobbins and S.W. Zucker. Endstopped neurons in the visual cortex as a substrate for calculating curvature. *Nature*, 329(6138):438–441, October 1987.

[19] J. Elder. Contour closure and the perception of shape. Master's thesis, McGill University, 1992.

[20] C.W. Eriksen and J.D. James. Visual attention within and around the field of focal attention: A zoom lens model. *Perception and Psychophysics*, 40(4):225–240, 1986.

[21] C.W Eriksen and Y. Yeh. Allocation of attention in the visual field. *Journal of Experimental Psychology: Human Perception and Perfoemance*, 11:583–597, 1985.

[22] L. Festinger, C.W. White, and M.R. Allyn. Eye movements and decrement in the mueller-lyer illusion. *Perception & Psychophysics*, 3:376–382, 1968.

[23] J.M. Findlay. Local and global influences on saccadic eye movements. In D.F. Fisher, R.A. Monty, and J.W. Senders, editors, *Eye Movements: Cognition and Visual Perception*. 1981.

[24] J.M. Findlay, D. Brogan, and M.G. Wenban-Smith. The spatial signal for saccadic eye movements emphasizes visual boundaries. *Perception & Psychophysics*, 53(6):633–641, 1993.

[25] P.J. Giblin and S.A. Brassett. Local symmetries of plane curves. *American mathematical monthly*, 92(10):689–707, 1985.

[26] G.J. Giefing, H. Janssen, and H. Mallot. Saccadic object recognition with an active vision system. In *Proc. International Conference on Pattern Recognition*, pages 664–667, 1992.

[27] B. Gillam. New evidence for closure in perception. *Perception & Psychophysics*, 17(5):521–524, 1975.

[28] M.E. Goldberg and R.H Wurtz. Activity of superior colluculus in behaving monkey. *Journal of Neurophysiology*, 35:560–574, 1972.

[29] M.M. Haith, T. Bergman, and M.J. Moore. Eye contact and face scanning in early infancy. *Science*, 198:853–855, 1977.

[30] P. He and E. Kowler. Saccadic localization of eccentric forms. *J. Opt. Soc. Am.*, 8(2):440–449, Feb. 1991.

[31] D.O. Hebb. *Organization of Behavior*. Wiley (New York), 1949.

[32] C. Ho. An efficient detector for ellipses and circles. *Pattern Recognition*, 1994. Accepted for publication.

[33] D.H. Hubel and T.N. Wiesel. Functional architecture of macaque monkey visual cortex. volume B, pages 1–59. Proc. R. Soc. London, 1977.

[34] J. Jonides. Further toward a model of the mind's eye's movement. *Bulletin of the Psychonomic Society*, 21(4):247–450, 1983.

[35] B. Julesz. Toward an axiomatic theory of preattentive vision. In Edelman, editor, *Dynamic Aspects of Neocortical Function*, pages 585–612. Wiley, 1984.

[36] B. Julesz and J.R. Bergen. Textons, the fundamental elements in preattentive vision and perception of textures. *Bell Systems Technical Journal*, 62:1619–1643, 1983.

[37] L. Kaufman and W. Richards. Spontaneous fixation tendencies of visual forms. *Pecception & Psychophysics*, 5(2):85–88, 1969.

[38] M.F. Kelly and M.D. Levine. The symmetric enclosure of points by planar curves. Technical Report CIM-1-93, McGill Research Centre for Intelligent Machines, 1993.

[39] W. Kessen, P. Salapatek, and M.M. Haith. The visual response of the newborn to linear contour. *Journal of Experimental Child Psychology*, 13:9–20, 1972.

[40] C. Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. In L.M. Vaina, editor, *Matters of Intelligence*, pages 115–141. Reidel, 1987.

[41] D. LaBerge and M.S. Buchsbaum. Positron emmission tomographic measurements of pulvinar activity during an attention task. *Journal of Neuroscience*, 10:613–619, 1990.

[42] Y. Lamdan, J.T. Schwartz, and H. Wolfson. On recognition of 3-d objects from 2-d images. In *Int. Conf. on Robotics and Automations*, pages 1407–1413. IEEE, April 1988.

[43] M.D. Levine. *Vision in Man and Machine*. McGraw-Hill, 1985.

[44] P.J. Locher and C.F. Nodine. Symmetry catches the eye. In J.K. O'Regan and A. Lévy-Schoen, editors, *Eye Movements: From Physiology to Cognition*. Elsevier Science Publishers B.V. (North-Holland), 1987.

[45] D. Marr. *Vision*. Freeman (San Francisco), 1982.

[46] C. Menz and R. Groner. Saccadic programming with multiple targets under different task conditions. In J.K. O'Regan and A. Lévy-Schoen, editors, *Eye Movements: From Physiology to Cognition*. Elsevier Science Publishers B.V. (North-Holland), 1987.

[47] R. Milanese. *Detecting salient regions in an image: from biological evidence to computer implementation*. PhD thesis, University of Geneva, 1993.

[48] M. Mishkin, L.G. Ungerleider, and K.A. Macko. Object vision and spatial vision: two cortical pathways. *Trends in Neuroscience*, pages 414–417, October 1983.

[49] J. Moran and R. Desimone. Selective attention gates visual processing in the extrastriate cortex. *Science*, 229:782–784, 1985.

[50] H.P. Moravec. Obstacle avoidance in navigation in the real world by a seeing robot rover. Technical Report AIM-340, Stanford, 1977.

[51] V.B. Mountcastle, J.C. Lynch, A. Georgopoulos, H. Sakata, and A. Acuna. Posterior parietal association cortex of the monkey: command functions for operations within extrapersonal space. *Journal of Neurophysiology*, 38:871–908, 1975.

[52] H. Muammar and M. Nixon. Approaches to extending the hough transform. volume 3, pages 1556–1559. ICASSP, 1989.

[53] U. Neisser. *Cognitive Psychology*. Appleton (New York), 1967.

[54] S. Peleg, O. Federbush, and R. Hummel. Custom made pyramids. In L. Uhr, editor, *Parallel Computer Vision*, pages 125–146. Academic Press (New York), 1987.

[55] W. Pitts and W.S. McCulloch. How we know universals: The perception of auditory and visual forms. *Bulletin of Mathematical Biophysics*, 7:89–93, 1947.

[56] M.I. Posner and S.E. Petersen. Localization of cognitive operations in the human brain. *Science*, 240:1627–1631, 1988.

[57] M.I. Posner and S.E. Petersen. The attention system of the human brain. *Annual Review of Neuroscience*, 13:25–42, 1990.

[58] R.M. Pritchard. Visual illusions viewed as stabilized retinal images. *Q. Journal of Experimental Psychology*, 10:77–81, 1958.

[59] D.R. Proffitt and J.E. Cutting. Perceiving the centroid of configurations on a rolling wheel. *Perception & Psychophysics*, 25:389–398, 1979.

[60] D.R. Proffitt and J.E. Cutting. Perceiving the centroid of curvilinearly bounded rolling shapes. *Perception & Psychophysics*, 28(5):484–487, 1980.

[61] D.R. Proffitt, M.A. Thomas, and R.G. O'Brien. The roles of contour and luminance distribution in determining perceived centers within shapes. *Perception & Psychophysics*, 33(1):63–71, 1983.

[62] D. Rafal and M.I. Posner. Deficits in human visual spatial attention following thalamic lesions. volume 84, pages 7349–7353. National Academy of Science, 1987.

[63] D. Reisfeld, H. Wolfson, and Y. Yeshurun. Detection of interest points using symmetry. Technical Report 177/90, Computer Science Department, Tel Aviv University, 1990.

[64] D. Reisfeld and Y. Yeshurun. Robust detection of facial features by generalized symmetry. In *11th IAPR International Conference on Pattern Recognition*, volume I, pages 117–120, 1992.

[65] C.S. Richards. Closure and gestalt notions on the visual memory of form: a review. *Journal of General Psychology*, 93:95–113, 1975.

[66] W. Richards and L. Kaufman. Centre-of-gravity tendencies for fixations and flow patterns. *Perception and Psychophysics*, 5(2):81–84, 1969.

[67] I. Rock. *The Logic of Perception*. MIT Press (Cambridge), 1983.

[68] A.S. Rojer and E.L. Schwartz. Design considerations for a space-variant visual sensor with complex-logarithmic geometry. In *Proc. ICPR*, pages 278–285, 1990.

[69] P. Salapatek and W. Kessen. Prolonged investigation of a plane geometric triangle by the human newborn. *Journal of Experimental Child Psychology*, 15:22–29, 1973.

[70] P.A. Sandon. Simulating visual attention. *Journal of Cognitive Neuroscience*, 2(3):213–231, 1990.

[71] G.L. Scott, S.C. Turner, and A. Zisserman. Using a mixed wave/diffusion process to elicit the symmetry set. *Image and Vision Computing*, 7(1):63–70, 1989.

[72] S. Shaw and M.D. Levine. Visual information processing in primate cone pathways: Part I, a model. *IEEE Trans. on Systems, Man and Cybernetics*. Accepted for publication.

[73] S. Shaw and M.D. Levine. Visual information processing in primate cone pathways: Part II, experiments. *IEEE Trans. on Systems, Man and Cybernetics*. Accepted for publication.

[74] D.L. Sparks. Translation of sensory signals into commands for control of saccadic eye movements: role of primate superior colliculus. *Physiological Review*, 66(1):118–171, 1986.

[75] S. Tölg. Gaze control for an active camera system by modeling human pursuit eye movements. In *Intelligent Robots and Computer Vision XI: Algorithms, Techniques and Active Vision*, volume 1825, pages 585–598. SPIE, 1992.

[76] J.T. Townsend. A note on the identification of parallel and serial processes. *Perception & Psychophysics*, 10(161-163), 1971.

[77] J.T. Townsend. Serial vs. parallel processing: sometimes they look like tweedledum and tweedledee but they can (and should) be distinguished. *Psychol. Sci.*, 1:46–54, 1990.

[78] A.M. Treisman. Preattentive processing in vision. *Computer Vision, Graphics, and Image Processing*, 31:156–177, 1985.

[79] A.M. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 12:97–136, 1980.

[80] A.M. Treisman and R. Paterson. Emergent features, attention and object perception. *Journal of Experimental Psychology: Human Perception and Performance*, 10:12–31, 1984.

[81] A.M. Triesman and S. Gormican. Feature analysis in early vision: evidence from search asymmetries. *Psychological Review*, 95:15–48, 1988.

[82] V. Virsu. Tendencies to eye movement, and misperception of curvature, direction, and length. *Perception & Psychophysics*, 9(1B):65–72, 1971.

[83] R.H. Wurtz, M.E. Goldberg, and D.L. Robinson. Behavioral modulation of visual responses in the monkey: stimulus selection for attention and movement. *Progress in Psychobiology and Physiological Psychology*, 9:43–83, 1980.

[84] R.H. Wurtz and C.W. Mohler. Enhancement of visual response in monkey striate cortex and frontal eye fields. *Journal of Neurophysiology*, 39:766–772, 1976.

[85] H. Yamamoto, Y. Yeshurun, and M.D. Levine. An active foveated vision system: Attentional mechanisms and scan path convergence measures. *CVGIP: Image Understanding*, 1994. Accepted for publication.

[86] Y. Yeshurun and E.L. Schwartz. Shape description with a space-variant sensor: algorithm for scan-path fusion, and convergence over multiple scans. In *Trans. Pattern Analysis and Machine Intelligence*, pages 1217–1222. IEEE, 1989.

[87] R.K.K Yip, P.K.S. Tam, and D.N.K. Leung. Modification of Hough transform for circles and ellipses detection using a 2-dimensional array. *Pattern Recognition*, 25(9):1007–1022, 1992.

[88] T. Zielke, M. Brauemann, and W. von Seelen. Cartrack: computer vision-based car-follower. In *IEEE Workshop on Applications of Computer Vision*, pages 156–163, 1992.