Foveated Visual Search for Corners

Thomas Arnow, *Member, IEEE,* and Alan C. Bovik, *Fellow, IEEE*

Abstract

We cast the problem of corner detection as a *corner search process*. We develop principles of foveated visual search and automated fixation selection to accomplish the corner search, supplying a case study of both foveated search and foveated feature detection. 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 eye-tracker. The comparison of fixation patterns is made using an information-theoretic measure. Results show that the algorithm is a good locater of corners, but does not correlate particularly well with human visual fixations.

Index Terms

Foveated vision, corner detection, machine vision, feature detection.

I. Introduction

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 [1]. 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 [2], [3]; image and depth gradients [4]; other edge factors [5]; proximity between objects [3]; object similarity [6], [7], [8], and combinations of randomized saliency and proximity factors [9]. 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 [10], [11].

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 [12][3]. This density decreases rapidly with eccentricity. In the central fovea, receptors are packed at a density of about 120/degree [13], [14]. 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 [15].

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 [16], [17]. 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 70° [13], [14], [18], [19]. Certainly the computation of new fixations must be fast, automatic, and image-driven to accomplish visual search with active, mobile cameras or eyes [20], [4], [21].

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 [22], [23], [24]. These studies have revealed limitations imposed by low-level factors [24] and the relationship between stimuli and the distribution of

attention [22]. Other studies have revealed the rules regarding where a saccade will land on a complex form [25]. Several workers studied the problem of integrating information across eye movements [26], [27], [28], [29], [30], relating shifts in attention and gaze [31], [32], relating top-down and bottom-up search [61]-[62], and relating visual search with visual memory [33], [34], [35]. 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 [36]. Little is known about fixation-selection mechanisms, how attention is distributed over time [37], 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 [20], [21], [38]. 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 eye-tracking, head tracking and other physical measurements [39].

Less work has been done on foveated computer processing algorithms that do not require eye-tracking. Exceptions include early work by Burt [1] on scene analysis and Klarquist et al. [4] 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 eye-tracked 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 [40], [41], and also by primitive shape attributes [42], [43]. 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 [44], [45], [46], [47], [48], [49], [50], [51], [52], [53], [54], [55]. In their early seminal work on computational vision, Marr and Hildreth [56], [12] 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 [44], 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 [57] 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.* [58] extended their work by adding an algorithm to reject unreliable features. Schmidt et. al. [59] 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.* [60] derive a "condition number" to assess the sensitivity of feature/corner detectors to perturbations in feature position. Gordon and Lowe [61] used a scale invariant feature transform [62] 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 [63] 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 pseudo-foveated processing scheme with a small pseudo-fovea surrounded by a lower-resolution pseudo-periphery [64].

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.* [46] 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.

In this paper, 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 eye-tracker. The comparison is made using an information-theoretic measure.

II. Computing Edge and 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 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.

A. Foveation Filtering

There are several possible methods for creating foveated images, the most popular of are those based on spatial-domain *foveation filtering*, and those based on wavelet-domain foveation. Foveation filtering is the most straightforward method [20], [38], [65] wherein a bank of low-pass filters is applied to the image on a point-wise basis, with bandwidths monotonically decreasing with eccentricity. The filters used are usually symmetric, unit-volume 2-D Gaussians of the form [20], [38]

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

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 [21], [38], [65]. 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 in Eq. (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 in Eq. (1) is

$$\omega_c = \frac{\sqrt{2\ln 2}}{\sigma} \tag{2}$$

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, that are concentric about the fixation point. The innermost ring is actually a circle centered about 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:

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

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 trade-off between execution time and continuity at the ring boundaries. The image between the i^{th} and the $(i-1)^{th}$ 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

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

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 [38], [66], [65]:

$$CT(f,e) = CT_0 e^{\alpha f \frac{e+e_2}{2}} \tag{6}$$

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:

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

In arriving at this formula, Geisler and Perry [66] fit Eq.(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 Eq.(8) and using an average for the high and low values for CT_0 yields a numeric relationship between cutoff spatial frequency and eccentricity:

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

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

B. 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 as 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 [[67], and those which modify this process by using smoothing along preferred directions prior to differentiation, viz., anisotropic filtering [67]. 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 [39]. 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 $\bigtriangledown S_\sigma$ 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 $\angle \bigtriangledown S_\sigma$ estimates the direction normal to the edge:

$$\mathbf{n}_\sigma(x, y) = \frac{\bigtriangledown S_\sigma(x, y)}{|\bigtriangledown S_\sigma(x, y)|} \tag{11}$$

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

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

It is easily shown [39] that the zero crossings of Eq. (12) are conveniently the same as those of

$$D(x, y) = \bigtriangledown S_\sigma(x, y). \bigtriangledown [\bigtriangledown S_\sigma(x, y). \bigtriangledown S_\sigma(x, y)] \tag{13}$$

Discrete implementation of Eq.(13) is accomplished using space-varying discrete directional Gaussian derivatives $G_{\sigma,x}(x, y) = \partial / G_{\sigma,x}(x, y)/\partial x$ and $G_{\sigma,y}(x, y) = \partial / G_{\sigma,y}(x, y)/\partial y$ to compute the discrete gradient expressions $\bigtriangledown S_\sigma(i, j)$ 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 [68], [69] as the distance from the foveation point increases; it is possible that this outlook may provide valuable insights into foveated edge detection processes.

C. 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 $x = x(t), y = y(t)$ [48]:

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

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

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

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 Eq.(14) yields:

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

It can be easily shown from the definition that κ equals the reciprocal of the radius of curvature. The curvature measure Eq.(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 Eq.(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.

Mehrotra *et al.* [46] 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 [45] 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 [49] detector declares a corner if the minimum change produced by a shift exceeds some threshold.

The Plessey corner finder [49], [44] 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 [70]] 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 [50] developed a variable scale corner detector based on the curvature formula Eq.(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 Eq.(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.

III. 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 Sections II-B and II-C. 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 Eq. (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 Fig. 1(c) shows the same for the low curvature point B. Finally, Fig. 1(d) shows the original contour in Fig. 1(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

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

where the coordinates of the k^{th} fixation point are denoted by i_{fk} and j_{fk}. Whenever a fixation point is generated, a 25×25 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.

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

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

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:

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

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

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

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 Eq.(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:

- 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.
- 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 Eq. (18) in Eq. (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 Eq. (13) is 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.

IV. 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.

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 algorithms results against the visual fixations, measured by a precision eye-tracker, 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.

A. 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 [42].

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 eye-tracked 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 eye-tracked 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 Fig. 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

$$\mathcal{D}(p||q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)} \tag{23}$$

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 $\mathcal{D}(p,q) = 0$ if and only if $p(x) = q(x)$ [71]. Since there is no reason to prefer an asymmetry, we use the symmetric distance [72]:

$$\mathcal{DS}(p||q) = \frac{1}{\left(\frac{1}{\mathcal{D}(p,q)} + \frac{1}{\mathcal{D}(q,p)}\right)} \tag{24}$$

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).

B. 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 image of a polyhedron. We counted 85 vertices in this image and handpicked the coordinates of each using a graphics program with a cross-hair 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 Eq. (21)), and calculated the KLD between the algorithmic and handpicked vertices. Results are shown in Figures 4(a),4(b) and 4(c).

C. Comparison With Eye-tracked 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 eye-tracker. The experimental protocol is described in the Appendix.

Comparison	Subject			
	HHC	IVDL	UR	YL
Polyhedron	0.411	0.603	0.876	0.398
Polyhedron	0.317	0.492	1.065	0.408
Lighthouse	2.743	2.216	2.505	1.927

TABLE I

KLD VALUES FOR (A) POLYHEDRON IMAGE - EYE-TRACKER VS. HAND PICKED 50 FIXATIONS (B) POLYHEDRON IMAGE - EYE-TRACKER VS. ALGORITHM 50 FIXATIONS (C) LIGHTHOUSE IMAGE - EYE-TRACKER VS. ALGORITHM 50 FIXATIONS

Comparison	
Polyhedron - 50 fixations	0.269
Polyhedron - 85 fixations	0.132

TABLE II

KLD VALUES FOR ALGORITHM VS. HANDPICKED FOR 50 AND 85 FIXATIONS

Here we test the algorithm on the two images shown above: the lighthouse on a seashore and the polyhedron. Table I provides KLD values of algorithm vs. eye-tracker 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 Eq. (24). Each subject ran about 50 fixations on the eye-tracker, 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 eye-tracker comparison is consistently worse than the handpicked versus algorithm. The eye-tracker 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 eye-tracker 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 eye-tracker ones. This further demonstrates that the algorithm locates corners far more accurately than human subjects.

Figures 6 and 7 show algorithmic versus eye-tracker 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 [73] and on a natural scene from the van Hateren database of naturalistic images [74].

V. CONCLUDING REMARKS

We have presented a foveated, multi-fixating strategy for locating corners in natural images. Our approach combines foveation, directional detection, and calculation of edge curvatures with generation of long and short saccades to establish foveal locations. We demonstrate the 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. Applications might include robotics directed applications involving scenes containing corners.

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 Fig. 1. The images were 1024×768 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 $\sim 1'$ of arc and a response time of under 1ms. 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 3×3 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.

ACKNOWLEDGMENT

Umesh Rajashekar for his help with the eye-tracker, for providing some useful references, and for providing the natural image used. Kalpana Seshadrinathan, for help with preparing the manuscript.

REFERENCES

[1] A. Srivastava, M. I. Miller, and U. Grenader, *The Handbool of Image and Video Processing*. Elsevier, 2005, ch. Statistical models for Bayesian object recognition, pp. 1341–1354.

[2] J. Clark and N. Ferrier, "Modal control of an attentive vision system," in *Computer Vision., 1988. Second International Conference on*, 1988, pp. 514–523.

[3] L. Wixson and D. Ballard, "Using intermediate objects to improve the efficiency of visual search," *International Journal of Computer Vision*, vol. 12, no. 2, pp. 209–230, 1994.

[4] W. Klarquist and A. Bovik, "Fovea: a foveated vergent active stereo vision system for dynamic three-dimensional scene recovery," *Robotics and Automation, IEEE Transactions on*, vol. 14, no. 5, pp. 755–770, 1998.

[5] C. Privitera and L. Stark, "Algorithms for defining visual regions-of-interest: comparison with eye fixations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 9, pp. 970–982, 2000.

[6] R. Rao and D. Ballard, "A multiscale filter bank approach to camera movement control in active vision systems," in *Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 2345. Dept. of Comput. Sci., Rochester Univ., NY, USA: SPIE, 1994, pp. 105–116.

[7] R. Rao, D. Ballard, G. Tesauro, D. Touretzky, and T. Leen, "Learning saccadic eye movements using multiscale spatial filters," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. Touretzky, and T. Leen, Eds. Dept. of Comput. Sci., Rochester Univ., NY, USA: MIT Press, 1995, pp. 893–900.

[8] R. Rao, G. Zelinsky, M. Hayhoe, D. Ballard, D. Touretzky, M. Mozer, and M. Hasselmo, "Modeling saccadic targeting in visual search," in *Advances in Neural Information Processing 8. Proceedings of the 1995 Conference*, D. Touretzky, M. Mozer, and M. Hasselmo, Eds. Dept. of Comput. Sci., Rochester Univ., NY, USA: MIT Press, 1996, pp. 830–836.

[9] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object detection," in *Proceedings. Fifth International Conference on Computer Vision (Cat. No.95CB35744)*. Media Lab., MIT, Cambridge, MA, USA: IEEE Comput. Soc. Tech. Committee on Pattern Anal. & Machine Intelligence, 1995, pp. 786–793.

[10] W. Gribble, "Slow visual search in a fast-changing world," in *Proceedings International Symposium on Computer Vision (Cat. No.95TB100006)*. Dept. of Electr. & Comput. Eng., Texas Univ., Austin, TX, USA: IEEE Comput. Soc. Tech. Committee for Pattern Anal. & Machine Intelligence (PAMI), 1995, pp. 515–520.

[11] J. Tsotsos, "On the relative complexity of active vs. passive visual search," *International Journal of Computer Vision*, vol. 7, no. 2, pp. 127–141, 1992.

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

[13] E. Kowler, *Eye Movements and their Role in Visual and Cognitive Processes*. Amsterdam: Elsevier, 1990, ch. The role of visual and cognitive processes in the control of eye movement, pp. 1–63.

[14] G. Sperling, *Eye Movements and their Role in Visual and Cognitive Processes*. Amsterdam: Elsevier, 1990, ch. Comparison of perception in the moving and stationary eye, pp. 307–352.

[15] W. S. Geisler and D. B. Hamilton, "Sampling-theory analysis of spatial vision," *Journal of the Optical Society of America A*, vol. 8, pp. 62–70, 1986.

[16] W. S. Geisler and M. S. Banks, *Handbook of Optics: Fundamentals, Techniques and Design*, 2nd ed. New York: McGraw -Hill, 1995, vol. 1, ch. Visual Performance.

[17] K. J. Wiebe and A. Basu, "Modelling ecologically specialized biological visual systems," *Pattern Recognition*, vol. 30, no. 10, pp. 1687–1703, Oct. 1997.

[18] R. W. Rodieck, *R. W. Rodieck*. Sunderland, MA: Sinauer Associates Inc., 1998.

[19] A. L. Yarbus, *Eye movements and Vision*. New York: Plenum Press, 1967.

[20] S. Lee and A. Bovik, "Fast algorithms for foveated video processing," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 2, pp. 149–162, 2003.

[21] P. Kortum and W. S. Geisler, "Implementation of a foveated image coding system for image bandwidth reduction," in *Proc. SPIE Int. Soc. Opt. Eng.*, vol. 2657, no. 1. San Jose, CA, USA: SPIE, Apr. 1996, pp. 350–360.

[22] A. M. Treisman and G. Gelade, "A feature-integration theory of attention," *Cognitive Psychology*, vol. 12, no. 1, pp. 97–136, Jan. 1980.

[23] J. M. Wolfe, K. R. Cave, and S. L. Franzel, "Guided search : An alternative to the feature integration model for visual search," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 15, no. 3, pp. 419–433, Aug. 1989.

[24] W. S. Geisler and K. L. Chou, "Separation of low-level and high-level factors in complex tasks: visual search." *Psychol Rev*, vol. 102, no. 2, pp. 356–378, Apr 1995.

[25] J. McGowan, E. Kowler, A. Sharma, and C. Chubb, "Saccadic localization of random dot targets," *Vision Research*, vol. 38, no. 6, pp. 895–909, 1998.

[26] J. Jonides, D. E. Irwin, and S. Yantis, "Integrating visual information from successive fixations." *Science*, vol. 215, no. 4529, pp. 192–194, Jan 1982.

[27] D. E. Irwin, S. Yantis, and J. Jonides, "Evidence against visual integration across saccadic eye movements," *Perception and Psychophysics*, vol. 34, pp. 49–57, 1983.

[28] D. E. Irwin, "Lexical processing during saccadic eye movements,," *Cognitive Psychology*, vol. 36, no. 1, pp. 1–27, June 1998.

[29] D. Irwin and R. Gordon, "Eye movements, attention and trans-saccadic memory," *Visual Cognition*, vol. 5, no. 1, pp. 127–155, 1998.

[30] H. Deubel, D. Irwin, and W. X. Schneider, *Current Oculomotor Research: Physiological and Psychological Aspects*. New York: Plenum, 1999, ch. The subjective direction of gaze shifts long before the saccade, pp. 65–70.

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

[32] ——, "Shifts of attention & gaze: Evidence for independence," in *Association for Research in Vision and Ophthalmology Annual Meeting*, Sarasota, Florida, April 1979.

[33] J. M. Wolfe, "Visual memory: What do you know about what you saw?" *Current Biology*, vol. 8, no. 9, pp. R303–R304, Apr. 1998.

[34] T. Horowitz and J. Wolfe, "Visual search has no memory," *Nature*, vol. 394, no. 6693, pp. 575–577, 1998.

[35] J. M. Wolfe and K. R. Cave, "The psychophysical evidence for a binding problem in human vision," *Neuron*, vol. 24, no. 1, pp. 11–17, Sept. 1999.

[36] J. Epelboim, R. Steinman, E. Kowler, M. Edwards, Z. Pizlo, C. Erkelens, and H. Collewijn, "The function of visual search and memory in sequential looking tasks," *Vision Research*, vol. 35, no. 23, pp. 3401–3422, 1995.

[37] G. J. Zelinsky, R. P. Rao, M. M. Hayhoe, and D. H. Ballard, "Eye movements during a realistic search task," *Investigative Opthalmology and Visual Science Supp.*, vol. 37, no. 3, p. S15, 1996.

[38] Z. Wang, L. Lu, and A. Bovik, "Foveation scalable video coding with automatic fixation selection," *IEEE Transactions on Image Processing*, vol. 12, no. 2, pp. 243–254, 2003.

[39] J. F. Canny, "Finding edges and lines in images," AI Lab, MIT, Tech. Rep. AD-A130824, 1983.

[40] R. G. 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.

[41] A. Hajemnik and W. Geisler, "Optimal eye movement strategies in visual search," *Nature*, vol. 434, no. 7031, pp. 387–391, 2005.

[42] U. Rajashekar, L. Cormack, and A. Bovik, "Point-of-gaze analysis reveals visual search strategies," in *Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 5292, no. 1. Dept. of Electr. & Comput. Eng., Univ. of Texas, Austin, TX, USA: SPIE-Int. Soc. Opt. Eng, 2004, pp. 296–306.

[43] U. Rajashekar, A. C. Bovik, and L. K. Cormack, "Visual search in noise: revealing the influence of structural cues by gaze-contingent classification image analysis." *J Vis*, vol. 6, no. 4, pp. 379–386, 2006.

[44] J. Noble, "Finding corners [image edge detection]," *Image and Vision Computing*, vol. 6, no. 2, pp. 121–128, 1988.

[45] P. Flynn and A. Jain, "On reliable curvature estimation," in *Computer Vision and Pattern Recognition, 1989. Proceedings CVPR '89., IEEE Computer Society Conference on*, 1989, pp. 110–116.

[46] R. Mehrotra, S. Nichani, and N. Ranganathan, "Corner detection," *Pattern Recognition*, vol. 23, pp. 1223–1233, 1990.

[47] J. Basak and D. Mahata, "A connectionist model for corner detection in binary and gray images," *Neural Networks, IEEE Transactions on*, vol. 11, no. 5, pp. 1124–1132, 2000.

[48] L. Kitchen and A. Rosenfeld, "Gray-level corner detection," *Pattern Recognition Letters*, vol. 1, no. 2, pp. 95–102, Dec. 1982.

[49] C. Harris and M. Stephens, "A combined corner and edge detector," in *Fourth Alvey Vision Conference*, 1988, pp. 147–151.

[50] F. Mokhtarian and R. Suomela, "Robust image corner detection through curvature scale space," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 12, pp. 1376–1381, 1998.

[51] H. Sossa and A. Palomino, "Model-based recognition of planar objects using geometric invariants," in *Proceedings. International Conference on Image Processing (Cat. No.96CH35919)*. Centro Nacional de Calculo-IPN, Mexico City, Mexico: IEEE Signal Process. Soc, 1996, vol. vol.3, pp. 603–606.

[52] Q. Ji and R. Haralick, "Corner detection with covariance propagation," in *Proceedings. 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.97CB36082)*. Intelligent Syst. Lab., Washington Univ., Seattle, WA, USA: IEEE Computer. Soc. Tech. Committee on Pattern Anal. & Machine Intelligence, 1997, pp. 362–367.

[53] Y.-H. Gu and T. Tjahjadi, "Corner-based feature extraction for object retrieval," in *Proceedings 1999 International Conference on Image Processing (Cat. 99CH36348)*. Dept. of Signals & Syst., Chalmers Univ. of Technol., Goteborg, Sweden: IEEE Signal Process. Soc., 1999, vol. vol.1, pp. 119–123.

[54] A. Rattarangsi and R. Chin, "Scale-based detection of corners of planar curves," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 4, pp. 430–449, 1992.

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

[56] D. Marr and E. Hildreth, "Theory of edge detection." *Proc R Soc Lond B Biol Sci*, vol. 207, no. 1167, pp. 187–217, Feb 1980.

[57] J. Shi and C. Tomasi, "Good features to track," in *Proceedings 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.94CH3405-8)*. Dept. of Comput. Sci., Cornell Univ., Ithaca, NY, USA: IEEE Comput. Soc. Tech. Committee on Pattern Anal. & Machine Intelligence, 1994, pp. 593–600.

[58] T. Tommasini, A. Fusiello, E. Trucco, and V. Roberto, "Making good features track better," in *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.98CB36231)*. Machine Vision Lab., Udine Univ., Italy: IEEE Comput. Soc. Tech. Committee on Pattern Anal. & Machine Intelligence, 1998, pp. 178–183.

[59] C. Schmid, R. Mohr, and C. Bauckhage, "Evaluation of interest point detectors," *International Journal of Computer Vision*, vol. 37, no. 2, pp. 151–172, 2000.

[60] C. Kenney, B. Manjunath, M. Zuliani, G. Hewer, and A. Van Nevel, "A condition number for point matching with application to registration and postregistration error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 11, pp. 1437–1454, 2003.

[61] I. Gordon and D. G. Lowe, "Scene modelling, recognition and tracking with invariant image features," in *International Symposium Mixed Augmented Reality (ISMAR)*, Arlington, VA, November 2004, pp. 110–119.

[62] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Dept. of Comput. Sci., British Columbia Univ., Vancouver, BC, Canada: IEEE Comput. Soc. Tech. Committee on Pattern Anal. & Machine Intelligence, 1999, vol. vol.2, pp. 1150–1157.

[63] I. Reid and D. Murray, "Tracking foveated corner clusters using affine structure," in *[1993] Proceedings Fourth International Conference on Computer Vision*. Dept. of Eng. Sci., Oxford Univ., UK: IEEE, 1993, pp. 76–83.

[64] D. Murray, P. McLauchlan, I. Reid, and P. Sharkey, "Reactions to peripheral image motion using a head/eye platform," in *[1993] Proceedings Fourth International Conference on Computer Vision*. Dept. of Eng. Sci., Oxford Univ., UK: IEEE, 1993, pp. 403–411.

[65] Z. Wang and A. Bovik, "Embedded foveation image coding," *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1397–1410, 2001.

[66] W. Geisler and J. Perry, "A real-time foveated multiresolution system for low-bandwidth video communication," in *Proceedings of the SPIE - The International Society for Optical Engineering*, vol. 3299. Center for Vision & Image Sci., Texas Univ., Austin, TX, USA: SPIE: Soc. Imaging Sci. & Technol, 1998, pp. 294–305.

[67] A. C. Bovik, Ed., *The Handbook of Image and Video Processing*, 2nd ed. Elsevier, 2005.

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

[69] T. Lindeberg, "Edge detection and ridge detection with automatic scale selection," in *Proceedings 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.96CB35909)*. CVAP, R. Inst. of Technol., Stockholm, Sweden: IEEE Comput. Soc. Tech. Committee on Pattern Analysis & Machine Intelligence, 1996, pp. 465–470.

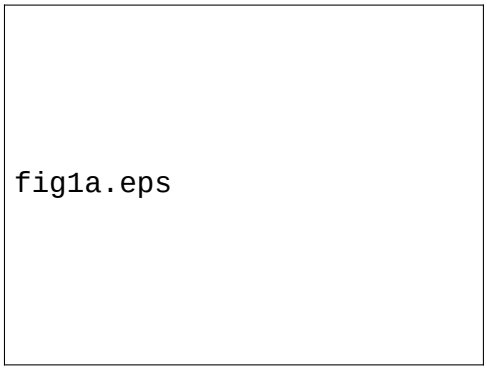
[70] S. M. Smith and J. M. Brady, “Susan - a new approach to low level image processing,” *International Journal of Computer Vision*, vol. 23, no. 1, pp. 45–78, 1997.

[71] E. W. Weisstein. Relative entropy. [Online]. Available: http://mathworld.wolfram.com/RelativeEntropy

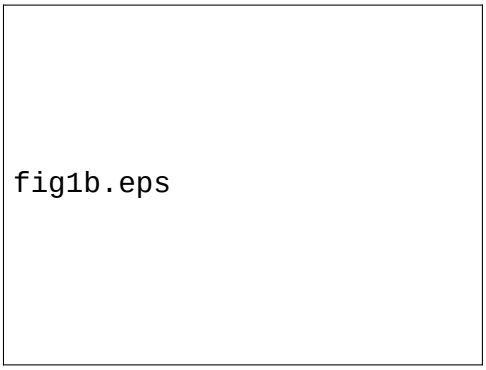
[72] Symmetrizing the kullback-leibler distance. [Online]. Available: http://cmc.rice.edu/docs/docs/Joh2001Mar1Symmetrizi

[73] K. Siddiqi, A. Shokoufandeh, S. Dickenson, and S. Zucker, “Shock graphs and shape matching,” in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*. Center for Comput. Vision & Control, Yale Univ., New Haven, CT, USA: Narosa Publishing House, 1998, pp. 222–229.

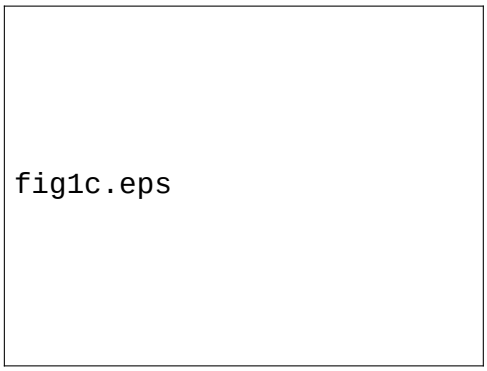
[74] J. H. van Hateren and A. van der Schaaf, “Independent component filters of natural images compared with simple cells in primary visual cortex.” *Proc Biol Sci*, vol. 265, no. 1394, pp. 359–366, Mar 1998.



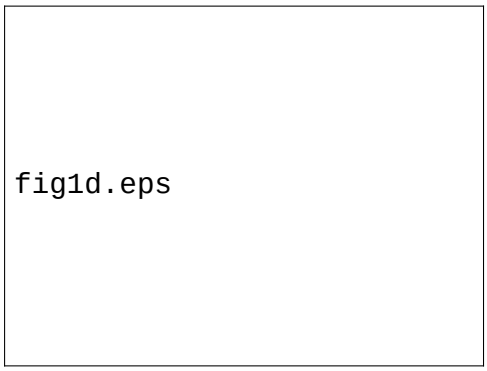
(a) Sample line of varying curvature



(b) Edge data points and Fit Polynomial



(c) Edge Data Points and Fit Polynomial



(d) Curvature Map

Fig. 1. Generation of curvature map on smoothed zero-crossing contour. (a) Zero-crossing contour; (b) Local polynomial fit near the low-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 1(a) with curvature coded as intensity (darker = higher curvature).

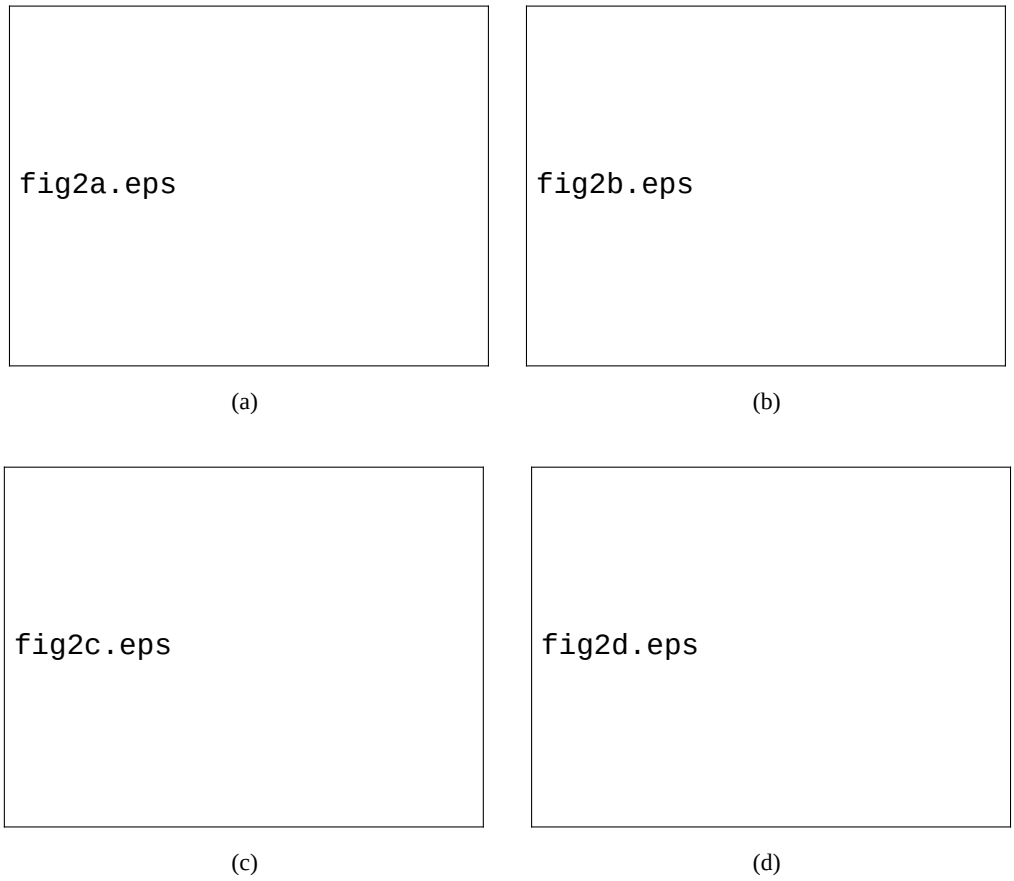
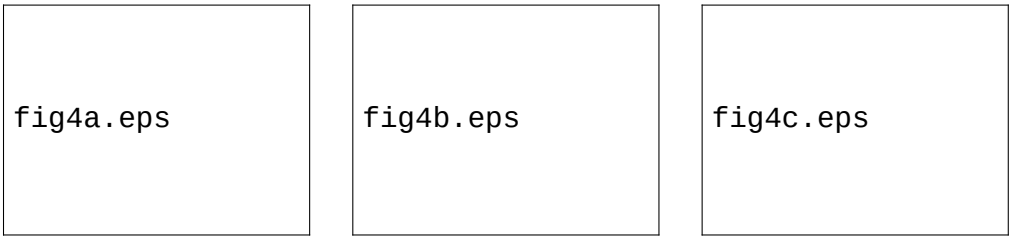


Fig. 2. Example of calculation of subsequent fixations. (a) Original *lighthouse* image (b) Foveated version by space-varying Gaussian filtering (c) Foveated edge map by foveated Canny edge detection (d) Foveated curvature map.

Fig. 3. Sample fixation points with Gaussian interpolation



(a) (b) (c)

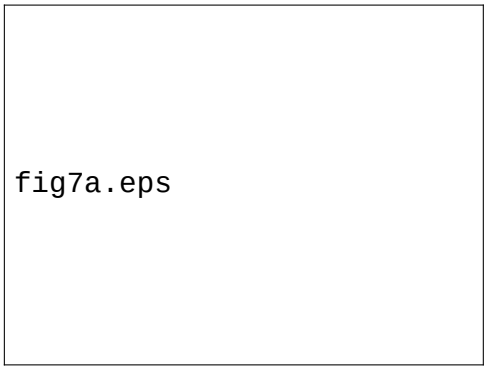
Fig. 4. Polyhedron image used to compare the corner detection algorithm with handpicked corners for 85 fixations. (a) For τ=0.1, the KLD = 0.1323. (b) For τ=0.2, the KLD=0.1490. (c) For τ=0.3, the KLD=0.3702.

Fig. 5. A comparison of the accuracy of corner locations, algorithm vs. eye-tracker for 50 fixations

(a) YL-Lighthouse (b) UR-Lighthouse

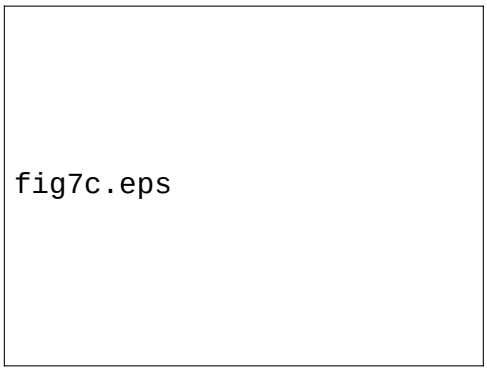
(c) IVDL-Lighthouse (d) HHC-Lighthouse

Fig. 6. Four observers vs. 50 algorithmic fixations - lighthouse



(a) YL-Polygon, Hand Picked

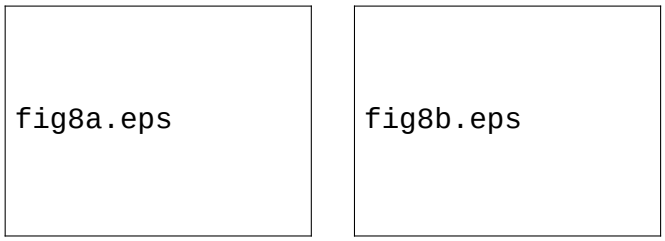
(b) UR-Polygon, Hand Picked



(c) IVDL-Polygon, Hand Picked

(d) HHC-Polygon, Hand Picked

Fig. 7. Four observers vs. 50 algorithmic fixations - lighthouse



(a) (b)

Fig. 8. Test of algorithm on tool image (left) and natural scene (right)