Copyright

by

Thomas Louis Arnow

2008

The Dissertation Committee for Thomas Louis Arnow Certifies that this is the approved version of the following dissertation:

Foveated Object Recognition by Corner Search

Committee:

Alan C. Bovik, Supervisor

Wilson S. Geisler III, Co-Supervisor

Joydeep Ghosh

H. Grady Rylander III

Charles H. Roth

Baxter Womack

Foveated Object Recognition by Corner Search

by

Thomas Louis Arnow, Bsc.; M.S.; M.S.

Dissertation

Presented to the Faculty of the Graduate School of
The University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

The University of Texas at Austin
May, 2008

Dedication

Rem tene; verba sequentur

Marcus Porcius Cato

ACKNOWLEDGEMENTS

I thank committee for time and effort spent on this project, my wife Maureen, for moral support, money, patience, and other kinds of assistance. Sandra Ballard of the English department of Appalachian State University also provided moral support. My son David and cat Something Wicked provided comfort and encouragement. Professor Alan C. Bovik my advisor spent an enormous amount of time on this effort as well as introducing me to machine vision. Professor Wilson S. Geisler III, my co-advisor introduced me to human vision. I had been interested in this field back in high school when upon reading about color television, I found out that the three primary colors are red, green, and blue, though we had been taught red yellow, and blue.

People at LIVE, especially Kalpana Seshadrinathan and Umesh Rajashekar provided valuable assistance with the eyetracker and provided references and test images. Kalpana helped me prepare papers for publication helped me get signatures.and answered all sorts of questions I had, not being around the campus.

Tanya Payne read early versions of this document and her husband Mark May lent me a tie for the qualification exam. My doctors Kimberly Heller, Juanita Sprute, Timothy Hlavinka, Charles Machell, and Keiko McManus preserved my health during this project.

Foveated Object Recognition by Corner Search

Publication No._____________

Thomas Louis Arnow, PhD.
The University of Texas at Austin, 2008

Supervisors: Alan C. Bovik
Wilson S. Geisler

Here we describe a gray scale object recognition system based on foveated corner finding, the computation of sequential fixation points, and elements of Lowe's SIFT transform. The system achieves rotational, transformational, and limited scale invariant object recognition that produces recognition decisions using data extracted from sequential fixation points. It is broken into two logical steps.

The first is to develop principles of foveated visual search and automated fixation selection to accomplish corner search. The result is a new algorithm for finding corners which is also a corner-based algorithm for aiming computed foveated visual fixations. In the algorithm, long saccades move the fovea to previously unexplored areas of the image, while short saccades improve the accuracy of putative corner locations. The system is tested on two natural scenes. As an interesting comparison study we compare fixations generated by the algorithm with those of subjects viewing the same images, whose eye movements are being recorded by an eyetracker. The comparison of fixation patterns is made using an information-theoretic measure. Results show that the algorithm is a good locator of corners, but does not correlate particularly well with human visual fixations.

The second step is to use the corners located, which meet certain goodness criteria, as keypoints in a modified version of the SIFT algorithm. Two scales are implemented. This implementation creates a database of SIFT features of known objects. To recognize an unknown object, a corner is located and a feature vector created. The feature vector is compared with those in the database of known objects. The process is continued for each corner in the unknown object until enough information has been accumulated to reach a decision.

The system was tested on 78 gray scale objects, hand tools and airplanes, and shown to perform well.

Table of Contents

ACKNOWLEDGEMENTS V

List of Tables x

List of Figures xi

CHAPTER 1.

INTRODUCTION

1.1 Foveated visual search. 1

1.2 Object recognition. 7

CHAPTER 2.

DETECTING CORNER FEATURES IN FOVEATED IMAGES 11

2.1. Foveation Filtering 11

2.2. Foveated Edge Detection 14

2.3. Detection of Corners 16

2.4 Sequential Computation of Fixation Points 19

2.5 Assessing the Fixation Points 24

2.6 Method of Comparison 26

2.7. Comparison with Handpicked Corners 29

2.8. Comparison With Eyetracked Fixations 29

CHAPTER 3.

TWO-DIMENSIONAL FOVEATED OBJECT RECOGNITION 41

3.1 Foveated Edge and Corner Features 44

3.2 Sequential Computation of Fixation Points (redux) 45

3.3 Foveated Object Recognition By Corner-Based SIFT 48

3.4 Review of SIFT 48
    3.4.1 Keypoint localization. 48
    3.4.2 Orientation invariance. 49
    3.4.3 Keypoint descriptor. 49
    3.4.4 Object recognition. 49
3.5 SIFT-Like Preprocessing Using Foveated Corners.. 50
3.6 Recognition Using Foveated Corners 55

CHAPTER 4.

RESULTS AND DISCUSSION **60**
4.1. Concluding Remarks 64
Appendix 66
Bibliography.. 67
Vita ……………………….. 82

List of Tables

Table I. KLD values for eyetracker vs. algorithm and eyetracker vs. handpicked 50 fixations. 32
Table II. KLD values for algorithm vs. handpicked for 50 and 85 fixations. 32
Table III. Object categories and two-letter ID codes. The number of each object type is indicated, as well as the number used from each object type for training purposes. Finally, the number of correctly recognized objects in each dataset (and the percentage of correctly-recognized objects) are given. 63

List of Figures

Figure 1. Generation of curvature map on smoothed zero-crossing contour. (*a*) Zero-crossing contour; (*b*) Local polynomial fit near the high-curvature point A. The axis coordinates are relative to A. (*c*) Local polynomial fit near the low-curvature point B. The axis coordinates are relative to B. (*d*) Zero-crossing contour in (a) with curvature coded as intensity (darker = higher curvature). 21
Figure 2. Example of calculation of subsequent fixations. (*a*) Original image lighthouse. (*b*) Foveated version by space-varying Gaussian filtering. (*c*) Foveated edge map by foveated Canny edge detection. (*d*) Foveated curvature map. 25
Figure 3. Sample fixation points with Gaussian interpolation. 28
Figure 4. Polyhedron image used to compare the corner detection algorithm with handpicked corners for 85 fixations. (a) raw image(*b*) For $\tau = 0.1$, the KLD = .1323. (*c*) For $\tau = 0.2$, the KLD = .1490. (*d*) For $\tau = 0.3$, the KLD = .3702. 30
Figure 5. A comparison of the accuracy of corner locations, algorithm vs. eyetracker 50 fixations. 33
Figure 6. Four observers vs. 50 algorithmic fixations – lighthouse. 35
Figure 7. Four observers vs. 50 algorithmic fixations – polyhedron. 37
Figure 8. Test of algorithm on tool image (top) and natural scene (bottom). 38
Figure 9. Test of algorithm on other images - part A. 39
Figure 10. Test of algorithm on other images. 40
Figure 11. Two corners located on a rectangle. 51
Figure 12. Illustration of orientation pre-processing. (*a*) (*b*) Foveation-filtered versions of corners 1 and 2 in Figure 11, respectively. (*c*), (*d*) Needle diagrams showing the gradients of (*a*), (*b*), respectively, with needle length coding (scaled) gradient magnitude and needle angle indicating gradient orientation. 53
Figure 13. Rotation of corners. (*a*), (*b*) Histograms of gradient orientations demonstrated on Corners 1 and 2. The neighborhood orientations computed on (*a*), (*b*) using (6) were -5.0° and -114.3°, respectively. (*c*), (*d*) Gradient maps of the two corners following rotation. 54
Figure 14. Illustration of the corner-keypoint database. Keypoints are shown as 2-D images. Objects labeled are X1, X2, and Y1. Here, X1 and X2 are examples of the same type of object and are adjacent in the database, while Y2 is a different type. Object X1 has three corners labeled *a* through *c*; X2 and Y1 each have two corners, labeled *a* and *b*. 56
Figure 15. Sample images of each category. (*a*) A brush; (*b*) a fighter jet; (*c*) a claw hammer; (*d*) a passenger jet; (*e*) a pair of pliers; (*f*) a screwdriver; (*g*) a sledge hammer; (*h*) a wrench. 62

CHAPTER 1.

INTRODUCTION

1.1 Foveated visual search.

One of the most difficult, ill-posed, and unsolved problems in the field of image analysis is that of visual search. Indeed, the problem remains poorly defined from an engineering perspective. Does visual search mean an algorithm for finding and identifying a specific object or class of objects in an image? The extensive literature on automated target recognition (ATR) exemplifies this philosophy [29]. Or, does visual search imply a general framework for finding information from visual data, but without object-specific guidance? There is only a small literature on generic automated visual search. Methods include search based on contrast [40], [41]; image and depth gradients [10]; other edge factors [42]; proximity between objects [43]; object similarity [44]-[46], and combinations of randomized saliency and proximity factors [47]. These automated methods, while reaching in interesting directions, remain generally unsuccessful, although active, directed search methods show promise in reducing the complexity of this severely ill-posed problem [48], [49].

While there are merits to both strategies, great benefit would result from the development of basic principles guiding the design of algorithms for visual search, which could be applied to a diversity of search applications, and which would address some of the factors that limit the success of visual search.

Indeed, primate and other biological vision systems have taken this approach, at least at the mechanical and data sampling level. A striking feature of primate and other animal visual systems is that they are foveated, with high resolution near the center of gaze that falls off as a power function of eccentricity, the angle away from the center of gaze. In humans, the fovea is a circular region of tightly packed cones, roughly 1.5 mm in diameter [3]. This density decreases rapidly with eccentricity. In the central fovea, receptors are packed at a density of about 120/degree [12] [13]. This corresponds to a resolution of .291 mm at a viewing distance of 100 cm. The optics of the eye filters out higher spatial frequencies, which could cause aliasing [14].

Foveated primate vision systems mechanically direct the fovea around a scene over time via very fast ballistic eye movements called saccades, resulting in series of static fixations [7], [11]. Foveation is an effective compromise between the demand for high-resolution vision and the limited transmission and processing bandwidths of the optic nerve and subsequent brain regions; foveation is a powerful form of visual data compression - the amount of information flowing from the retina to the brain is far less than if the entire retina was sampled at foveal density.

The brain uses peripheral, low-resolution information to decide which region of the image warrants the next fixation. This is accomplished quickly – the human eye typically makes more than 10,000 saccades per hour, ranging in distance from a few seconds of arc to up to over 70º [16], [13], [12], [15]. Certainly the computation of new fixations must be fast, automatic, and image-driven to accomplish visual search with active, mobile cameras or eyes [8], [9], [10].

Rather than processing a wide field of view (FOV) visual stream all at once, high-spatial-resolution search is conducted over a very small FOV (the image on the fovea) while wide-FOV search, rich with context but lacking detail, occurs over the peripheral field of view. Candidate discoveries in the periphery can be rapidly analyzed at high resolution via saccadic eye movements that redirect the candidate to an area of interest; in the absence of candidates, eye movements may be sequentially deployed to enlarge the search space.

Much more research has been applied to the problem of how visual search is accomplished by primate and other biological vision systems. Results from the cognitive and perceptual sciences provide interesting insights into how humans search visual environments [50]-[52]. These studies have revealed limitations imposed by low-level factors [52] and the relationship between stimuli and the distribution of attention [50]. Other studies have revealed the rules regarding where a saccade will land on a complex form [53]. Several workers studied the problem of integrating information across eye movements [54]-[58], relating shifts in attention and gaze [59], [60], relating top-down and bottom-up search [61]-[62], and relating visual search with visual memory [63]-[65]. Yet little is known about the tremendous amount of learning and plasticity needed to efficiently search for objects in complex visual environments. Only recently has the influence of learning and memory loads on search been investigated [66]. Little is known about fixation-selection mechanisms, how attention is distributed over time [67], and how these mechanisms maximize visual search efficiency.

In any case, it is clear that visual information gathering and visual search is greatly augmented by deploying the highly efficient foveation-fixation-scanpath process. We believe that this elegant solution can, and should, be adapted into computational systems for visual information acquisition and processing. Most practical image processing systems, however, do not operate with mobile cameras, which means that the role of foveation in such systems takes a modified role. Such foveated systems that operate without moving cameras we shall call static foveated systems. Instead of the foveation being determined by the fixation of the acquisition hardware (camera or eye), it is accomplished in software according to some criteria. One powerful and popular example is foveated image compression, where images or videos are foveated to achieve substantially increased compression [8], [9], [22]. This requires knowing where the spatial fixation on the image of the human observer is, and the distance of the observer from the image, so that the foveated fall-off can be matched to that of the observer's eye. This can be accomplished by eyetracking, head tracking and other physical measurements [22].

Less work has been done on foveated computer processing algorithms that do not require eyetracking. Exceptions include early work by Burt [30] on scene analysis and Klarquist et al. [10] on computational stereopsis. Broadly speaking, the idea is to allocate dense visual data representation and processing resources to those regions of the image which seem to have promising information, while applying fewer resources to the peripheral data processing – while retaining potentially valuable peripheral information which may guide further fixations and processing.

Biological visual systems that perform visual search certainly benefit from the mechanical fixation-foveation process. We believe that automated systems will realize similar benefit by the use of static foveated processing – even in the absence of moving cameras, and without the benefit of eyetracked human observers. However, since there is no well-developed theory of visual search – foveated or otherwise – we must begin from scratch. While there is no general agreement on how visual search is conducted, there is support for the notion that it contains both "bottom-up" elements as well as "top-down" elements. Top-down processing suggests that algorithms should retain internal models of what is being searched for, and that the search process becomes essentially that of matching these models to the image on a local basis. Bottom-up processing supplements this concept by the idea that these internal models are constructed from simple features, such as contrast, contours, surfaces, disparities, colors, etc.

In foveated visual search systems, there is also the interesting central problem of deciding where to place the center of foveation during the search process. In such systems, the goal of the placement is not to match the position of gaze, but rather, to optimize the gathering of information that is likely relevant to the object(s) being searched for. Subsequent fixations should be chosen based on the available foveated data. The amount of information available to the search algorithm regarding the object(s) of interest (assuming it is present in the image) is then determined by the proximity of the object to the current fixation. There is some evidence that in human visual search, the selection of next fixations is effected by such low-level features as contrast [31], [32], and also by primitive shape attributes [33], [34]. However, the visual psychophysics literature, while certainly more advanced on the topic of foveated visual search than the computational literature, still supplies little guidance towards the development of computational algorithms.

We believe that both high- and low-level factors are necessary for visual search, but that low-level features are a pre-requisite to high-level modeling. Determining which low-level features are best utilized is an open problem that will require reconciling high- and low-level issues. In this paper, we address both issues by proposing an approach to foveated search of low-level features, specifically corners - points of locally maximum contour curvature, and discontinuities in contour orientation.

Corners have long been recognized as rich bearers of visual information, and numerous algorithms have been proposed for detecting corners and using them as features in basic visual tasks such as object recognition, stereo matching, shape analysis and optical flow computation [1], [4]-[6], [17]-[19], [35]-[39]. In their early seminal work on computational vision, Marr and Hildrith [2], [3], regarded corners as being of high visual saliency, and designated them as being members of the discontinuity class in the theory of the full primal sketch. They viewed corners as an important member of the class of image primitives that are used as building blocks for representing objects in image understanding systems, whether biological or computational. Other features exist, of course, such as edges. However, corners are more localized than edges, and as pointed out by Nobel [1], are superior to edges for defining the shapes of objects, since edges detectors only provide location information in a single direction (normal to the edge). Shi and Tomasi [72] derived a simple model to determine which features are best for tracking in a video signal. They determined that corners belong to this class. Tommasini *et. al.* [73] extended their work by adding an algorithm to reject unreliable features. Schmidt *et. al.* [76] discuss the detection of "interest points" - intensity changes in 2-D that include corners, T-junctions, dots and other features - and evaluate their usefulness for image registration. Kenney *et. al.* [77] derive a "condition number" to assess the sensitivity of feature/corner detectors to perturbations in feature position. Gordon and Lowe [79] used a scale invariant feature transform [80] to extract features defined as the extrema of a scale-varying Difference of Gaussian (DoG) convolved with the image. Features detected in this manner included edges and corners.

In a paper with a general philosophy similar to ours, Reid and Murray [81] describe a method of obtaining a fixation point on a moving object in an active vision system using two or three cameras. They track corners in real time over a cluster of frames using a Kalman filter. Another feature of their system is a simple psuedo-foveated processing scheme with a small psuedo-fovea surrounded by a lower-resolution psuedo-periphery [82].

Certainly corners present advantages as a discrete image feature, since they are simultaneously information-rich, yet require minimal description. Accurate corner information is not easy to acquire; for example, Mehrotra et al. [4] points out that edge detectors tend to perform poorly near corners, suggesting that corner detection by locating intersections between edges can lead to poor performance.

Here we cast the problem of corner detection as a corner search process. We apply principles of foveated visual search and automated fixation selection in accomplishing the corner search. Thus, we approach the search process from a low level, searching for objects without requiring building blocks to represent them, since the objects being searched for are the same as the features. In this way, we hope to contribute by supplying a case study of both foveated search, and foveated feature detection. The result is a new algorithm for finding corners (viewed from the perspective of foveated feature detection), but which may also be considered as a corner-based algorithm for aiming computed visual fixations (along with a computed fovea), with the eventual goal of extracting information that is useful for more sophisticated object recognition systems.

With this last interpretation in mind, as an interesting comparison study we also compare fixations generated by this algorithm with those of subjects viewing the same images, whose eye movements are being recorded by an eyetracker. The comparison is made using an information-theoretic measure.

1.2 Object recognition.

Understanding and modeling object recognition remains one of the principal unsolved problems in both computational and biological vision. It is necessary both for the survival of living organisms as well as to advance vision-based robotics and has been researched for decades. Computer approaches to object recognition begin with a digitized image stored as an array of pixels but rapidly diverge from that common starting point. If the image can be converted to a binary silhouette, then relatively simple methods may be applied. Examples are invariant moments such as Hu [84] and Zernike [85], which are rotational, translational, and scale invariant. Of course, the ultimate goal would be to recognize objects in a natural scene with a confidence level approaching or exceeding that of living organisms. Since binarizing an image of an unconstrained natural scene seldom yields usable results, more general methods are called for. A popular framework is the Scale Invariant Feature Transform (SIFT) [80], which locates low-level features in an image, then creates rotationally invariant maps of gradients about each point, at a series of varying scales.

In this paper we propose a gray-scale object recognition system that achieves recognition by combining an essential aspect of natural vision called foveation with corner finding and a modified implementation of SIFT. Foveation refers to a vision system implementation that has a varying spatial resolution mimicking that of the human eye. The finest resolution occurs at the center of gaze but falls off drastically with eccentricity, or angular distance from the center of the fovea. In humans, the fovea is a circular region of tightly packed cones, roughly 1.5 mm in diameter [3]. Outside the fovea, packing density decreases rapidly with eccentricity. In the central fovea, receptors are packed at a density of about 120/degree [12], [13], corresponding to a resolution of .291 mm at a viewing distance of 100 cm. The optics of the eye filter out high spatial frequencies, effectively preventing aliasing [14]. Foveated vision is described in detail in [9]-[11],[15].

Foveation is part of a remarkable natural engineering solution that allows human (and other) organisms to selectively deploy visual resources to regions of interest. By mechanisms of attention the head and eyes are moved in order to place the direction of gaze, and hence, high-resolution visual resources beginning at the fovea, at points of interest. Each steady position of the fovea is called a fixation point. Even with the head steady, the primate eye scans a scene via very fast movements called saccades – typically more than 10,000 per hour, resulting in series of static fixations [11].

Foveation and fixations mediate the conflicting needs for high-resolution vision and the limited transmission and processing bandwidths of the optic nerve and subsequent brain regions. Certainly, these mechanisms create significant efficiencies as the primate organism undergoes daily tasks, such as visually searching for, and recognizing objects. Such efficiencies also have great potential for computational vision as well. For example, robotic vision systems that deploy moving cameras with foveated sensing by intelligent fixation strategies should prove to be highly flexible and efficient. Of course, much work remains to be done on developing protocols for deciding protocols for deciding automatic visual fixations [10], [86].

Foveation can also play an effective role for analyzing images that are not being taken by mobile cameras. For example, images and videos coded for viewing by the foveated human eye can achieve significant gains in compression [22], [68], [24], communication throughput [8], [110], stereopsis [10], [110], and for visual search of low levels image features, such as edges and corners [87].

It is our view that foveation and fixation selection are principles that will find significant utility for improving a wide variety of image processing and analysis applications. In this paper, we explore the use of foveated visual search as a way of making object recognition systems more efficient. Specifically, we develop a scale- and rotation-invariant two-dimensional object recognition system that operates on gray-scale images of of simple candidate objects to be recognized. The modus operandi develops in two stages: first, corners are discovered using a foveated corner search algorithm developed in [87]; then these corners are used in a SIFT-like recognition algorithm. The recognition algorithm is applied incrementally as the corners are discovered, so that only as many corners are found as are needed to effect recognition. In this way, efficiencies are gained both in the corner search process and in the recognition process.

CHAPTER 2.

DETECTING CORNER FEATURES IN FOVEATED IMAGES

While foveation presents significant advantages for visual search via an efficient allocation of resources, it presents new challenges for accomplishing low-resolution and spatially-varying object recognition, since each foveated view distorts the image away from fixation by reducing the resolution. Near the fixation point, fine features, such as edges and corners, are resolved well. Away from the fixation point, these fine details may be attenuated, distorted, or lost.

The overall approach that we will take towards searching for corners in images will involve foveating the image, deciding a most likely location of a corner, moving the fixation to that vicinity, refining the corner location estimate, identifying the corner – then choosing a next likely corner location, and so on. The details of the overall search methodology will be given later. An essential ingredient for choosing likely corner locations is a corner detection algorithm that operates on foveated data, and the output of which can be analyzed and interpreted in the context of foveation.

In the following we describe, in the following order, the method we use to create foveated images; the method of edge detection we use on foveated images, and the method of corner detection we use on the detected foveated edge maps.

2.1. Foveation Filtering

There are several possible methods for creating foveated images, the most popular are those based on spatial-domain *foveation filtering,* and those based on wavelet-domain foveation. Foveation filtering is the most straightforward method [8], [22], [68] wherein a bank of low-pass filters is applied to the image on a pointwise basis, with bandwidths monotonically decreasing with eccentricity. The filters used are usually symmetric, unit-volume 2-D Gaussians of the form [8], [22]

$$G_{\sigma}(x,y) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2+y^2}{2\sigma^2}} . \tag{1}$$

The other popular approach is to selectively subsample and quantize the image data in the wavelet domain, leading to decreased resolution away from the fovea. Such techniques have proven very effective for image and video compression [9], [22], [68]. We choose to use the simple and direct method of foveation filtering with Gaussian filters, owing to the simplicity of the method, and since the artifacts that naturally arise in accomplishing wavelet-domain quantization might lead to spurious corner responses. While wavelet-domain methods are certainly of high interest, in this first study we choose to adopt the direct approach, which yields foveated images which vary smoothly.

The cutoff frequencies of the Gaussian filters (1) can, in principle, be made to decrease continuously with eccentricity, to match the sampling grain of the image, it is also possible to more coarsely quantize the cutoff frequencies so that concentric rings of constant cutoff frequency are formed on the image surrounding the point of foveation. This simplifies practical implementation while effecting the foveated appearance of the image only slightly. The half magnitude cutoff frequency of the Gaussian (1) is

$$\omega_c = \frac{\sqrt{2 \cdot \ln(2)}}{\sigma} . \tag{2}$$

Hence one can implement foveation filtering by making σ an increasing function of eccentricity.

Several approximations serve to simplify implementation and to improve performance. The support of a Gaussian is infinite but a good approximation can be made by centering it in an array of about 3σ pixels square. Instead of continuously varying σ with eccentricity, the image is divided into a series of n bands, B_i that are concentric about the fixation point. The innermost ring is actually a circle with center the fixation point, while the outermost ring extends to the borders of the image.

We use a simple formula to determine the radius of each ring:

$$r_0 = 0,\ r_i = (i+2)^{1.6},\quad i = 3,4,\ \ldots n-1,\ r_n = \infty \tag{3}$$

where the radius of the i^{th} ring is r_i (pixels). The innermost ring has a radius of 5.8 pixels, and the distances between the rings increases with eccentricity. This formula provides a reasonable tradeoff between execution time and continuity at the ring boundaries. The image between the i^{th} and the $(i\text{-}1)^{st}$ ring is convolved with a Gaussian $G_{\sigma_i}(x,y)$, where σ_i increases monotonically with eccentricity.

Since convolving Gaussians with small values of σ takes less processing time than with larger values, efficiency is achieved by implementing larger Gaussian convolutions via repeated convolutions of smaller Gaussians. Repeated convolutions with Gaussians of spatial parameters designated $\sigma_i, i = 1,\ldots,k$ is equivalent to a single convolution with a Gaussian with spatial parameter

$$\sigma = \sqrt{\sum_{i=1}^{k} \sigma_i^2}\ . \tag{4}$$

The foveation filtered image is created by the following process. The input image is first convolved with a Gaussian of spatial parameter σ_1 and the results stored. This blurred image is next convolved with a Gaussian of spatial parameter σ_2 and the results stored. The process continues for the maximum possible number of bands. Later, when a fixation point is created, each band is filled from the appropriate stored image. Hence, the k^{th} filtered image is

$$I_k = I \mathop{\ast}_{i=1}^{k} G_{\sigma_i}(x,y), \tag{5}$$

where I is the original input image.

The values of σ_i used in the algorithm are set to approximate the spatial frequency response of the human vision system (HVS) based on the following widely used formula [22], [24], [68]:

$$CT(f,e) = CT_0 e^{\alpha f \frac{e+e_2}{e_2}} . \tag{6}$$

Here *CT(f,e)* is the contrast threshold expressed as a function of spatial frequency f (cycles/degree) and eccentricity e (degrees). CT_0 is the minimum contrast threshold, α, a spatial frequency decay constant, and e_2 is the eccentricity in degrees at which the contrast threshold drops to one half of maximum. The half-magnitude spatial cutoff frequency f_c can be expressed a function of eccentricity by solving:

$$-\log(CT_0) = \alpha\, f_c \frac{e+e_2}{e_2} \tag{7}$$

which yields:

$$f_c = \frac{-\log(CT_0)e_2}{\alpha\left(e+e_2\right)} \tag{8}$$

In arriving at this formula, Geisler and Perry [24] fit (6) to various sets of experimental data taken from the vision literature. They found good consistency with the following parameter selections: $\alpha = 0.106$, $e_2 = 2.3$, and $1/76 < CT_0 < 1/64$. Substituting these values into (8) and using an average for the high and low values for CT_0 yields a numeric relationship between cutoff spatial frequency and eccentricity:

$$f_c = \frac{92.024}{e+2.3} \quad . \tag{9}$$

The spread parameters of the Gaussians may then be found from (2).

2.2. Foveated Edge Detection

In our approach to corner search, edges recovered from the foveated images are used as features input to a corner detection apparatus. Edge detection is a subject that has been studied with considerable intensity for more than four decades. As such, there is a great variety of edge detection choices and considerable variance in edge detection philosophies. The most prominent categories of edge detectors are probably those which compute image derivatives, such the gradient, the laplacian, or directional derivatives of the image intensity, with appropriate smoothing either built-in or accomplished before implementation of discrete derivative approximations [69, Chapter 4, 13], and those which modify this process by using smoothing along preferred directions prior to differentiation, *viz.,* anisotropic filtering [69, Chapter 4.14]. There is no doubt that a great variety of edge detection operators may be applied to foveated data. In the approach given here, we will utilize the relatively simple and straightforward Canny edge detector for several reasons [21]. First, the Canny operator provides excellent localization in the edge detection results; second it is simple and naturally defined; third, it gives good performance where the edge curvature is high, and lastly, it does not require any kind of iterative processing, unlike anisotropic schemes. Given the framework of corner-finding via sequential fixations that we are presenting here, direct, locally-computed approaches appear to be a more natural choice, because of the need for rapid, localized processing.

We briefly describe the Canny operator in the context of foveated edge detection. Given an image $I(x, y)$, the usual method is to form the Gaussian smoothed image

$$S_{\sigma}(x,y) = G_{\sigma}(x,y) * I(x,y) \tag{10}$$

from which an estimate of the gradient ∇S_{σ} is computed. In our application, I is not convolved by a single Gaussian, but is instead smoothed by a space-variant Gaussian. In the Canny formulation, the unit vector in the gradient direction $\angle \nabla S_{\sigma}$ estimates the direction normal to the edge:

$$\mathbf{n}_{\sigma}(x,y) = \frac{\nabla S_{\sigma}(x,y)}{\left|\nabla S_{\sigma}(x,y)\right|}. \tag{11}$$

Putative edge locations are then marked by the zero crossings of the twice directional derivative in the direction of the normal (11):

$$\frac{\partial^2 S_\sigma(x,y)}{\partial \mathbf{n}_\sigma^2(x,y)} = \mathbf{n}(x,y)\cdot\nabla\left[\frac{\nabla S_\sigma(x,y)\cdot\nabla S_\sigma(x,y)}{\left|\mathbf{n}_\sigma(x,y)\right|^2}\right]. \tag{12}$$

It is easily shown [21] that the zero crossings of (12) are conveniently the same as those of

$$D(x,y) = \nabla S_\sigma(x,y)\cdot\nabla\left[\nabla S_\sigma(x,y)\cdot\nabla S_\sigma(x,y)\right]. \tag{13}$$

Discrete implementation of (13) is accomplished using space-varying discrete directional Gaussian derivatives $\partial G_{\sigma,x}(x,y)/\partial x$ and $\partial G_{\sigma,x}(x,y)/\partial y$ to compute the discrete gradient expressions $\nabla S_\sigma(i,j)$ at each discrete image coordinate (i, j).

The zero-crossing maps obtained by a space-varying edge detector may be viewed as an oriented slice through edge scale-space [70], [71] as the distance from the foveation point increases; it is possible that this outlook may provide valuable insights into foveated edge detection processes.

2.3. Detection of Corners

Many researchers have studied corner detection, although there has not been any prior work that we have been able to find involving corner detection on foveated data. However, corners are usually regarded as points of high curvature, or of curvature discontinuity, along the contours of detected boundaries, edges, or local image intensity profiles. Of course, different definitions of curvature exist. A common and effective definition is to take the curvature κ as the derivative of tangent angle, with respect to arc length, of a parametric curve $x = x(t),\ y = y(t)$ [17]:

$$\kappa \equiv \frac{d\phi}{ds} = \frac{\frac{d\phi}{dt}}{\sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2}} \tag{14}$$

where s is arc length, ϕ is the tangent angle, and where

$$\phi = \arctan\left(\frac{\frac{dy}{dt}}{\frac{dx}{dt}}\right). \tag{15}$$

Shortening the notation and taking the derivative gives:

$$\frac{d\phi}{dt} = \frac{x^{'}y^{''} - x^{''}y^{'}}{1 + \left(\frac{y^{'}}{x^{'}}\right)^2}, \tag{16}$$

which substituted into (14) yields:

$$\kappa = \frac{x^{'}y^{''} - y^{'}x^{''}}{\left(x^{'2} + y^{'2}\right)^{3/2}}. \tag{17}$$

It can be easily shown from the definition that κ equals the reciprocal of the radius of curvature. The curvature measure (17) on a digitized curve is highly sensitive to noise because of the computed derivatives, so commonly the curve is smoothed, e.g., a low order polynomial is fit to the curve in a sliding window, and the derivatives of the polynomial are used in (17) to calculate a value for curvature at the center of the window. A local maximum of $|\kappa|$ may be taken to indicate the presence of a sharp bend in the curve or a corner.

Mehrota *et al.* [4] developed corner finders based on directional first and second derivatives of Gaussians, which can detect half-edges at any desired angle to each other.

Flynn and Jain [5] describe a series of corner detectors based on a variety of curve fitting methods. They also mention the necessity for smoothing the curves.

The Moravec interest operator is based on the response of a small averaging window to an image. If the window straddles an edge, then moving it parallel to the edge direction creates a small change in response whereas moving it normal to the edge creates a large response. If the window straddles a corner, however, moving it in any direction will cause a large response. The Moravec [18] detector declares a corner if the minimum change produced by a shift exceeds some threshold.

The Plessey corner finder [18], [1] is based on a matrix M of products and squares of directional image derivatives. At points where two eigenvalues of M are large, small shifts of the window position in any direction will cause a large change in its average response, indicating that the point may be a corner.

The SUSAN corner [20] detector applies a moving circular template to an image and declares a corner at points where the value at the center of the template is approximately equal to a small portion of the entire template.

Mokhtarian and Suomela [19] developed a variable scale corner detector based on the curvature formula (14) and the Canny edge detector. They initially convolve the image with a wide Gaussian, smoothing corners into broad curves. Locating the position of maximum curvature gives an estimate of the corner position, which they refine by narrowing the scale of the Gaussian, and by tracking the corner as it moves.

While there has not been any definitive study conducted which would indicate which corner detection algorithm is to be preferred – unlike, e.g., edge detection theory, where a variety of optimal criteria have led to so-called optimal edge detectors – we use the formula (14) for a variety of reasons: it uses a very natural definition, it is a localized computation, it is widely used, and although derivative-based, the use of smoothing in our approach to creating foveated images reduces the sensitivity of the operator to noise.

2.4 Sequential Computation of Fixation Points

A fixation selection measure $m_{i,j}$ is then computed over the entire edge map (as explained below), and the next fixation is placed at the pixel with the highest value of this measure. A new foveated edge map is created based on the new fixation position and the search algorithm is invoked to produce a short saccade, using a different calculation for the fixation selection measure than for a long saccade. A new foveated edge map is created and another short saccade generated. Short saccades are generated until a corner is deemed found, or until a corner is not found, which is assumed when short saccades are continually generated. If seven short saccades are generated in succession, or if a corner strength measure is sufficiently large to positively identify a corner, then the search is deemed to have failed and a long saccade is generated, which moves the fovea to a different region of the image.

We now describe the fixation selection algorithm in detail. At each fixation a foveated edge map is computed as described in Chapter 2. A curvature map is computed along the edge (zero-crossing) loci. In order to reduce the effects of noise on the derivative computations, a simple third-order polynomial is locally fit at each point on the zero-crossing contour. The curvature (17) is then computed at each point (i, j) that lies on the smoothed zero-crossing contours. The curvature strength $\kappa_{i,j}$ is one of the multiplier factors in the fixation selection measure $m_{i,j}$.

Figure 1 illustrates the calculation of a curvature map: Figure 1(*a*) depicts a contour with two points indicated: A and B. Figure 1(*b*) depicts a close-up of point (A) -

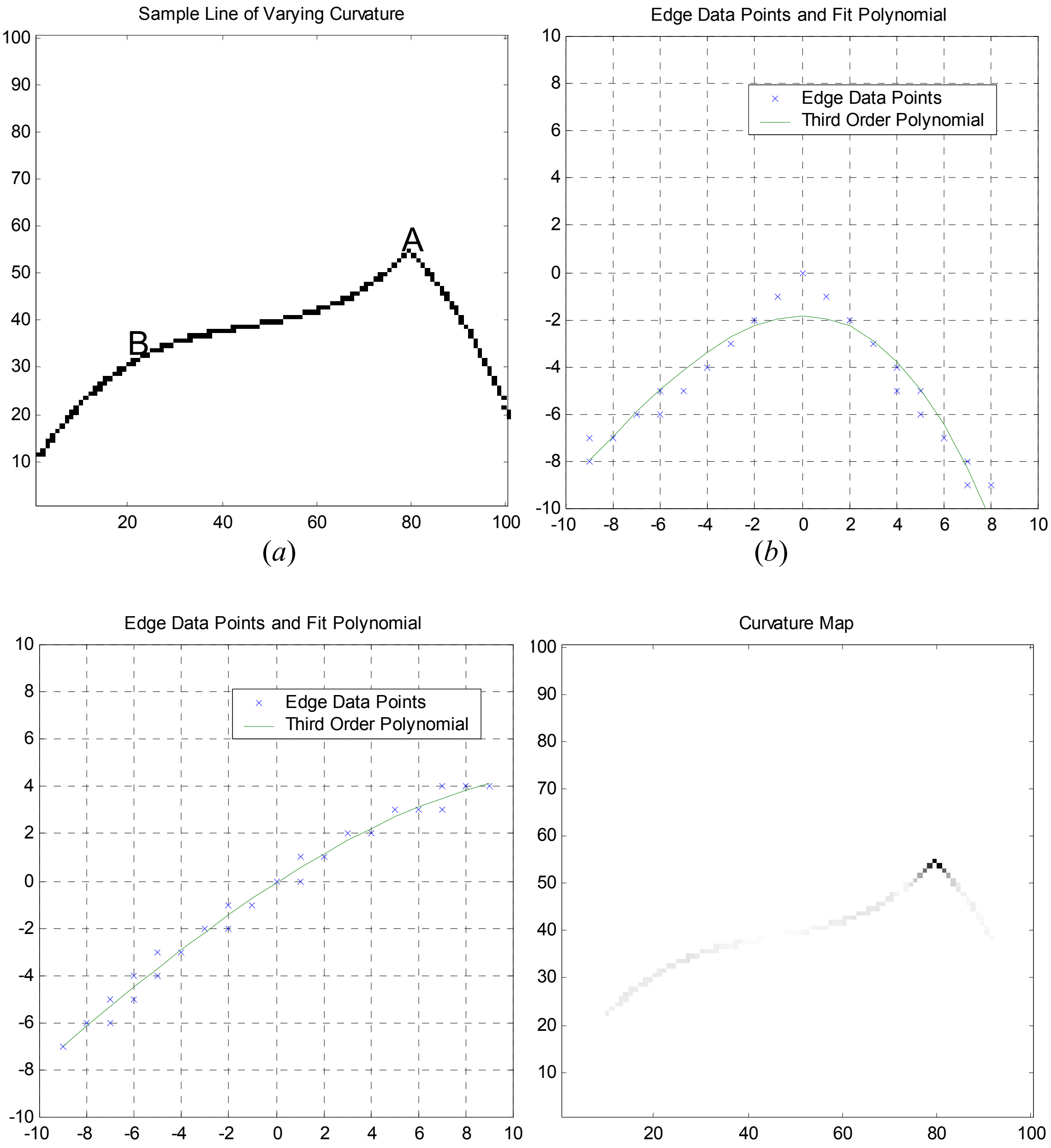
a high curvature point – along with its local polynomial fit, while Figure 1(c) shows the same for the low curvature point B. Finally, Figure 1(*d*) shows the original contour in (a) with curvature coded by the intensity of the line (darker = higher curvature).

We believe that curvature alone is not a suitable measure for placement of subsequent fixation points for two reasons. First, even if the fixation selection algorithm were probabilistically-driven, very high-curvature locations would be visited repeatedly. Our goal is to successfully search for as many corners that are in the image as possible. Secondly, noise or low-contrast curves may create zero-crossing loci having high curvatures, thus attracting the fovea to uninteresting regions or artifacts in the image.

To address the first of these problems, an array of history information is maintained and used to define a second multiplier factor in the fixation selection measure $m_{i,j}$. Let

$$h_{i,j} = \begin{cases} 1 \;; |i - i_{fk}| \le 12, |j - j_{fk}| \le 12 \\ 0 \;; \text{otherwise} \end{cases} \tag{18}$$

where the coordinates of the k_{th} fixation point are denoted by i_{fk} and j_{fk}. Whenever a fixation point is generated, a 25x25 unit square centered at the fixation is added to the history array. In this way, long saccades are prevented from landing too close to previously-visited locations.



(*a*) (*b*)

(*c*) (*d*)

Figure 1. Generation of curvature map on smoothed zero-crossing contour. (*a*) Zero-crossing contour; (*b*) Local polynomial fit near the high-curvature point A. The axis coordinates are relative to A. (*c*) Local polynomial fit near the low-curvature point B. The axis coordinates are relative to B. (*d*) Zero-crossing contour in (a) with curvature coded as intensity (darker = higher curvature).

Two observations motivate the next term used in the fixation selection measure. First, corners that lie further from the current fixation point (i_f, j_f) will be more severely blurred by foveation, so the apparent curvature of distant corners will be reduced. Secondly, once a corner is found at (i_f, j_f), a large saccade is desired to cause the algorithm to scan the image more quickly. Hence the distance factor

$$d_{i,j} = \sqrt{(i - i_f)^2 + (j - j_f)^2} \,. \tag{19}$$

is used as a multiplier in the fixation selection measure $m_{i,j}$. This term compensates for the fact that corners away from the foveation point turn into broad curves by giving extra weight to curves far from the fovea. In addition, it forces the fixation point to move large distances between fixations, forcing it to scan the entire image more quickly.

We have also chosen to include an edge strength factor in the fixation selection measure. Our viewpoint is that corners having large edge magnitudes are more likely to be associated with significant image structure. The edge strength factor is simply the squared gradient magnitude of the Gaussian-smoothed space-variant image.

$$s_{i,j} = \left|\nabla S_\sigma(i,j)\right|^2 \tag{20}$$

The use of this term introduces an additional problem. A high contrast edge of low curvature may attract the fovea to an uninteresting region of the image. To eliminate edges of low curvature, we apply a threshold τ to the curvature data:

$$\kappa_{i,j} = 0, if\ \kappa_{i,j} < \tau \tag{21}$$

However, since foveation greatly reduces the apparent curvature of corners (an effect that increases with distance), it is possible that no computed curvature may exceed τ. In such instances, τ is temporarily set to zero until a new long saccade is generated.

The overall fixation selection measure is

$$m_{i,j} = \begin{cases} \frac{\kappa_{i,j}}{1+\kappa_{i,j}} \cdot \frac{s_{i,j}^2}{\max(s_{i,j}) + s_{i,j}} \cdot \frac{d_{i,j}}{\max(d_{i,j}) + d_{i,j}} \cdot \frac{1}{1+5h_{i,j}^2}; C = 0 \\ \frac{\kappa_{i,j}}{1+\kappa_{i,j}} \cdot \frac{s_{i,j}}{1+50s_{i,j}} \cdot \frac{d_{i,j}}{1+d_{i,j}} \qquad ; C > 0 \end{cases} \tag{22}$$

where the maxima are taken over the entire image, and where C controls the length of the saccade. When $C = 0$, a long saccade is to be generated, and when $C > 0$ a short saccade is to be generated according to the formula in (22). C is initially given a value of zero, and is incremented by one with each saccade, until it is reset to zero. There are two conditions under which C is reset to zero: (*i*) The measured curvature $\kappa(i_f, j_f)$ at the current fixation point exceeds a threshold (0.9 in our algorithm), indicating the presence of a corner. (*ii*) Seven short saccades have been generated: $C > 7$. The value of 7 is arbitrary and is normally never reached. Its purpose is to force a long jump should the fixation point ever reach an empty part of the image where short saccades are unable to remove it.

Note that long saccades ($C = 0$) are discouraged from approaching previous saccades owing to the inclusion of the history term (18) in (22), but this is excluded for short saccades ($C > 0$) which attempt to zero in on strong local corners.

The global maximum of $m_{i,j}$ provides the coordinates for the next fixation point. When the next fixation is made, the saccade length control variable C is incremented from 0. After reaching 7 (following 6 subsequent short saccades) it is reset to 0 forcing another long jump. This produces a sequence of one long saccade, intended to explore a new region of the image, followed by several short ones (fewer than 7), which pinpoint the corner accurately.

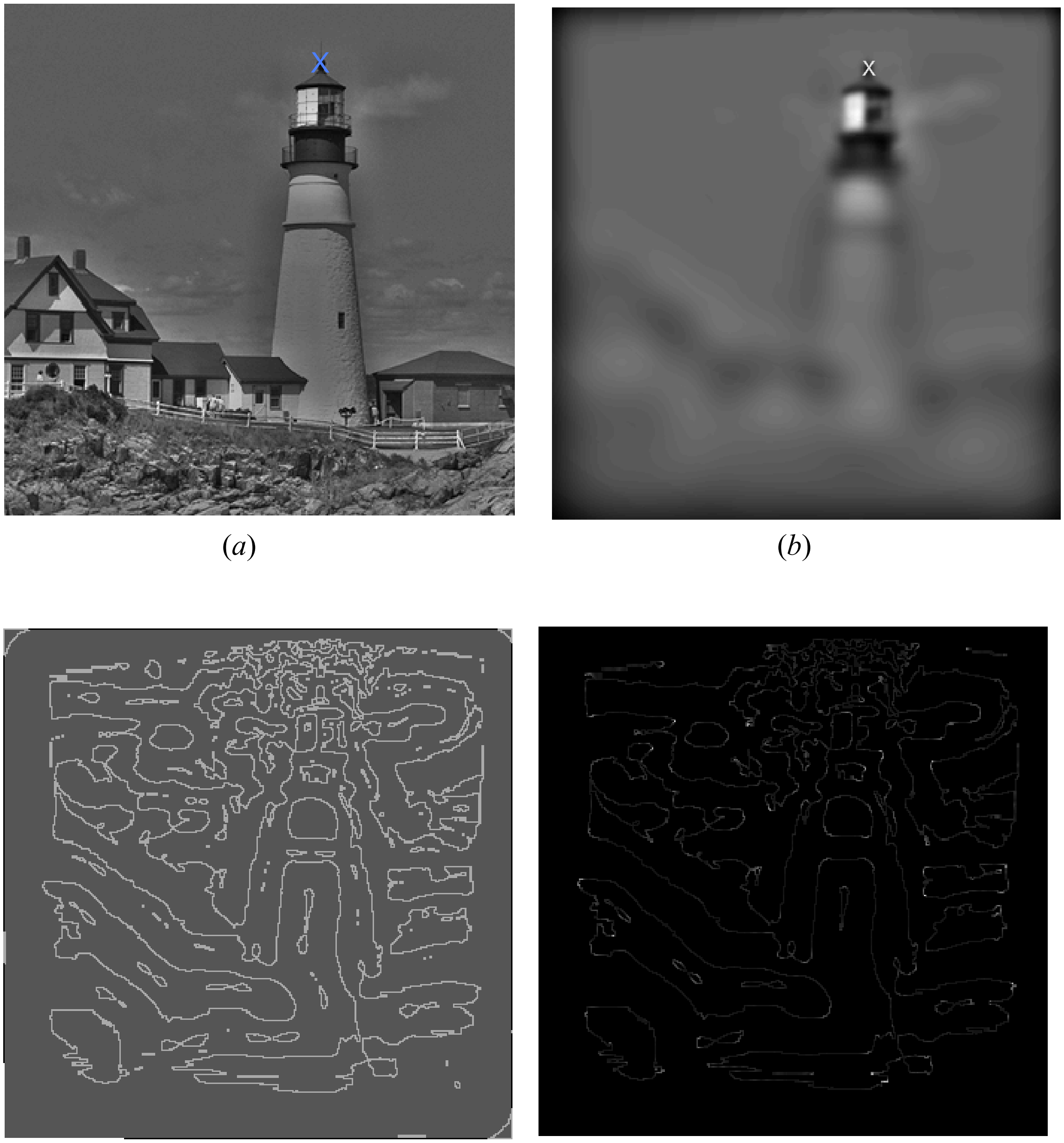
Finally, the algorithm may be terminated in a number of ways, depending on the application. It may be terminated after a fixed number of fixations, or after the fixation selection measure $m_{i,j}$ fails to exceed a predetermined threshold over several attempts, indicating that the pool of available and unvisited corners in the image is exhausted.

We illustrate the steps of the algorithm by example. Figure 2(*a*) is the image *lighthouse*. In each image, the fixation point is designated by the symbol "X." In this example, the current fixation point is presumed to be at the peak of the lighthouse, as indicated. Figure 2(*b*) shows a foveated version of *lighthouse* – although, of course, this image is not calculated by the algorithm, since (13) in discrete form is used to generate the zero crossings. Figure 2(*c*) depicts the foveated edge map calculated by the foveated Canny edge detector, and Figure 2(*d*) is the foveated curvature map, with intensity made proportional to curvature.

2.5 Assessing the Fixation Points

It is desirable to be able to assess the efficacy of any image feature extraction mechanism, since accurate extraction is necessary to the success of most image analysis or classification algorithms. However, testing the effectiveness of corner-finding algorithms is difficult, for reasons similar to those which limit methods for testing edge detectors. Corners, like edges, lack a precise definition; they manifest innumerable variations in attributes such as magnitude, sharpness, scale, duration, and so on. Indeed, detected corners are more difficult to assess since they are usually computed from already vaguely-defined edges. For edges and corners, there is no existing effective ground truth in natural images.

Nevertheless, we have attempted to validate our method through comparisons to corner maps computed by humans in two different ways.



(*a*) (*b*)

(*c*) (*d*)

Figure 2. Example of calculation of subsequent fixations. (*a*) Original image lighthouse. (*b*) Foveated version by space-varying Gaussian filtering. (*c*) Foveated edge map by foveated Canny edge detection. (*d*) Foveated curvature map.

In the first method, we compare the corners detected by our algorithm to *handpicked* corners chosen by a human. To eliminate any questions of bias on the part of the human corner-finder, we have applied this method only to an image of a geometric object with corners that are evident. This is useful since it benchmarks our algorithm on an image with an effective ground truth.

In the second method, we compare the algorithm's results against the visual fixations, measured by a precision eyetracker, of human subjects viewing a naturalistic image. The subjects were asked to accomplish a simple task: to search for corners in the image. Each subject was briefed beforehand to give them idea of what was meant by a corner (without referring to any image used here). This experiment has the virtue of supplying a ground truth of sorts of images of the real world. However, the results are naturally limited by the fact that human visual fixations are guided by many low-level and high-level mechanisms, even in subjects instructed to perform a specific visual task.

2.6 Method of Comparison

Before explaining the procedures for obtaining comparison data, we explain the method used to make the comparisons. The method used needed to satisfy several criteria: (*a*) computed and handpicked corners and fixations are defined on sparse sets of singleton points in the image plane; (*b*) exact hits between detected corners and either fixations or handpicked points are likely to be relatively rare. Owing to these limitations we opted to use a method of comparing sparse sets of visual fixations similar to one used in [26].

The first step is to create a dense fixation-point (or handpicked point, or corner point) image by a process of interpolation. Begin with a zero array with domain the same as the image. Then, at the coordinates of each fixation point (for eyetracked results), handpicked point, or algorithm-computed corner, compute an isotropic 2-D Gaussian with space constant chosen such that the half-peak width is equal to the diameter of the foveola – or about 1° of visual angle – for the eyetracked observers. Each Gaussian has unit peak value. The same space constant is used for the Gaussians that interpolate the handpicked and computed results. In each dense fixation/corner image, the Gaussians are summed to create an overall "fixation" map. As each Gaussian is generated, it is integrated into the current dense fixation map using the summed weighting $1-(1-p)(1-q)$, where at any coordinate p is the value of the existing map and q is the value of the Gaussian centered at the new fixation. When the map is completed, it is normalized to have unit volume (unit array sum). This makes it possible to interpret the dense fixation images as probability maps (2-D empirical mass functions) of fixation placement associated with each image.

This process is illustrated in Figure 3, which shows the Gaussians computed from a set of sample fixation points (marked as 'X'), with overlapping envelopes summed. Replacing each fixation point by a 2D Gaussian is a simple method for approximating the probability that a neighborhood region around the fixation could have been selected as a fixation point. Using a Gaussian to interpolate each fixation point allows for uncertainty in its location which can arise from small errors in the calibration, and allows for imperfect accuracy of the eye movement measurements.

To compare the probability maps from visual fixations, from handpicked corners, and from computed corners, we use a standard information-theoretic measure of the similarity between probability density functions: a modified Kullback-Leibler distance (KLD). The KLD measure the relative entropy between two probability functions

$$\mathrm{D}(p \parallel q) = \sum_{x \in \mathrm{X}} p(x) \log\left(\frac{p(x)}{q(x)}\right). \tag{23}$$

The KLD is not a true distance function, since it is not symmetric and does not obey the triangle inequality. However, it is convex and $\mathrm{D}(p \parallel q) = 0$ if and only if $p(x) = q(x)$ [27]. Since there is no reason to prefer an asymmetry, we use the symmetric distance [28]:

$$\mathrm{DS}(p \parallel q) = \frac{1}{\frac{1}{\mathrm{D}(p \parallel q)} + \frac{1}{\mathrm{D}(q \parallel p)}} \tag{24}$$

to quantify the distance between (interpolated) corner locations computed using our proposed algorithm with either (interpolated) handpicked corners, or with (interpolated) recorded eye fixations (measured from observers looking for corners).

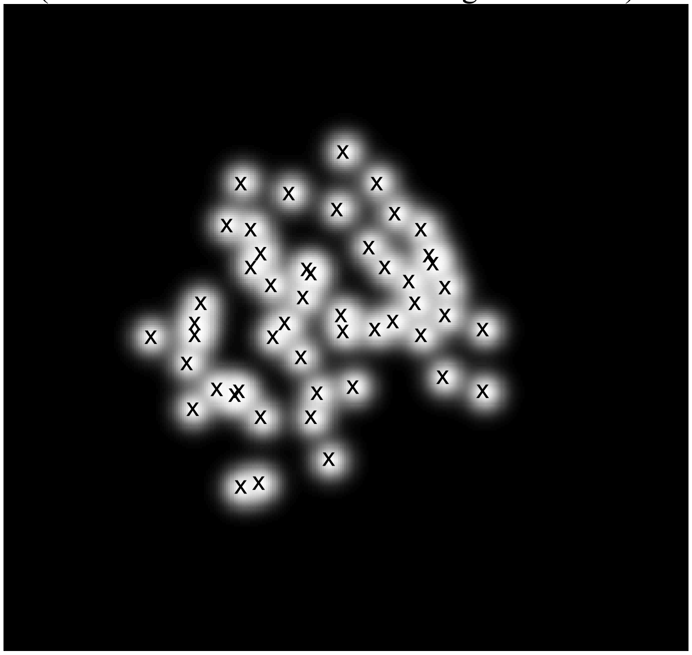


Figure 3. Sample fixation points with Gaussian interpolation.

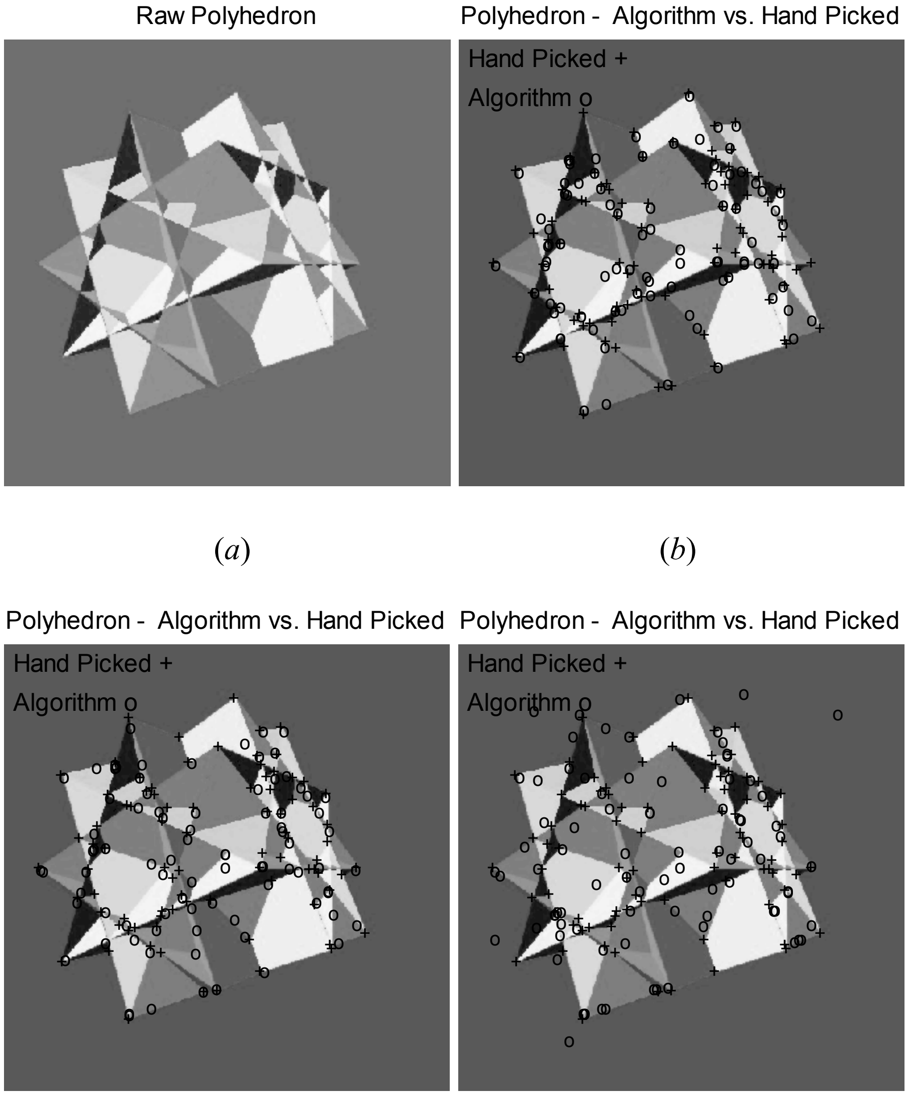
2.7. Comparison with Handpicked Corners

Our first comparison is made with an image of an object with reasonably well-defined corners. The algorithm was run on this and the resulting computed corners compared with those obtained by human handpicking of corners. Figure 4 contains images of a polyhedron. We counted 85 vertices in this image and handpicked the coordinates of each using a graphics program with a crosshair cursor. While many of the vertices result in unambiguous corners, other vertices present less obvious corners owing to their geometric placement, the shading of the object, and so on.

We ran the algorithm until it computed 85 fixations on the polyhedron, for three different values of τ (see (21)), and calculated the KLD between the algorithmic and handpicked vertices. Results are shown in Figs. 4(*b*)-(*d*).

2.8. Comparison With Eyetracked Fixations

In addition to testing algorithm generated fixations with handpicked ones, we have compared the algorithm generated ones with those of four human subjects viewing each test image through an eyetracker. The experimental protocol is described in the Appendix.



(*a*) (*b*)

(*c*) (*d*)

Figure 4. Polyhedron image used to compare the corner detection algorithm with handpicked corners for 85 fixations. (a) raw image(*b*) For $\tau = 0.1$, the KLD = .1323. (*c*) For $\tau = 0.2$, the KLD = .1490. (*d*) For $\tau = 0.3$, the KLD = .3702.

Here we test the algorithm on the two images shown above: the lighthouse on a seashore shown above in Figure 2(*a*) and the polyhedron shown in 4(*a*). Table I provides

KLD values of algorithm vs. eyetracker generated fixations for each of the seven subjects. It also has a KLD value for polyhedron handpicked versus algorithm generated fixations. Values were calculated using the symmetrical KLD shown in (24). Each subject ran about 50 fixations on the eyetracker, so the algorithm was run for 50 fixations, as shown in Table I. The threshold τ was set to 0.1 since that yielded the lowest measured value of the KLD for the case of the polyhedron handpicked versus algorithm.

Table II shows algorithm versus handpicked for 50 and 85 algorithmic fixations. The lowest measured distance is between the polyhedron handpicked and algorithmic, demonstrating that the algorithm performs its intended task of finding corners well. Since there were 85 handpicked vertices, it was to be expected that the run with 85 algorithmic fixations would give a lower distance than the one with 50 fixations, which was the case. The polyhedron handpicked versus eyetracker comparison is consistently worse than the handpicked versus algorithm. The eyetracker versus algorithm is worse still. From this one might deduce that corners are poor predictors of visual fixations, yet, in a complex scene such as this, corners are one of many different classes of features that attract fixations.

To test the accuracy of the algorithm for finding corners, we calculated (for the polyhedron image) the distance from each algorithmic fixation to the nearest handpicked one, and repeated the process for all eyetracker fixations. If the minimum distance for each fixation point was less than or equal to a given value, a “match” was declared. Figure 5 shows a comparison of matches as tolerance varies from zero to one degree. At zero tolerance, neither method shows matches. As tolerance increases, the handpicked matches increase much faster than the eyetracker ones. This further demonstrates that the algorithm locates corners far more accurately than human subjects.

Table I. KLD values for eyetracker vs. algorithm and eyetracker vs. handpicked 50 fixations.

Comparison			Subject	
	HHC	IVDL	UR	YL
Polyhedron - Eyetracker vs. Hand Picked	0.411	0.603	0.876	0.398
Polyhedron - Eyetracker vs. Algorithm	0.317	0.492	1.065	0.408
Lighthouse - Eyetracker vs. Algorithm	2.743	2.216	2.505	1.927

Table II. KLD values for algorithm vs. handpicked for 50 and 85 fixations.

Comparison	
Polyhedron - Random vs Hand Picked 85 Fixations	2.033
Polyhedron - Algorithm vs. Hand Picked 50 Fixations	0.269
Polyhedron - Algorithm vs. Hand Picked 85 Fixations	0.132

Figures 6 and 7 show algorithmic versus eyetracker fixations for four subjects apiece, for the lighthouse and polyhedron. The polyhedron images include the handpicked vertices. In addition, Figure 8 shows algorithmic fixations on an image of tools from the Rutgers Tool Database [78] and on a natural scene from the van Hattern database of naturalistic images [83]. Figures 9 and 10 show runs of the algorithm on eight additional images.

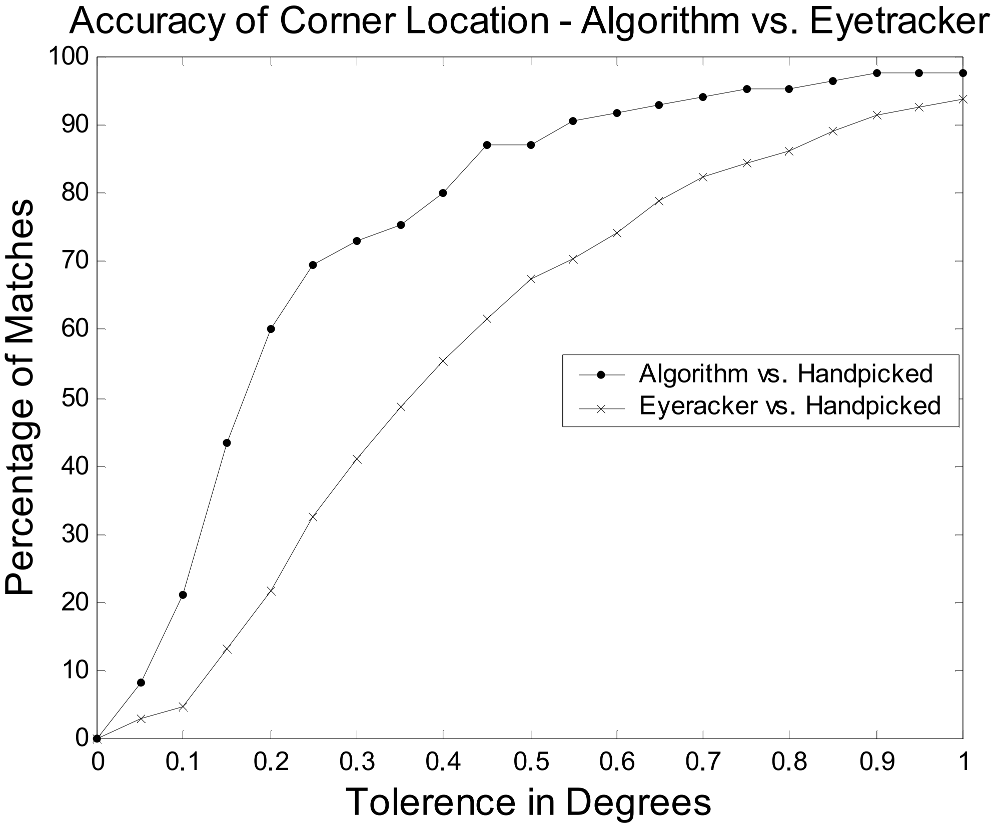
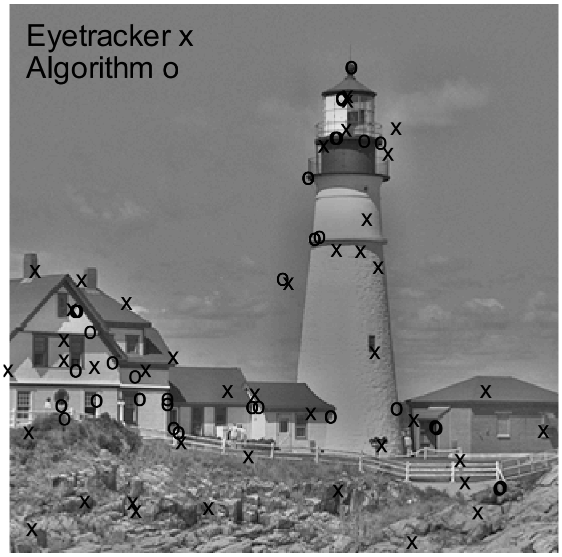
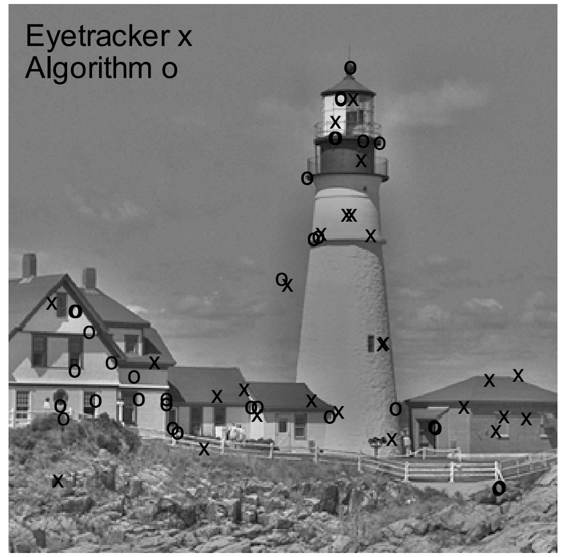


Figure 5. A comparison of the accuracy of corner locations, algorithm vs. eyetracker 50 fixations.



YL - Lighthouse



UR - Lighthouse

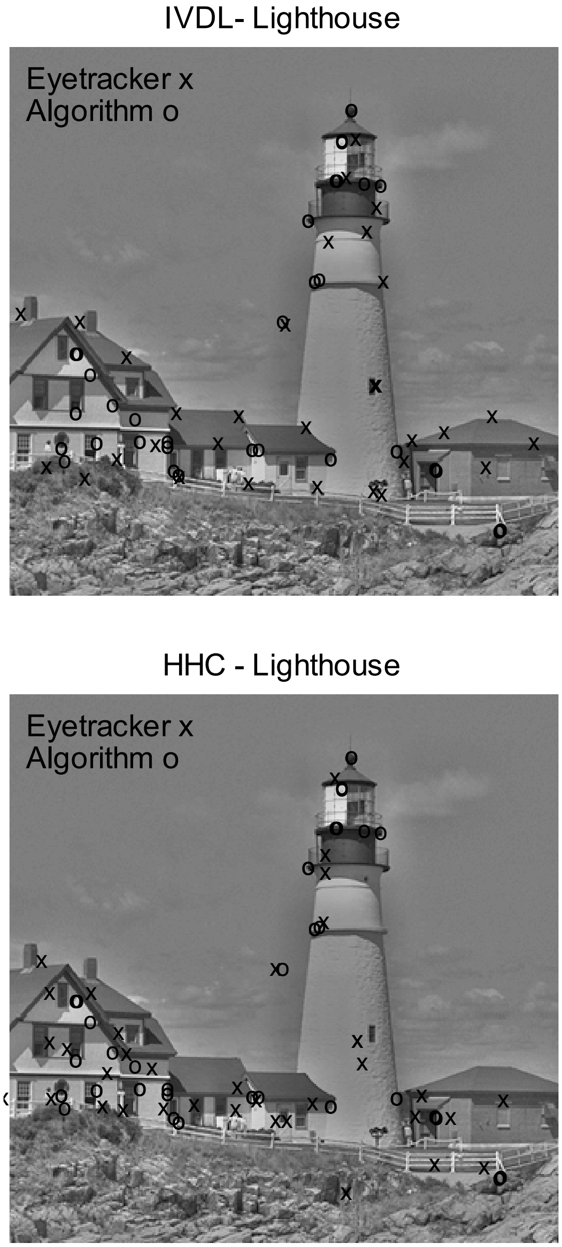
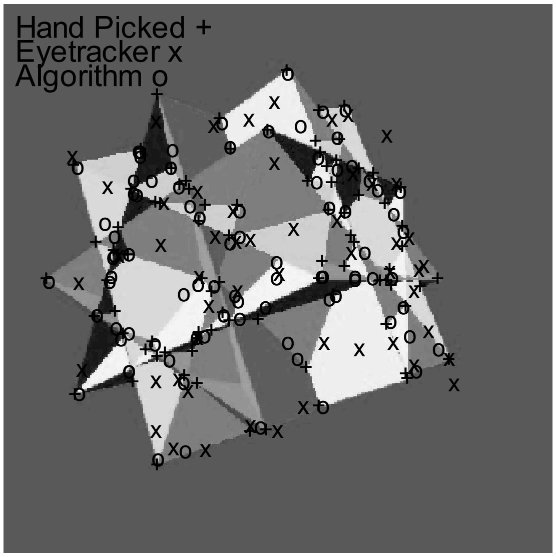
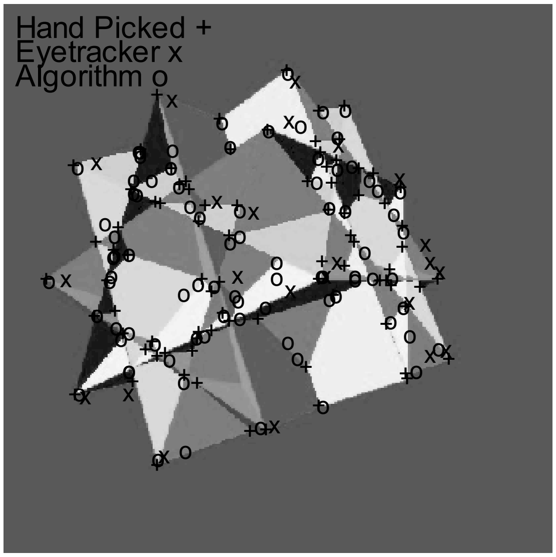


Figure 6. Four observers vs. 50 algorithmic fixations – lighthouse.

YL - Polygon   Hand Picked



UR - Polygon   Hand Picked



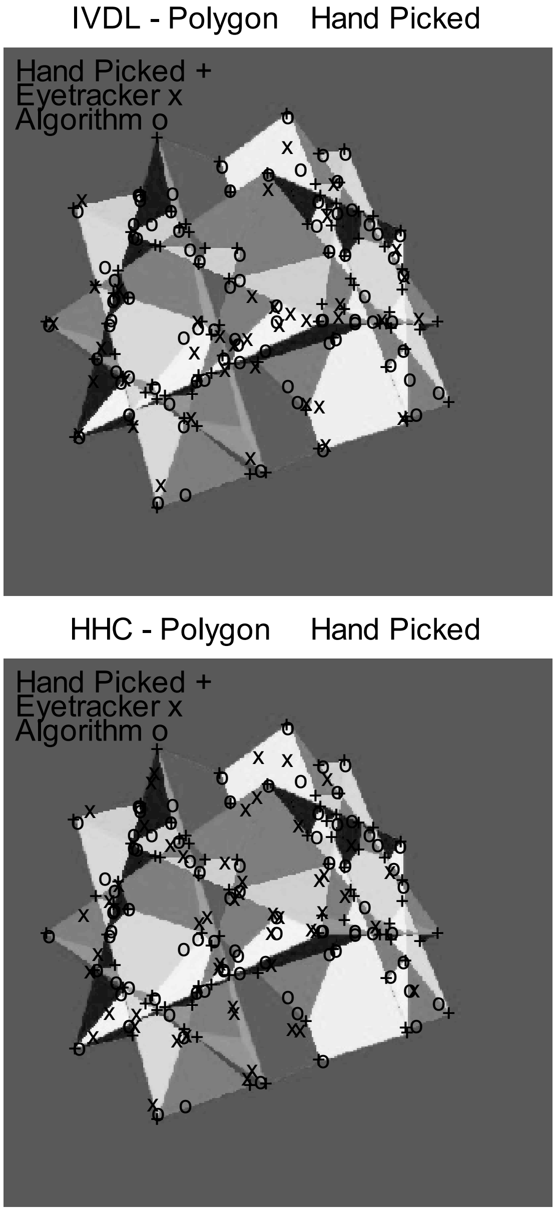


Figure 7. Four observers vs. 50 algorithmic fixations – polyhedron.

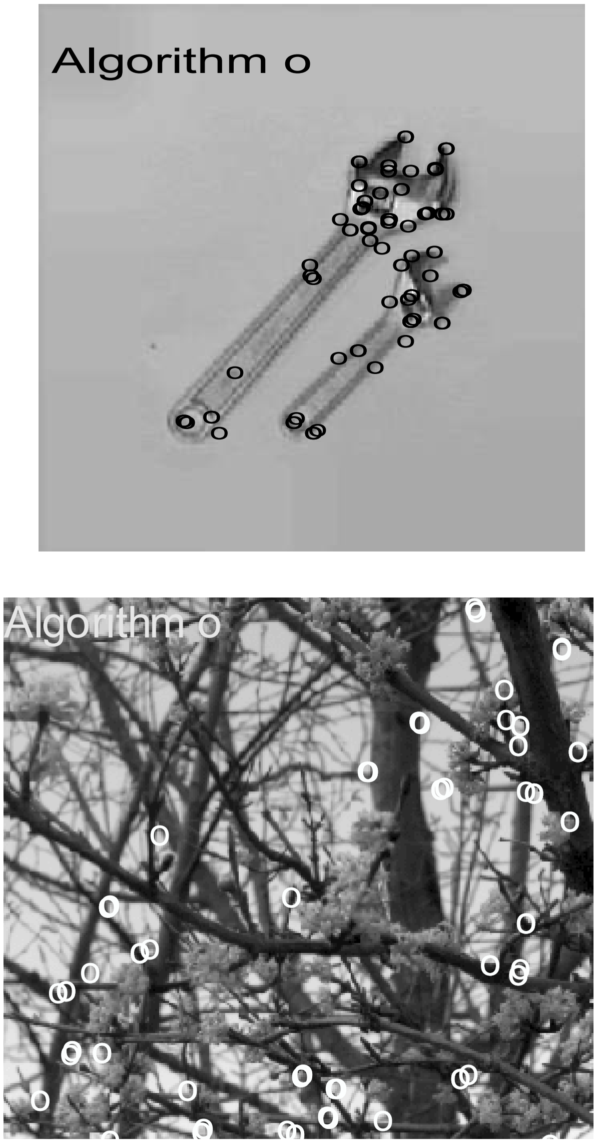


Figure 8. Test of algorithm on tool image (top) and natural scene (bottom).

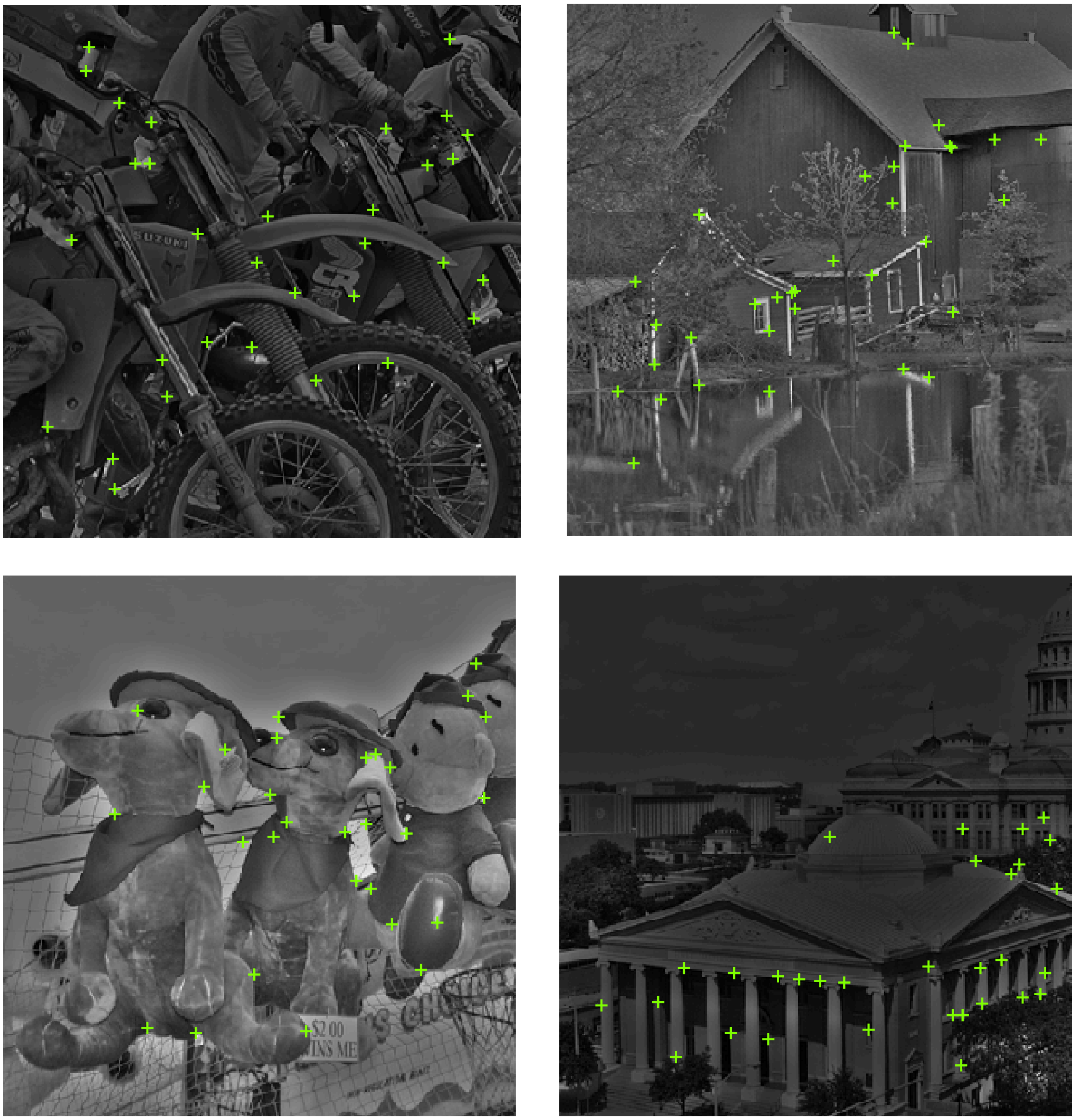


Figure 9. Test of algorithm on other images - part A.

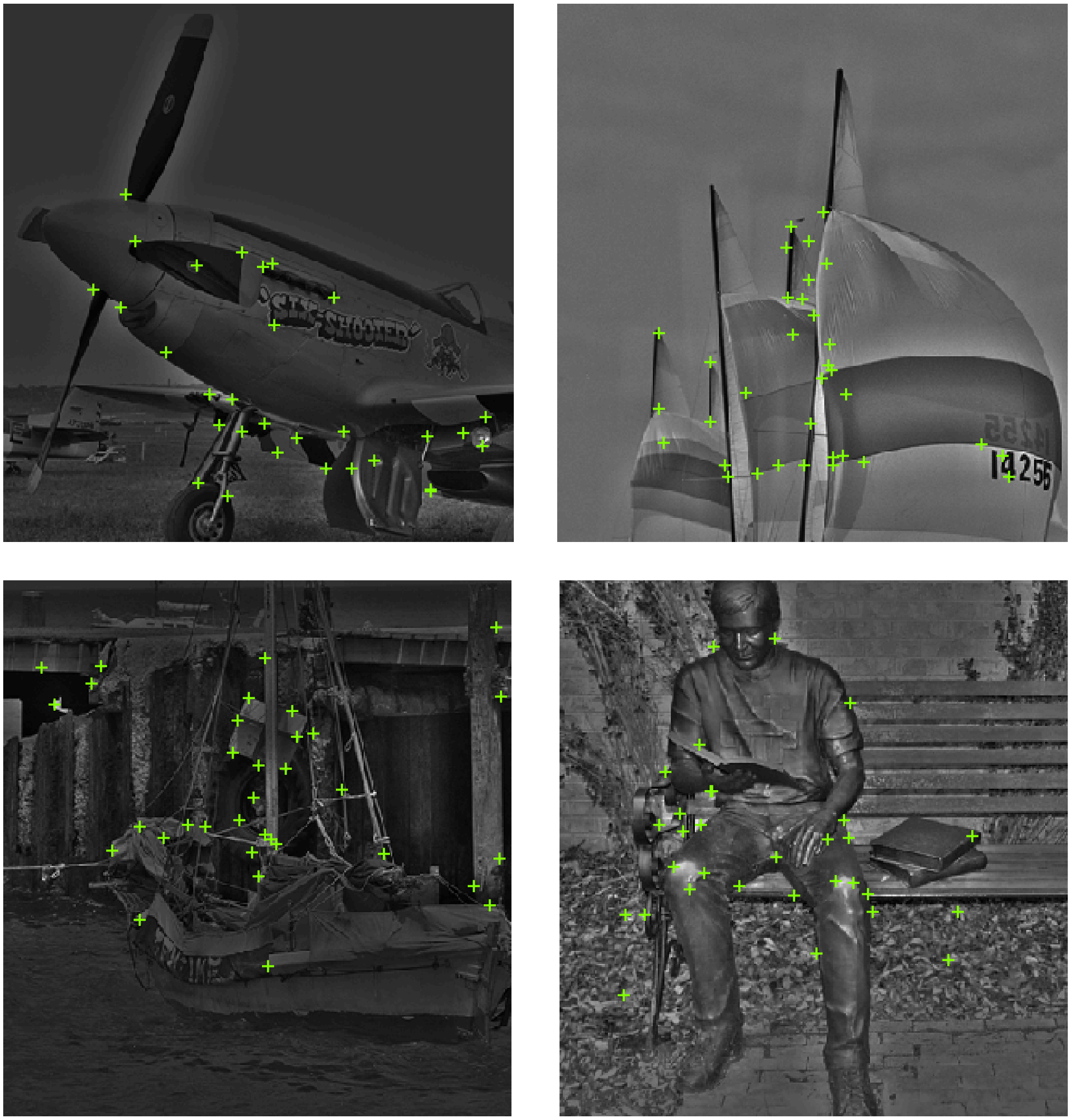


Figure 10. Test of algorithm on other images.

CHAPTER 3.

TWO-DIMENSIONAL FOVEATED OBJECT RECOGNITION

Two broad approaches toward object recognition are described in this section. The first approach deals with objects that can be easily segmented via binarization or contour detection. The second set deals with object recognition in natural images.

Sclaroff [97] and Sclaroff and Pentland [111] perform recognition of tool, animal, and fish images by means of models of object deformation models. They calculate the energy to deform an unknown image to a model in the database of known objects. Objects requiring less deformation energy are better fits.

Chung & Wang [88] used Hu moments for recognition of binary images of hand tools. They calculate a feature vector of seven moments for each object to be recognized, and then compare the results of a backpropagation artificial neural network (ANN) with two statistical classifiers, single nearest-neighbor and minimum mean distance. The ANN gives somewhat better results then the other two classifiers.

Tsang & Au [89] used a genetic algorithm for projective invariant object recognition of hand tools. First, they take a picture of the unknown object, then extract a contour. Finally, using a genetic algorithm, they compare the contour of the unknown object with a data set of contours of known objects.

Cai and Liu [90] combined HMM with Fourier spectral features for recognition of hand tools and handwritten numerals. They also extract contours and sample them at equidistant points along each contour.

Sossa and Palomino [35] perform recognition based on polynomial approximations of contours of binary images of geometrical objects. They first locate corners of 2D objects and then connect the corners with straight line segments. Next, they extract geometric invariant features from *metasegments*, groups of consecutive line segments and use these features to compare with a database of known objects. This system provides independence of camera viewpoints.

Stein and Medioni [91] use a system somewhat similar to [35] to recognize animal and airplane shapes. They also use polygonal approximations to contours and group adjacent segments, calling these groups *super segments*. They then quantize features of the *super segments* into a Gray Code to improve robustness. Matching is done with a hash table.

Fred, Marques, and Jorge [92] compare hidden Markov models (HMM) against stochastic finite state grammars for 2D hand tool recognition. They first extract the contour of the object then sample it at equidistant points, and convert it to a differential chain code, which they then model with an HMM.

A great deal of effort has been applied to the problem of recognizing objects in two-dimensional gray-level images, where it is assumed that the object being searched for has a known pose and hence projection (*viz.,* this is not 3-D search). In our approach, we do not make any assumption that the object to be identified can be segmented or the image binarized to reveal the object easily, and no segmentation process is used. Instead, features are extracted from the gray-level images and used for object recognition purposes on naturalistic images. Other approaches have operated under these more general assumptions. For example, SIFT is a general-purpose recognition philosophy invariant under rotation, translation, and scale that in [80], utilizes histograms of gradients around points of interest called *keypoints*. The essential philosophy of SIFT is used in our own work described below, but operating in a foveated framework and using specific localized features.

In [93], an object class recognition system is described that operates by decomposing an object into its component parts. SIFT is then used to locate keypoints and feature vectors based on location, scale, and “appearance.” A maximum likelihood method is then used to determine whether each vector belongs to a component, and hence a particular object. In [105], a system based on pyramid matching is described for recognizing categories in natural scenes. This technique hierarchically partitions images into sub-regions and calculates SIFT and other features.

A complex recognition system is described in [94], and extended in [95], based on models of visual cortex. A network consisting of four alternating layers of simple and complex cells implemented using the local maxima of Gabor filters and radial basis functions fed to a linear classifier. In [96], a recognition system based on saliency (measured as high local entropy), scale, and image content is described using a Bayesian framework for recognition. Another Bayesian approach to category recognition is described in [2], using the saliency features developed in [96]. Their system requires only a few images to learn a category of complex objects, with perhaps hundreds of parameters. Their method uses probabilistic knowledge accumulated from previously accumulated categories, even if unrelated.

In all of the preceding approaches described, and in fact all that we know of, images containing the objects to be recognized are processed in a uniform manner: the features that are used are computed from the full resolution image at every location, and once collected, are then used to attempt recognition. By contrast, in what we describe the features are detected in a foveated manner, and the process of recognition proceeds in a sequential manner as the features are computed. In this sense, the overall object recognition system may be viewed as foveated.

3.1 Foveated Edge and Corner Features

There is no consensus opinion yet formed regarding which low-level image features can be used to best advantage for generic object recognition applications. Depending on the application, low-level primitives such as edges, lines, corners, textures (and so on) have been proposed. An argument can be made for any of these; in our demonstration of the possibility of conducting foveated visual search, we choose to use gray-level corners. Corners as bearers of visual information are described above in the introduction and in [87]. Above in Chapter 2 and in [87] we described a foveated corner search algorithm that sequentially seeks corners using a foveated search protocol, and which is the basic feature selection mechanism used here. The approach is very briefly recapped here and changes to the algorithm for object recognition duly noted.

In the simple approach to foveated object recognition described here, the following steps are therefore taken: a sequence of points of foveation ("fixations") are serially computed as described in the next section 3.2. At each fixation a foveated edge map using the new concept of foveated Canny edge detector is computed as described in the preceding. A curvature map is computed along the zero-crossing ZC loci. Stable results are obtained by fitting a third-order polynomial at each point on the ZC contour. The curvature $\kappa_{i,j}$ (17) is then computed at each coordinate (*i, j*) on the smoothed zero-crossing contours. This general procedure is described in more detail above.

The curvature strength $\kappa_{i,j}$ at each (*i, j*) is one factor used in three ways: First, whether a corner has been found at the current fixation coordinates (i_f, j_f), which occurs when

$$\kappa_{\left(i_f, j_f\right)} > 0.9. \tag{25}$$

Secondly, the curvature strength $\kappa_{i,j}$ is used to determine the next fixation point to be reached by a short or long saccade, if one is needed, as described in Sections 2.4 and 3.2 Thirdly, the corners computed thus far at each fixation, and their curvature strengths κ, are used to attempt recognition of the object, as described in Chapter 3.

3.2 Sequential Computation of Fixation Points (redux)

The fixation selection algorithm used here differs from that above and in [87] in that it is configured for the specific application of object recognition. Fixation points are serially generated in an effort to find usable corner points for the object recognition process. In the absence of computed information, the first fixation point is taken to be the center of the image. This is largely an expedient, although we observe that according to Kowler [12], in humans participating in experimental visual search tasks, “saccades often land in the center of the entire stimulus configuration.” While this is certainly overly broad and task-dependent, in the absence of any other information this choice seems sensible, since the image center minimizes the maximum distance to any other point in the image, hence provides the best overall “look” when foveating.

A foveated edge map is computed over which a *fixation selection measure* $m_{i,j}$ is computed. The next fixation is coincident with the highest value of $m_{i,j}$. The search process then proceeds in two modes: first, local search, where small changes in fixation, or *short saccades* are computed as corners are locally searched for. Either a corner is found or the local search fails. If a corner is found, an attempt is made to recognize the object using the corners found thus far. The second mode is global search. If the object is not recognized with sufficient confidence, or if no new corner was found, then a larger change in fixation, or *long saccade,* is made. The criteria for fixation selection for the two modes differ, as described next.

When a new fixation point has been selected using a long saccade, a foveated edge map is created that is centered at the new fixation position. The local search algorithm is then invoked. At each local fixation coordinate, the corner strength measure is computed to decide whether a corner is deemed present. If it is not, then a short saccade is produced. A new foveated edge map is created, the corner measure, computed and, if necessary, another short saccade generated. Short saccades are generated until a corner is deemed found, or until a corner is not found. If seven short saccades are generated in succession without a corner being found (the computed corner strength measure is insufficiently large to positively identify a corner), then the search is deemed to have failed and a long saccade is generated, which moves the fovea to a different region of the image.

The fixation selection measure $m_{i,j}$ for both long and short saccades is composed of a variety of factors, most of which are detailed above: curvature strength $\kappa_{i,j}$; a re-visit inhibit factor $h_{i,j}$ that takes value P when a new fixation lies within N_h rows or columns of any P prior fixations ($N_h = 11$ in this implementation), thus discouraging long saccades from landing too close to previously-visited locations; the Euclidean distance $d_{i,j}$ that gives more distant potential fixations/corner greater weight, to compensate for the fact that the curvature strength $\kappa_{i,j}$ diminishes with distance owing to foveation; and the squared gradient magnitude $s_{i,j}$ of the gaussian-foveated image, which gives edges of larger magnitude greater weight in the fixation selection process. An important modification to the algorithm in [87] is made to ensure that only sharp corners are detected. Since high contrast edges of low curvature may attract fixations to uninteresting (in the sense of low-curvature corner information) regions of the image, the overall measure $m_{i,j}$ is set to zero if $\kappa_{i,j} < \tau$. The principle here is that for corner-based object recognition, points of higher curvature will yield better recognition discrimination than relatively smooth corners. This also effects the features that are used in the modified SIFT implementation, described below. In the rare instance that $\kappa_{i,j} < \tau$ for all (i, j), then τ is temporarily set to zero until a new long saccade is generated.

The overall fixation selection measure, a modification of (22), is given by:

$$m_{i,j} = \begin{cases} \left(\dfrac{\kappa_{i,j}}{1+\kappa_{i,j}}\right)^{2} \cdot \left(\dfrac{s_{i,j}}{\max(s_{i,j})+s_{i,j}}\right)^{2.5} \cdot \left(\dfrac{d_{i,j}}{\max(d_{i,j})+d_{i,j}}\right)^{2} \cdot \dfrac{1}{1+5\left(h_{i,j}\right)^{2}} & ; C = 0 \\ \left(\dfrac{\kappa_{i,j}}{1+\kappa_{i,j}}\right) \cdot \left(\dfrac{s_{i,j}}{1+50 s_{i,j}}\right) \cdot \left(\dfrac{d_{i,j}}{1+d_{i,j}}\right) & ; C > 0 \end{cases} \qquad (26)$$

where maxima are taken over the entire image, and where C controls the saccade lengths. When $C = 0$, a long saccade is generated, and when $C > 0$ a short saccade is generated. C is initially zero when the first (center) fixation is assigned, and incremented by unity with each succeeding saccade, until it is reset to zero. It is reset to zero whenever a corner is found according to (25), or whenever seven consecutive short saccades are generated ($C > 7$). The value of 7 is arbitrary and is normally never reached. Its purpose is to force a long jump should the fixation point ever reach an empty part of the image where short saccades are unable to remove it.

Long saccades ($C = 0$) are discouraged from approaching previous saccades owing to the presence of $h_{i,j}$ in (26), but not for short saccades ($C > 0$) which zero in on strong local corners.

The global maximum of $m_{i,j}$ provides the coordinates for the next fixation point. When the next fixation is made, the saccade length control variable C is incremented from 0. After reaching 7 (following 6 subsequent short saccades) it is reset to 0 forcing another long jump. This produces a sequence of one long saccade, intended to explore a new region of the image, followed by several short ones (fewer than 7), which pinpoint the corner accurately.

One particular of our implementation for this application is that corners are not found within a distance of 50 pixels from any border of the image edge. So that information is not lost near the image borders, the images are extended in size by a process of reflection them across each image boundary. The fixation selection measure and all of the constituent calculations are accomplished on the reflected image.

3.3 Foveated Object Recognition By Corner-Based SIFT

We have chosen to implement and demonstrate the principles of our method by adapting a well-known general approach. The Scale Invariant Feature Transform (SIFT) algorithm [80] operates by extracting low-level features from images. The features used in [80] are chosen to be invariant to translation, scale, rotation, and at least partially invariant to changing viewpoints, and variations in illumination. The feature representations computed by SIFT are argued to be analogous to those of neurons in inferior temporal cortex, a region believed used for object recognition by primates.

3.4 Review of SIFT

Here follows a brief, simplified description of the processing steps in the original SIFT algorithm [80]:

3.4.1 KEYPOINT LOCALIZATION. Difference-of-gaussians (DoG) bandpass filters are applied to the image over a range of dyadic scales. Local extrema of the DoG responses are found at each scale. These local extrema are candidate *keypoints,* or points of interest.

3.4.2 ORIENTATION INVARIANCE. A local neighborhood about each keypoint is defined. A rotation, scale- and translation-invariant keypoint descriptor is then computed, as follows. Over each neighborhood, a histogram is computed of gaussian-weighted summed gradient magnitudes at each orientation, and divided into 36 10° gradient orientation bins. The histogram is locally smoothed. The neighborhood is then rotated by the gradient orientation corresponding to the peak histogram value. Finally, a 16x16 sub-window is extracted following the rotation, yielding a rotation-invariant keypoint descriptor.

3.4.3 KEYPOINT DESCRIPTOR. The 16x16 rotated sub-window is blurred by a gaussian with space constant equal to that of the scale of the keypoint. The sub-window is weighted by another gaussian centered at the keypoint, thus masking small localization errors. The window is divided into 4x4 sub-windows; over each of which a histogram of gradient magnitudes is computed, and divided into 8 bins by gradient orientation. The result is a 4x4x8 or 128 element feature vector description of the image about the keypoint. This vector is trilinearly interpolated to reduce errors from small mislocations. Lastly, normalization and thresholding are applied to make the keypoint less sensitive to variations in illumination.

3.4.4 OBJECT RECOGNITION. A database is created of keypoints from known objects called the "training set," although no training takes place in the sense of a neural network. For each keypoint from an unknown object, a set of Euclidean distances is calculated. The ratio of the lowest distance to the next lowest between the keypoint and the database indicates whether a match has been found. A ratio much smaller than one indicates a high probability of a match.

Our adaption of the SIFT shares similar functionality to SIFT although it differs in regard to the feature sets used and in the foveated operating environment. Rather than defining keypoints to be scale-space extrema, we use corners located by the foveated algorithm described above and in [87]. In addition, the algorithm runs on only two scales, rather than a range of scales, and at both scales the same located corners are used. The viewpoint taken here is that since we seek local sharp corners on objects, it should be adequate to utilize highly localized representations of the discovered corners; of course, this also reduces the overall computational complexity. The second and third steps in our method are essentially the same as those in the original algorithm, though the keypoints receive no postprocessing.

3.5 SIFT-Like Preprocessing Using Foveated Corners

A blurred version of each detected corner at location **i** is available as the response of a local foveation filter (with local space constant $\sigma_{\mathbf{i}}$) that was used to discover the corner, as described above and in [87]. A small 16x16 neighborhood is extracted around this smoothed corner, which defines the finer spatial scale of the "keypoint" corner. To create a coarser scale, the local neighborhood is filtered again so that the local scale of the twice-blurred neighborhood is $2\sigma_{\mathbf{i}}$. These smoothed corner features and their local neighborhoods serve as the "keypoints" in our adaptation of SIFT.

We next describe our mechanism for handling orientation-invariance, which slightly modifies SIFT. Briefly, gradient orientation is computed as a gradient magnitude-weighted average of phases, taking into account phase wraparound. We have found that this delivers results that are superior, in our application, to the maxima of phase histograms method used in the original SIFT.

In a non-foveated multi-scale system, features such as edges and corners will appear, move, and disappear as the scale changes. In a foveated system, however, where the image is relatively sharp in the vicinity of the fovea, the *locations* of corners should be roughly independent of scale. Therefore, for corner location we only use the smaller scale, which greatly accelerates the algorithm.

Figure 11 shows two corners on a rectangle, designated Corner 1 and Corner 2, to be used to demonstrate the method of orientation handling. Figures 12(*a*) and 12(*b*) depicts details (zooms) of the corners after foveation filtering. The gradients of these are plotted as needle diagrams (length and orientation coding gradient magnitude and orientation, respectively) in Figures 12(*c*) and 12(*d*). Next, the histograms of are computed for each keypoint neighborhood, defined as the (gaussian-weighted) sum of gradient magnitudes corresponding to each orientation. The overall orientation of the keypoint neighborhood is then calculated as the weighted average of the histogrammed orientations

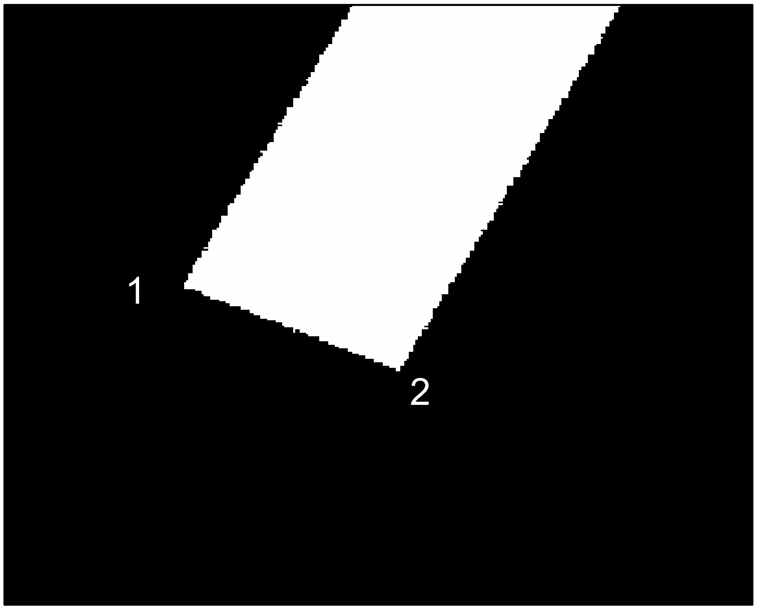
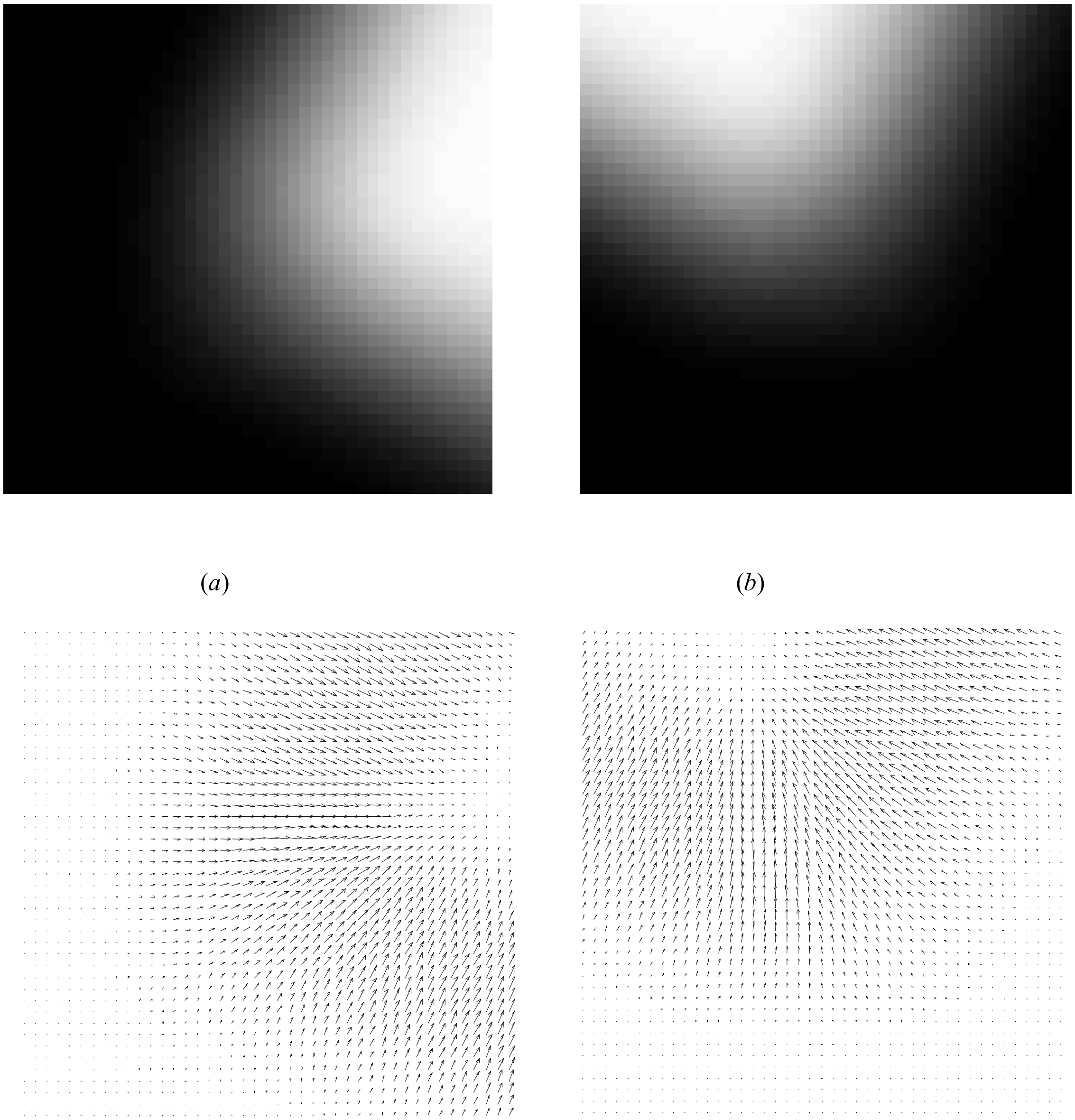


Figure 11. Two corners located on a rectangle.

$$o = \frac{\sum_i m_i p_i}{\sum_i m_i} \tag{27}$$

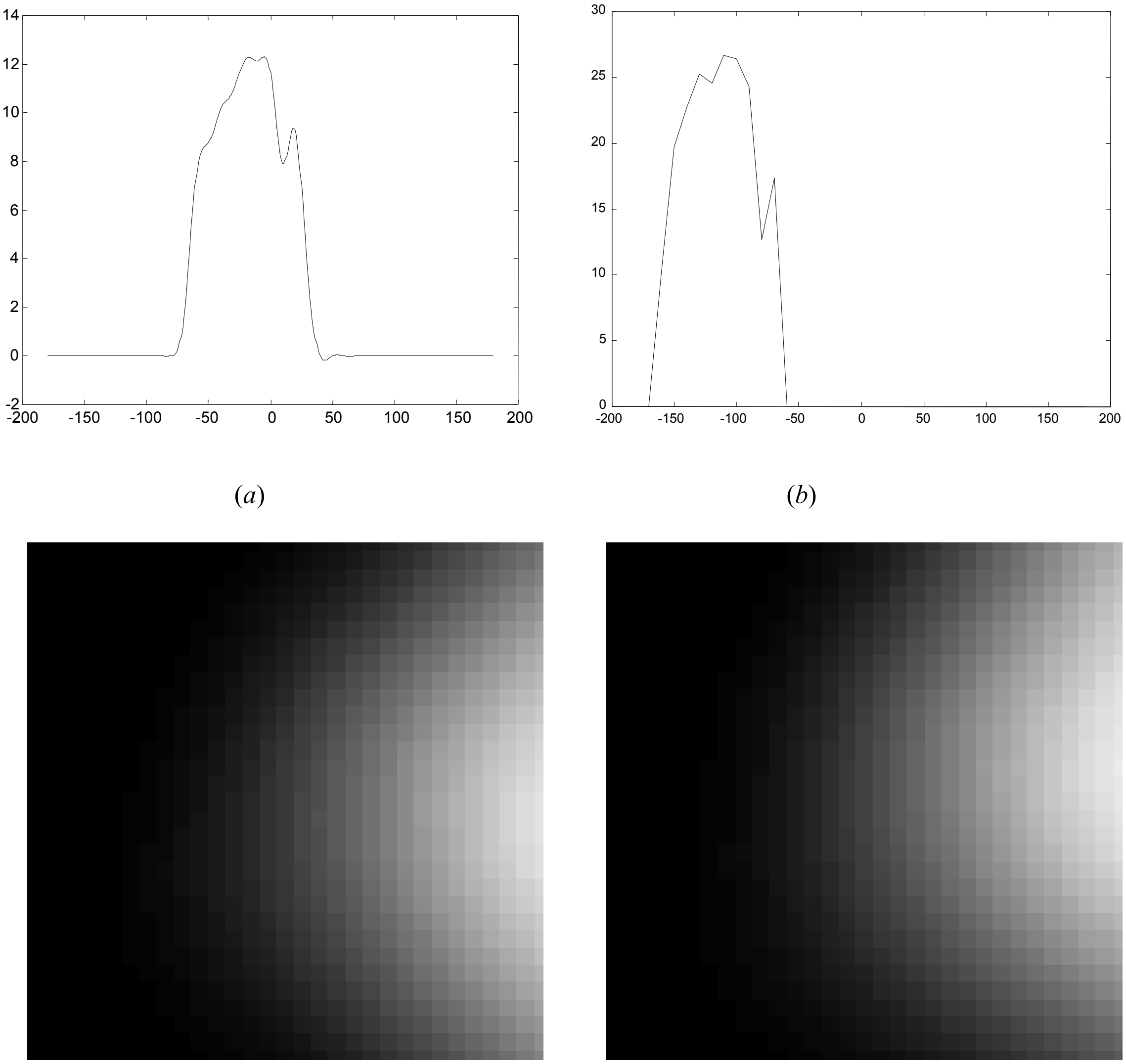
where the p_i are the neighborhood gradient orientations (phase unrolled for averaging), and the m_i are the associated gradient magnitudes. Continuing our idealized example, Figure 13(*a*) and 13(*b*) depict the gradient orientation histograms from the gradient maps in Figs. 12(*c*) and 12(*d*). Once the local keypoint neighborhood orientation *o* has been calculated using (6), the neighborhood is rotated CCW by the amount *o*. Figures 13(*c*) and 13(*d*) show the rotated Corner 1 and Corner 2, respectively.



(*a*) (*b*)

(*c*) (*d*)

Figure 12. Illustration of orientation pre-processing. (*a*) (*b*) Foveation-filtered versions of corners 1 and 2 in Figure 11, respectively. (*c*), (*d*) Needle diagrams showing the gradients of (*a*), (*b*), respectively, with needle length coding (scaled) gradient magnitude and needle angle indicating gradient orientation.



(*a*) (*b*)

(*c*) (*d*)

Figure 13. Rotation of corners. (*a*), (*b*) Histograms of gradient orientations demonstrated on Corners 1 and 2. The neighborhood orientations computed on (*a*), (*b*) using (6) were -5.0° and -114.3°, respectively. (*c*), (*d*) Gradient maps of the two corners following rotation.

3.6 Recognition Using Foveated Corners

Two related algorithms are used in the recognition system. One creates a database of SIFT-like features of known objects, with features from both scales intermixed. We call this the *training set.* The other identifies an unknown object by comparing its SIFT feature for each located corner with the features stored in the database. Of course, different objects may possess different numbers of corners. Figure 14 illustrates the organization of the database. Objects are grouped by category. The keypoints belonging to each object are placed together in the order in which the corner finding routine reports them. A separate file contains the categories and names of the object belonging to each entry.

Not all of the corners located by the corner finding algorithm are used as SIFT features. Some detected corners are duplicates or near-duplicates of ones that have already been located. Others have relatively low curvature or gradient strength. The routine that creates the database collects a set of trial corners, each element of which contains curvature and location, and filters out those having a curvature $\kappa \leq 0.2$ or a gradient strength $s \leq 0.2$ times the maximum gradient strength of the entire foveated image. The set of filtered trial corners is sorted by decreasing curvature. The corner having the highest curvature is saved into a final set. From then on, corners from the trial set are placed into the final set if they are at a distance greater than $\sqrt{10}$ pixels from all other corners in the final set. This prevents clustered corners from being used for recognition.

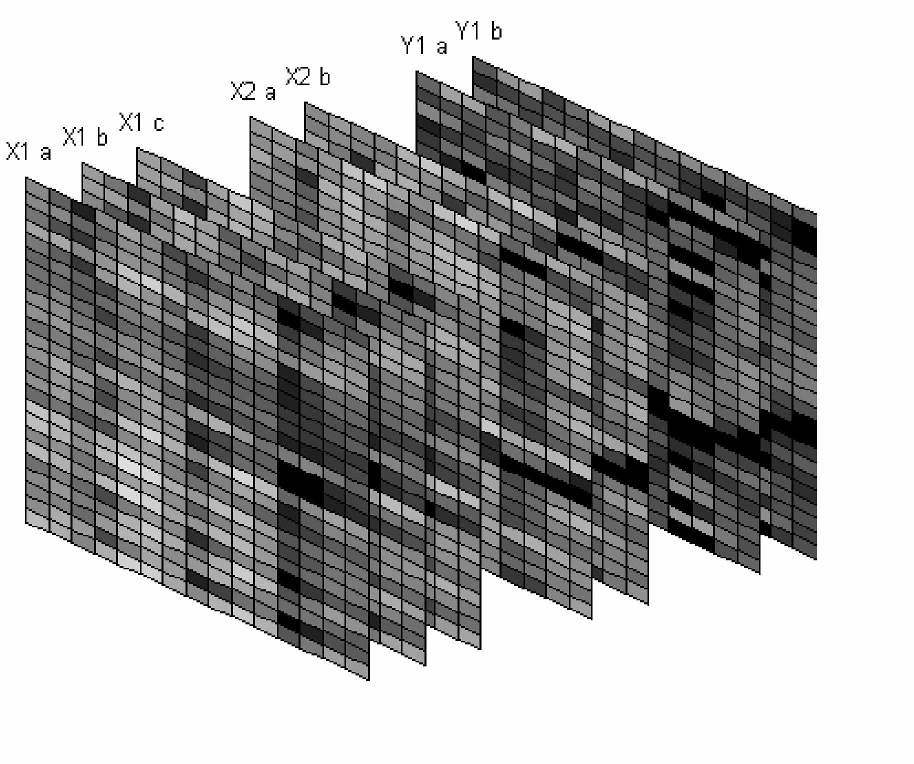


Figure 14. Illustration of the corner-keypoint database. Keypoints are shown as 2-D images. Objects labeled are X1, X2, and Y1. Here, X1 and X2 are examples of the same type of object and are adjacent in the database, while Y2 is a different type. Object X1 has three corners labeled *a* through *c*; X2 and Y1 each have two corners, labeled *a* and *b*.

Identification of an unknown object occurs somewhat differently, since corners are passed to the recognition algorithm "on the fly." Each corner on the unknown object is located, then filtered as described above, and if accepted, compared with the database. A correlation is calculated between it and each corner in the database:

$$C_{ab} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} b_{ij}}{\left(\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}^{2}\right)^{.5}\left(\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}^{2}\right)^{.5}} \tag{28}$$

where C_{ab} is the (normalized) correlation between features *a* and *b*.

Object recognition is based on a voting system that is based in turn on the correlations calculated in (28). The algorithm is complicated by the fact that there is no simple way to match corresponding corners between known (database) and unknown objects.

The following describe the database in set theoretic terms. The j^{th} object is represented in the database by the set of SIFT features at both scales in no particular order:

$$\mathbf{O}_j = \left\{ s_{j1}, s_{j2}, \ldots, s_{jN_j} \right\} \tag{29}$$

where s_{ji} is the SIFT feature from the i^{th} corner of the j^{th} object, and N_j is the number of features on the j^{th} object.

Several objects that are adjacent in the database are grouped together to form a *category*. The k^{th} category is:

$$\mathbf{C}_k = \{ \mathbf{O}_l \mid l \in K_k \} \tag{30}$$

where K_k is the set of objects in the k^{th} category.

Finally, the entire database is the set of the sets of categories in consecutive order.

$$\mathbf{DB} = \{ C_1, C_2, \ldots, C_N \} \tag{31}$$

where N is the number of object categories. In our simulations, we will use N= 8 categories of objects. Each category will receive a number of votes that is stored in an array having the same number of categories as the database:

$$\mathbf{V} = \{ v_1, v_2, \ldots, v_N \}. \tag{32}$$

The vector **V** is initialized to zero prior to corner discovery.

To recognize an unknown object, each corner is compared in succession with each corner in the database, and the correlations calculated using (28), creating a set of correlations whose cardinality equals that of the database.

We describe the sequence of steps in the object recognition process next. First, a corner on the object is located through the foveated search process. If it is of sufficient curvature and gradient strength, then the correlation C_{ij} between it and each corner in the database is calculated, where i represents the corner in the unknown object and j represents a corner in the database. The category, $j_{\max}$, corresponding to the maximum of all the correlations of corner i over the database is then determined

$$j_{\max} = \arg\max_{j} C_{ij} \,. \tag{33}$$

The category or categories corresponding to $j_{\max}$,which can be multivalued, each receive a vote:

$$\underset{k \in j_{\max}}{v_k = v_k + 1} \tag{34}$$

The algorithm accumulates information as it seeks corners and calculates SIFT features. The criterion for stopping the algorithm and declaring a winner is consistency of selected winners. After at least 8 corners have been selected, the algorithm will stop when at least 70% of the selected winners have the same classification. If no such condition is reached, the algorithm stops after 12 corners have been found.

When the stopping point is reached, the category with the highest number of votes is declared the winner. Since votes are small integer values, they are subject to frequent ties. If a tie is detected, the following procedure is used to calculate the winner.

Let M_j be the sum of the correlations of the first I features of the unknown object against the j^{th} object in the database.

$$M_j = \sum_{i=1}^{I} m_{ij} \tag{35}$$

Then, the winning category for tie breaking is the one which corresponds to the maximum value of M_j. A further tie is virtually impossible since M_j is the sum of floating point correlations.

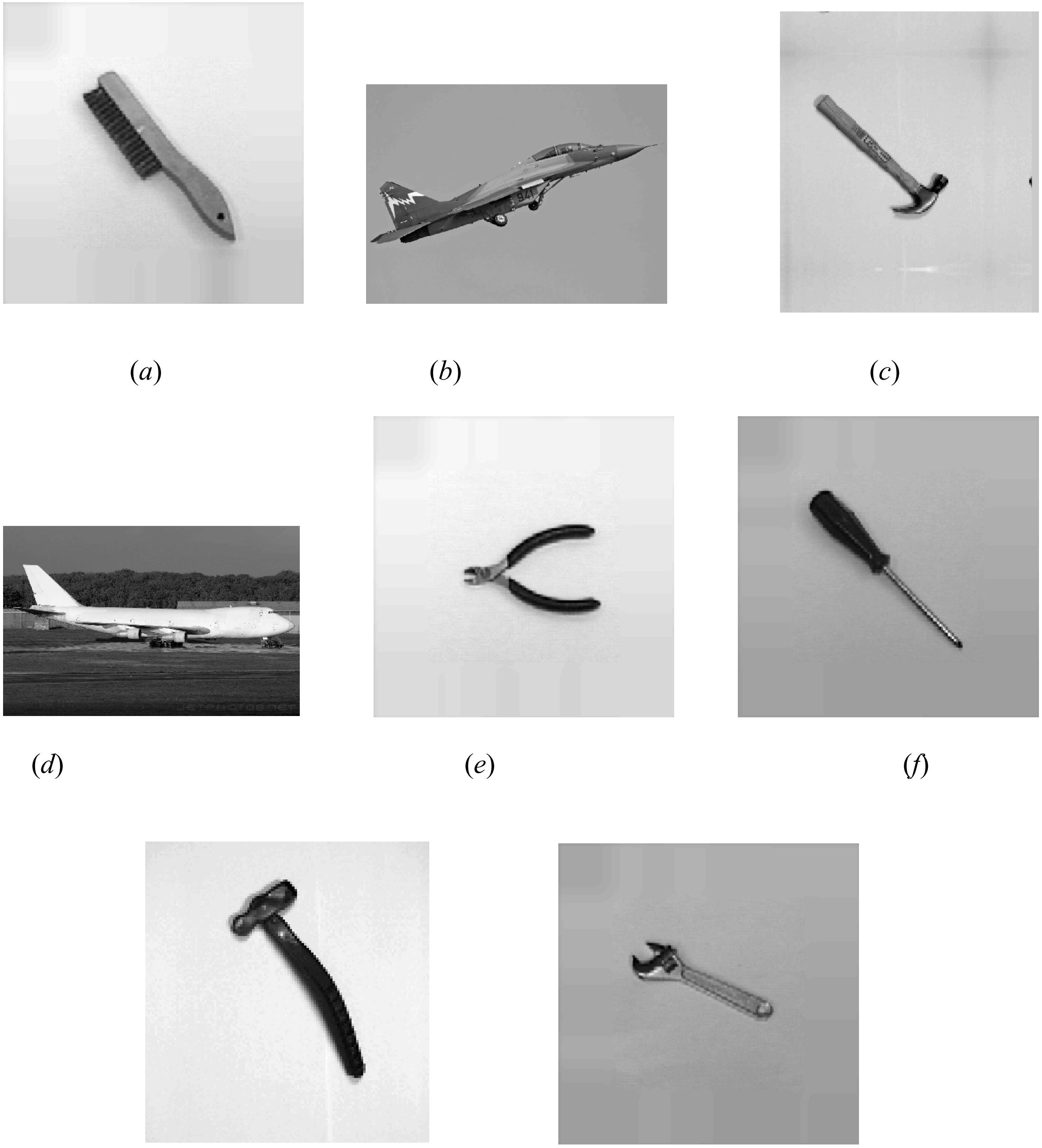
CHAPTER 4.

RESULTS AND DISCUSSION

In order to test the efficacy of the algorithm in terms of recognition performance, we assembled a diverse database containing eight categories of objects. These include hand tools from the Sclaroff database [97], the Rutgers University Tool Image Database [98], [78], and images of fighter and passenger jets. Figure 15 shows sample images taken from each category. It should be noted that these images are true gray-scale images that are not always easily segmented from their backgrounds. The object images also display internal non-uniformities, so that object recognition does not reduce to simple boundary matching. It is likely that internal corners are discovered on any of the objects during the foveated corner search process. Also, the images are not all of the same size. The overall database on which the algorithm was run was comprised of 78 objects with the number in each category shown in Table III. Table III also gives the number of objects in each category, the number of objects that were used for training, and the number of objects (and percentages) correctly identified. As shown, the overall recognition rate was 84.6% correct. Among those objects which were in the training set, there were 35 correct, 1 incorrect for a total of 97.2% correctly recognized objects. This indicates the approach to be resilient in consistently recognizing objects with known attributes. Among those not in the training set, the recognition rate was naturally lower, with 31 correct and 11 incorrectly recognized objects for an overall success rate of 73.8% correct.

It is worthwhile to make comparisons with other, related efforts. In [80], the SIFT algorithm was used to detect and locate objects in gray-level scenes, but did not report statistics on recognition performance. In a later study [95], the SIFT algorithm was tested on a database of 102 categories: the Caltech-101 image dataset [100] and a background category. They choose 15 or 30 training images at random from each category, place the remaining items in the test set. Their result was 51% correct for 15 training images, and 56% for 30 training images. In another experiment they detected and located automobiles in the UIUC car dataset [104]. The results varied between 90% and 100% correct depending on the details of the experiment. In [105], the authors tested a spatial pyramid matching technique for recognizing object categories in natural scenes, using a combination of SIFT and other features. A dataset with fifteen scene categories of 200 to 400 images each was used. They used 100 images per class for training and the remainder for testing, and obtained classification rates between 72.2% and 74.7%. In another set of experiments, they used the Caltech-101 dataset, which contains 31 to 800 images per category. Their best reported result was a recognition rate of 64.6% correct.

Although the greatest value of our approach lies in the exploration of the under-utilized, yet sensible approach of fixation-foveation in problems such as visual search and recognition, we are pleased by the performance of the algorithm developed using these principles. We envision that other foveated features, including original (foveated) SIFT features, features derived from foveated human attention studies [86], or features from receptive field models [106]-[108] might be used to good effect.



(*a*) (*b*) (*c*)

(*d*) (*e*) (*f*)

(*g*) (*h*)

Figure 15. Sample images of each category. (*a*) A brush; (*b*) a fighter jet; (*c*) a claw hammer; (*d*) a passenger jet; (*e*) a pair of pliers; (*f*) a screwdriver; (*g*) a sledge hammer; (*h*) a wrench.

Table III. Object categories and two-letter ID codes. The number of each object type is indicated, as well as the number used from each object type for training purposes. Finally, the number of correctly recognized objects in each dataset (and the percentage of correctly-recognized objects) are given.

Code	Tool/Object Type	Number Available	In Training Set	Num Correct	Pct. Correct
Br	Brush	5	3	3	60.0
Fj	Fighter Jet	7	3	5	71.4
Ha	Hammer (Claw)	5	3	4	80.0
Pj	Passenger Jet	8	4	8	100
Pl	Pliers	8	4	8	100
Sc	Screw Driver	10	5	9	90.0
Sl	Sledge Hammer	10	5	6	60.0
Wr	Wrench	25	9	23	92.0
	Total Objects	78	36	62	84.6

Foveated search techniques have great potential to improve the efficiency of robotic vision systems equipped with movable camera(s). The system here accomplishes foveation by software, hence requires more computation and memory than hardware foveation, e.g, using combinations of wide angle lenses (generating peripheral vision) and telephoto lenses (generating foveal vision) [101], other lens-based foveation methods that preferentially magnify the image near the "fovea" [102], and foveated CCD silicon retinae [103]. Such hardware foveation devices may lead to foveated object recognition systems requiring relatively little computing power relative to non-foveated systems.

4.1. Concluding Remarks

We have proposed and developed a gray-level object recognition system based on the sequential foveated detection of high-information corners used as features in a SIFT-like recognition system. The algorithm has two logical sections: The first uses a multi-fixating strategy for locating corners in natural images. It combines foveation, directional detection, and calculation of edge curvatures with generation of long and short saccades to establish foveal locations. Similar foveated search processes in biological vision systems served as an inspiration to this approach. We demonstrate the corner location system on a complex natural scene and on a view of a polyhedron. Results show that the algorithm performs well on strong edges with sharp corners and less well in areas of fine detail. The results demonstrated by our simulations of the corner search routine are quite promising, as they compare well with other studies on true gray-scale images.

The object recognition system uses the output of the corner finder routine as input to a modified SIFT transform. The system places SIFT features of known objects into a database and compares features from corners of an unknown object, as they are located.

The recognition system was tested on a series of grayscale images of hand tools and airplanes and like the corner finder, gave good results.

However, we feel that the primary contribution that we have made is the proof-of-principle that foveation can be used productively in computed visual processes such as search and recognition. Progress in this direction is still very young, of course, and our work has focused on searching for low-levels features only in a foveated environment, and using these features for planar object recognition. There remains considerable efforts towards understanding how foveation search and recognition might be deployed in unconstrained environments, 3-D environments, and in the presence of motion.

Applications might include robotics directed applications involving scenes containing corners. Naturally, the method could be improved by the introduction of richer features, such as color [99]. One can also imagine improving the algorithm by employing information regarding the spatial relationships of corners on objects.

Appendix

Four male observers, aged 32, 28, 29, 29, none of them familiar with the corner finding algorithm or the objectives of this work, were used for the experiment. All observers either had normal or corrected-to-normal vision. The stimuli consisted of two images: the lighthouse on the seashore, and the view of a polyhedron illustrated in Figure 4(*a*). The images were 1024 x 768 pixels and were displayed on a 21" monitor at a distance of 134 cm. from the observer. This set-up corresponds to about 60 pixels/degree of visual angle, so the images extend 20.67 by 12.8 degrees. Observers were presented with each image for 30 seconds and instructed to look for corners in the displayed image. About 50 fixations were recorded for each observer. Human eye movements were recorded using an SRI Generation V Dual Purkinje eye tracker. It has an accuracy of < 10' of arc, precision of ~ 1' of arc and a response time of under 1 ms. A bite bar and forehead rest was used to restrict the observer's head movements. The observer was first positioned in the eye tracker and a positive lock established onto the observer's eye. A linear interpolation on a 3x3 calibration grid was then done to establish the linear transformation between the output voltages of the eye tracker and the position of the observer's gaze on the computer display. The output of the eye tracker (horizontal and vertical eye position signals) was sampled at 200Hz and stored for offline data analysis.

Bibliography

[1] J. Alison Noble. Finding corners. *Image and Vision Computing*, 6:121--128, 1988.

[2] D. Marr and E. Hildreth. Theory of edge detection, *Proceedings Royal Society London B.*, 207:187--217, 1980.

[3] D. Marr. *Vision*. W. H. Freeman and Company, New York, 1982.

[4] R. Mehrotra, S. Nichani, and N. Ranganathan. Corner detection. *Pattern Recognition*, 23(11):1223--1233, 1990.

[4] J. Flynn and A. K. Jain. On reliable curvature estimation. *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, June 1989, 110--116.

[5] J. Basak and D. Mahata. A connectionist model for corner detection in binary and gray images. *IEEE Transactions on Neural Networks*, 11(2):1124--1132, 2000.

[6] K. Wiebe & A. Basu. Modeling ecologically specialized biological visual systems. *Pattern Recognition,* 30(10):1687--1703, 1997.

[7] S. Lee and A. C. Bovik. Fast algorithms for foveated video processing. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(2):149--162, 2003.

[8] P. T. Kortum and W. S. Geisler. Implementation of a foveated image coding system for image bandwidth reduction. *Proceedings of SPIE* 2657:350--360, 1996.

[9] W. N. Klarquist and A. C. Bovik. FOVEA: A foveated vergent active stereo vision system for dynamic three-dimensional scene recovery. *IEEE Transactions on Robotics and Automation*, 4(2):755--770, 1998.

[10] W. S. Geisler & M. S. Banks. Visual performance, in *Handbook of Optics, Volume 1: Fundamentals, Techniques, & Design Second Edition*, M. Bass(Ed.) New York, McGraw-Hill, 1995.

[11] E. Kowler. The role of visual and cognitive processes in the control of eye movement. In E. Kowler (Ed.) *Eye Movements and Their Role in Visual and Cognitive Processes*, Amsterdam, Elsevier, 1--63, 1990.

[12] G. Sperling. Comparison of perception in the moving and stationary eye. In E. Kowler (Ed.) E*ye Movements and Their Role in Visual and Cognitive Processes*, Amsterdam, Elsevier, 307--352, 1990.

[13] W. S Geisler and D.B. Hamilton. Sampling-theory analysis of spatial vision. *Journal of the Optical Society of America A (Optics, Image Science and Vision)*, 3(1):62-70, 1986.

[14] R, W, Rodieck. *The First Steps in Seein*g, Sinauer Associates Inc., Sunderland, MA, 1998.

[15] A. L. Yarbus. *Eye Movements and Vision*. Plenum Press, New York, 1967.

[16] L. Kitchen and A. Rosenfeld. Gray level corner detection. *Pattern Recognition Letters*, 1:95--102, 1982.

[17] C. Harris and M. Stephens. A combined corner and edge detector. *Fourth Alvey Vision Conference*, 147--151, 1988.

[18] F. Mokhtarian and R. Suomela. Robust image corner detection through curvature scale space. *IEEE Transactions on Pattern Analysis and Machine intelligence.* 20(12):1376--1381, 1998.

[19] S. M. Smith and J.M. Brady. SUSAN - a new approach to low level image processing. *International Journal of Computer Vision.* 23(1):45--78, 1997.

[20] J. F. Canny. *Finding Edges and Lines in Images*, AI Lab MIT Technical Report AD-A130824, 1983.

[21] Z. Wang, L. Lu and A. C. Bovik. Foveation scalable video coding with automatic fixation selection. *IEEE Transactions on Image Processing.* 12(2):243--254, 2003.

[22] D. H. Ballard and C. M. Brown. *Computer Vision.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[23] W. S. Geisler, and J. S. Perry. A Real-time foveated multiresolution system for low-bandwidth video communication. *Proceedings of SPIE volume 3299,*

Human Vision and Electronic Imaging III; Bernice E. Rogowitz, Thrasyvoulos N. Pappas; Eds. Jul 1998, 294--305.

[24] G. Guy and G. Medioni. Inferring global perceptual contours from local features. *International Journal of Computer Vision*, 20:113-133, 1996.

[25] U. Rajashekar, L. K. Cormack and A. C. Bovik. Point of gaze analysis reveals visual search strategies, *Proceedings SPIE Human Vision and Electronic Imaging IX*, 5292:1--22 ,San Jose, CA, Janurary 2004,.

[26] E. W. Weisstein. Relative entropy. From *MathWorld--A Wolfram Web Resource.*, http://mathworld.wolfram.com/RelativeEntropy.

[27] D. H. Johnson and S. Sinanović. *Symmetrizing the Kullback-Leibler Distance*, http://cmc.rice.edu/docs/docs/Joh2001Mar1Symmetrizi.

[29] A. Srivatsava, M.I. Miller and U. Grenander. Statistical models for bayesian object recognition. *The Handbook of Image and Video Processing* (A.C. Bovik, Ed.), 1341--1354, 2005.

[30] P. J. Burt. Smart sensing within a pyramid vision machine, *Proceedings of the IEEE*, 76(8):1006--1015, Aug. 1988.

[31] J. Najemnik and W.S. Geisler. Optimal eye movement strategies in visual search, *Nature*, vol. 434, pp. 387-391, 2005.

[32] R. Raj, W.S. Geisler, R.A. Frazor and A.C. Bovik. Contrast statistics for foveated visual systems: Fixation selection by minimizing contrast entropy. *Journal of the Optical Society of America A (Optics, Image Science and Vision)*, 22(10):2039--204

[33] U. Rajashekar and L.K. Cormack. Influence of structural cues in visual search revealed by gaze contingent image analysis. *Journal of Vision*, submitted.

[34] U. Rajashekhar and L.K. Cormack. Point of gaze analysis reveals visual search strategies, Proceedings of SPIE Conference on Human Vision and Electronic Imaging IX, volume 5292, San Jose, California, January 18-22, 2004.

[35] H. Sossa and A. Palomino. Model-based recognition of planar objects using geometric invariants, *Proceedings of the International Conference on Image Processing,* Lausanne, Switzerland, Sept. 16-19, 1996.

[36] J. Qiang, R.M. Haralick. Corner detection with covariance propagation, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Juan, Puerto Rico, June 17-19, 1997.

[37] Y.-H. Gu and T. Tjahjadi. Corner-based feature extraction for object retrieval, *Proceedings of the International Conference on Image Processing,* Kobe, Japan, Oct. 25-28, 1999.

[38] A. Rattarangsi and R.T. Chin. Scale-based detection of corners of planar curves, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 4(4):430--439, 1992.

[39] Y. H. Liu and X. Cai. An efficient and robust corner detection algorithm, *Proceedings of the Fifth World Congress on Intelligent Control and Automation,* Hangzhou, China, June 15-19, 2004.

[40] J.J. Clark and N.J. Ferrier. Modal control of an attentive vision system, *Second International Conference on Computer Vision*, 509--513, Dec. 1988.

[41] L.E. Wixson *Gaze Selection for Visual Search*. PhD dissertation, University of Rochester, Computer Science Dept, 1994.

[42] C.M. Privitera and L.W. Stark. Algorithms for defining visual regions-of-interest: comparison with eye fixations, *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 22(9):970--982, 2000.

[43] L. E. Wixson. Using intermediate objects to improve the efficiency of visual search. International Journal of Computer Vision, 12(2/3):209-230, 1994.

[44] R. P. N. Rao and D. H. Ballard. A multiscale filter bank approach to camera movement control in active vision systems. SPIE Volume 2354, pp. 105-116, 1994.

[45] R. P. N. Rao, and D.H. Ballard. Learning saccadic eye movements using multiscale spatial filters. *Advances in Neural Information Processing Systems,*

G. Tesauro, D. S. Touretzky and T. K. Leen (Eds.), Cambridge, MIT Press, pp. 893-900, 1995.

[46] R. P. N Rao, G. J. Zelinsky, M. M. Hayhoe, and D. H. Ballard. Modeling saccadic targeting in visual search. *Advances in Neural Information Processing Systems,* G. Tesauro, D.S. Touretzky and T.K. Leen (Eds.), Cambridge, MIT Press, 1995.

[47] B. Moghaddam and A. Pentland. Probabilistic visual learning for object detection. *Fifth International Conference on Computer Vision,* Cambridge, MA, June 1995.

[48] W. S. Gribble. Slow visual search in a fast-changing world, *Proceedings of the 1995 IEEE Symposium on Computer Vision (ISCV-95)*, Coral Gables, Florida, November 19-21, 1995.

[49] J. K. Tsotsos. On the relative complexity of active vs. passive visual search. *International Journal of Computer Vision*, 7(2):127--141, 1992.

[50] A. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 12(1):97-136, 1980.

[51] J. M. Wolfe, K. R. Cave, and S.L. Franzel. Guided search: An alternative to the modified feature integration model for visual search. *Journal of Experimental Psychology: Human Perception and Performance,* 15:419-433, 1989.

[52] W. S. Geisler and K. Chou. Separation of low-level and high-level factors in complex tasks: visual search. *Psychological Review*, 102(2):356-378, 1995.

[53] J. W. McGowan, E. Kowler, A. Sharma and C. Chubb. Saccadic localization of random dot targets. *Vision Research*, 38(6):895--909, 1998.

[54] J. Jonides, D. E. Irwin and S. Yantis. Integrating information from successive fixations. *Science*, vol. 215(4529):192--194, 1982.

[55] D. E. Irwin, S. Yantis and J. Jonides. Evidence against visual integration across saccadic eye movements. *Perception and Psychophysics*, 34(1): 49-57, 1983.

[56] D. E. Irwin. Lexical processing during saccadic eye movements. *Cognitive Psychology*, 36(1):1-27, 1998.

[57] D. E. Irwin, and R. Gordon. Eye movements, attention, and transsaccadic memory. *Visual Cognition*, 5(1-2):127-155, 1998.

[58] H. Deubel, D. Irwin and W. X. Schneider. The subjective direction of gaze shifts long before the saccade, in W. Becker, H. Deubel, and T. Mergner, eds., *Current Oculomotor Research: Physiological and Psychological Aspects* (New York: Plenum), 65--70, 1999.

[59] R. M. Klein and E. Hansen. On the relationship between shifts of attention and shifts of gaze. *Psychonomics Society Meeting*, Phoenix, Arizona, November 1979.

[60] R. M. Klein and E. Hansen. Shifts of attention & gaze: Evidence for independence. *Association for Research in Vision and Ophthalmology* Annual Meeting, Sarasota, Florida, April 1979.

[61] J. M. Wolfe, S. J. Butcher, C. Lee, and M. Hyle. Changing your mind: On the contributions of top-down and bottom-up guidance in visual search for feature singletons. *Journal of Experimental Psychology: Human Perception and Performance*, 29(2):483--502, 2003.

[62] J. M. Wolfe. Moving towards solutions to some enduring controversies in visual search. *Trends in Cognitive Sciences*, 7(2):70--76, 2003.

[63] J. M. Wolfe. Visual Memory: What do you know about what you saw? *Current Biology,* 8:R303-R304, 1998.

[64] T. S. Horowitz and J. M. Wolfe. Visual search has no memory. *Nature*, 357:575--577, 1998.

[65] K. R. Cave and J. M. Wolfe. The psychophysics of the binding problem. *Neuron*, 24(1):11--17, 1999.

[66] J. Epelboim, R. M. Steinman, E. Kowler, M. Edwards, Z. Pizlo, C. J. Erkelens and H. Collewijn. The function of visual search and memory in sequential looking tasks. *Vision Research*, 35:3401-3422, 1995.

[67] G. J. Zelinsky, R. P. Rao, M. M. Hayhoe, and D.H. Ballard. Eye movements during a realistic search task. *Investigative Ophthalmology and Visual Science Supplement.*, 37:S15, 1996.

[68] Z. Wang and A. C. Bovik. Embedded foveation image coding. *IEEE Transactions on Image Processing*, 10(10):1397-1410, October 2001.

[69] A. C. Bovik (Ed.). *The Handbook of Image and Video Processing, Second Edition*, Elsevier, 2005.

[70] A. P. Witkin. Scale-space filtering. *Eight International Joint Conference on Artificial Intelligence,* Karlsruhe, Germany, Aug 8-12, 1983.

[71] T. Lindeberg. Edge and ridge detection with automatic scale selection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR'96,* San Francisco, June 16-21, 117--154 1996.

[72] J. Shi and C. Tomasi. Good Features to Track, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR'94,* 593--600, 1994.

[73] T. Tommasini and A. Fusiello and E. Trucco, and V. Roberto. Making good features track better**.** *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR'98* , 178--183, 1998.

[75] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision,* Cambridge University Press, 2000.

[76] C. Schmidt, R. Mohr and C. Baukhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 27:151--172, 2000.

[77] C. Kenney, B. Manjunath, M. Zuliani. G. Hewer, and A. Van Nevel, A condition number for point matching with application to registration and post-registration

error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1437--1445, 2003.

[78] K. Siddiqi. A. Shokoufandeh, S. Dickinson and S. Zucker, ShockGraphs and Shape Matching, *IEEE International Conference on Computer Vision, Bombay*, January 4-7, 1998.

[79] I. Gordon and D. G. Lowe. Scene modelling, recognition and tracking with invariant image features. *International Symposium on Mixed Augmented Reality (ISMAR),* Arlington, VA, 110--119 Nov. 2004.

[80] D. G. Lowe. Object recognition from local scale-invariant features. *IEEE International Conference on Computer Vision*, Corfu, Greece, 1150--1157 Sep. 1999.

[81] I. D. Reid and D. W. Murray. Tracking Foveated Corner Clusters Using Affine Structure. *IEEE International Conference on Computer Vision*, Berlin, Germany. 76--83, May 1993.

[82] D. W Murray, P. F McLauchlan, I. D. Reid and P. M. Sharkey. Reactions to peripheral image motion using a head/eye platform. *IEEE International Conference on Computer Vision*, Berlin, Germany. 403--411, May 1993.

[83] J. H. van Hateren and A. van der Schaaf. Independent component filters of natural images compared with simple cells in primary visual cortex. *Proceedings of the Royal Society London B,* 265(1394):359–366 1998.

[84] M. K. Hu. Visual pattern recognition by moment invariants. IRE Transactions on Information Theory, IT-8(2):179--187, 1962.

[85] M. Teague. Image analysis via the general theory of moments. *Journal of the Optical Society of America A (Optics, Image Science and Vision)*, 70(8):920-930, Aug. 1980.

[86] U. Rajashekar, I. van der Linde, A. C. Bovik and L. K. Cormack. GAFFE: A gaze-attentive fixation finding engine. *IEEE Transactions on Image Processing*, to appear, 2008.

[87] T. Arnow and A. C. Bovik. Foveated visual search for corners. *IEEE Transactions on Image Processing,* 16(3):813-823, 2007.

[88] Y.Y.Chung, M.T.Wong, Neural Network Based Image Recognition System Using Geometrical Moment. *Proceedings of IEEE International Region 10 Annual Conference – Speech and Image Technologies for Computing and Telecommunications,* TENCON 97, 2-4 December, Brisbane Australia, 383-386, 1997.

[89] P. W. M. Tsang and T. S. Au. A genetic algorithm for projective invariant object recognition IEEE Region 10 Ann Conf, pp. 26-29, Nov. 1996.

[90] J. H. Cai and Z. Liu. Hidden Markov models with spectral features for 2D shape recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(12):1454-1458, 2001.

[91] F. Stein and G. Medioni. Efficient two dimensional object recognition. 10th International Conference on Pattern Recognition, 13--17, June 1990.

[92] A. Fred, J. Marques, and P. Jorge. Hidden Markov models vs. syntactic modeling in object recognition. *IEEE International Conference on Image Processing*, Santa Barbara, CA, Oct. 26--29, 1997.

[93] S. Helmer and D. G. Lowe. Object recognition with many local features. *Workshop on Generative Model Based Vision (GMBV)*, Washington, D.C., July 2004.

[94] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber and T. Poggio. Object recognition with cortex-like mechanisms. *IEEE Transaction on Pattern Analysis and Machine Intelligence,* 29(3):411--426, 2007.

[95] J. Mutch and D. G. Lowe. Multiclass object recognition with sparse, localized features. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR*), New York, NY, June 2006.

[96] T. Kadir and J. M. Brady. Scale, saliency and image description. *International Journal of Computer Vision,* 45(2):83--105, 2001.

[97] S. Sclaroff. Deformable Prototypes for encoding shape categories in image databases. *Pattern Recognition*. 30(4):627--642, 1997.

[98] S. Dickenson. The Rutgers University Tool Image Database, Online. Available at: http://www.cs.rutgers.edu/pub/sven/rutgers-tools.

[99] J. R. Jordan, W. S. Geisler, and A. C. Bovik. Color as a source of information in the stereo correspondence process. *Vision Research*, 30(12):1955-1970, 1990.

[100] F. Li R. Fergus,. P. Perona. One-shot Learning of object categories, IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(4):594--611, 2006.

[101] B. Scassellati. *A binocular foveated active vision system*, Massachusetts Institute of Technology Artificial Intelligence Laboratory, A.I. Memo No. 1628. Online. Available at http://hdl.handle.net/1721.1/6670, 1998.

[102] K. Kuniyoshi, N. Kita, K. Sugimoto, S. Nakamura, T. Suehiro. A foveated wide angle lens for active vision. *Proceedings of the IEEE International Conference on Robotics and Automation*, Nagoya, Japan, May 21-27, 1995.

[103] J. Skribanowitz and A. Konig. Dedicated frontends for embedded vision systems. *Proceedings 5th Biannual World Automation Congress,* Orlando, FL, 2002.

[104] Shivani Agarwal, Aatif Awan, and Dan Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1475--1490, 2004.

[105] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* New York, NY, 2: 2169--2178, June 2006.

[106] A. C. Bovik, M. Clark and W. S. Geisler. Multichannel texture analysis using localized spatial filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):55--73, 1990.

[107] M. Clark, A. C. Bovik and W. S. Geisler. Texture segmentation using Gabor modulation/ demodulation. *Pattern Recognition Letters*, 6:261--267, 1987.

[108] M. Clark and A. C. Bovik. Experiments in segmenting texton patterns using localized spatial filters. *Pattern Recognition*, 22(6):707--717, 1989.

[109] S. Lee, M. S. Pattichis, and A. C. Bovik. Foveated video compression with optimal rate control. IEEE Transactions on Image Processing, 10(7):977--992, 2001.

[110] J. Monaco and A. C. Bovik. Foveation and uncalibrated active stereo vision, *IEEE Transactions on Image Processin*g, to appear, 2008.

[111] S. Sclaroff and A. Pentland. Modal matching for correspondence and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. 17(6):545–561, 1995.

Vita

Thomas Louis Arnow was born in Detroit Michigan in 1946, the son of Harold Bernard and Harriette Louisa Simpson Arnow. He graduated from McGill University in 1968 with a bachelor's of science in chemistry. He graduated from the University of Texas at San Antonio in 1979 with a master's degree in mathematics, computer science, and systems design and from the University of Texas at Austin in 1991 with a master's degree in electrical engineering. He has worked as a programmer and computer consultant for many years and worked on various projects involving image and signal processing and human vision modeling. He is currently adjunct faculty at Palo Alto College, San Antonio, Texas.

Permanent address: 303 Mary Louise Drive
San Antonio, Texas 78201

This dissertation was typed by Thomas Louis Arnow.