Eye detection in a face image using linear and nonlinear filters

Saad A. Sirohey¹, Azriel Rosenfeld*

Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, MD 20742-3275, USA

Received 25 August 1999; received in revised form 12 May 2000; accepted 30 May 2000

Abstract

This paper describes two methods of eye detection in a face image. The face is first detected as a large flesh-colored region, and anthropometric data are then used to estimate the size and separation of the eyes. When a linear filtering method, using filters based on Gabor wavelets, was then applied to detect the eyes in the gray-level image of the face, the detection rate was good (80% on one dataset, 95% on another), but there were many false alarms. A nonlinear filtering method was therefore developed to detect the corners of the eyes in the color image of the face. This method gave a 90% detection rate with no false alarms. © 2001 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

Keywords: Face detection; Eye detection; Linear filters; Nonlinear filters; Corner detection

1. Introduction

Detection of faces and facial features using machine vision techniques has many useful applications. These applications range from feature analysis, i.e., eye, nose, and mouth detection, speech reading, facial expression analysis, etc., to holistic analysis, i.e., face recognition, gender classification, etc. Though human beings accomplish these tasks countless times a day, they are very challenging for machine vision. For a comprehensive survey see Ref. [1]. This paper deals with the problem of detecting faces in color images, with the specific goal of locating the eyes. A review of the relevant literature is given in Section 2.

In a frontal digital image of a face it is not easy to see any 3-D structure. However, the eyes form prominent 2-D features under uniform or smoothly varying illumination. In almost all cases the eyes come in pairs and each eye, if it is open, can be described as having a white sclera and a dark iris. These facts provide a basis for detecting the eyes, once the face has been detected.

In this paper we perform face region detection using flesh-tone signatures in color images, as described in Section 3. We estimate the approximate scale of the face from the “face blob” thus obtained. Using this scale estimate we define filtering methods for detecting the eyes, as described in Section 4. Successful eye detection also serves to verify the correctness of the face region detection.

2. Literature review

This section reviews pertinent literature on both face detection (Section 2.1) and facial feature detection, with emphasis on the eyes (Section 2.2). Literature dealing with the holistic analysis or recognition of faces is generally not covered here; for a review the reader is referred to Ref. [1].

2.1. Face detection

We first review face detection in gray-level images; face detection in color images will then be reviewed.
One of the earliest papers that reported the presence or absence of a face in an image was Ref. [2]. An edge map extracted from the image is matched to an oval template whose position and size can vary. At positions where potential matches are reported, the face hypothesis is confirmed by checking for edges at the expected positions of the eyes, mouth, etc. The performance of this technique was sensitive to the illumination direction.

Kelly [3] introduced a top-down image analysis approach which he called “planning” for automatically extracting the head and body outlines from an image and subsequently locating the eyes, nose, and mouth. His head extraction algorithm worked as follows: Reduced-size versions of the original image, obtained by local averaging and sampling, are first searched for edges that may form the outline of a head. The extracted edge locations are then projected back to the original image, and a fine search is performed locally for edges that belong to the head outline. Heuristics are used to connect these edges. Once a connected head outline is obtained, the expected locations of the eyes, nose and mouth are examined to locate these features; heuristics are again employed in this search process.

Govindaraju et al. [4] define a computational model for locating the face in a cluttered image. Their technique utilizes a deformable template which is slightly different from that of Yuille et al. [5] (see Section 2.2). The template is composed of three segments that are defined by the major curvature discontinuities of the head outline. These three segments correspond to the right side, the left side and the hairline. Each of these segments is assigned a four-tuple consisting of the arc length of the curve, the chord in vector form, the area enclosed between the curve and the chord, and the centroid of this area. If all three of the segments are present in particular orientations, a face is detected. The center of the three segments gives the location of the center of the face. The templates are allowed to translate, scale and rotate according to spring-based models, and a cost function is used to evaluate hypothesized face candidates. Their method was tested on about 10 images; they claim to have never missed a face, but they also got false alarms.

Craw et al. [6] describe a method for extracting the head area from an image, subject to constraints on the location of the head in the image. Resolutions of \(8 \times 8\), \(16 \times 16, 32 \times 32, 64 \times 64\) and \(128 \times 128\) (full scale) are used in a multiresolution scheme. At the lowest resolution a template of the head outline is constructed. Edge magnitudes and directions are calculated from the gray-level image using a Sobel mask, and a line follower is used to connect the outline of the head. After the head outline has been located a search for lower-level features such as eyes, eyebrows, and lips is conducted, guided by the location of the head outline, using a similar line following method; but this search was not always successful.

Another method of finding the face in an image is described by Burt [7]. It utilizes a coarse-to-fine template matching approach to locate the head. To illustrate the usefulness of the method, it was applied in a “smart transmission” system which could locate and track the head in a video sequence and use the head location information in an object-based compression algorithm.

Craw et al. [8] describe a system for detecting and measuring facial features. Their work was motivated in part by automated indexing of police mug-shots. They endeavored to locate 40 feature points in a gray-scale image; these feature points were chosen according to the system defined by Shepherd [9], which was also used as a criterion of judgment. Their system too used a hierarchical coarse-to-fine search, and a template defined using the principle of polygonal random transformation of Grenander et al. [10]. The approximate location, scale and orientation of the head are obtained by iterative deformation of the template by random scaling, translation and rotation. Constraints are imposed to ensure that these transformations do not lead to results that have no resemblance to the human head. Optimization is achieved by simulated annealing [11]. After a rough idea of the location of the head is obtained, refinement is done by transforming individual sides of the polygon. The authors claimed successful segmentation of the head in all 50 images that were tested. In 43 of these images a complete outline of the head was obtained; in the remaining seven they failed to find the chin. The detailed template of the face included eyes, nose, mouth, etc.; in all, 1462 possible feature points were searched for. The authors claimed to have identified 1292 of these feature points. They attribute the 6% incorrect identification rate to be due to the presence of beards and mustaches, which caused mistakes in locating the chin and the mouth. Because of its use of optimization and random transformation, their system was computationally intensive.

In earlier work by the author [12] the face is segmented from a moderately cluttered background, using an approach that makes use of both the intensity image of the face and the edge image found using the Canny edge finder [13]. Pre-processing tasks include locating intersection points of edges (occlusions of objects), assigning labels to contiguous edge segments, and linking most likely similar edge segments at intersection points. The face is approximated by an ellipse. An accuracy of above 80% was obtained when the process was applied to a set of 48 cluttered images.

Yow and Cipolla [14] describe a feature-based face detection system. They model the face using horizontal lines indicating the mouth, end of nose, eyes, and eyebrows. To compensate for occlusion or partial responses, the face model is subdivided into four components each comprising a different subset of the original model. This subset is further divided into five groups that are
perceptually paired into horizontal pairs and vertical pairs. Features are detected using an edge detector and a line linker. The results are grouped according to rules that assign a probability of each sublevel linking to the next higher level. A region in the image is selected as a face if this probability is greater than a threshold. Experiments were conducted on a set of 110 images of faces at different scales, orientations and viewpoints. An 85% success rate was achieved. They reported that although their algorithm was insensitive to scale to some extent, it was not completely scale-invariant.

An information-based discrimination method for face detection is used in Ref. [15]. The Kullback relative information was used to discriminate between the face and non-face sets. The training set consisted of 11 × 11-pixel face images taken from the FERET database. The pre-processing phase involved only histogram equalization. Non-face examples were constructed by using general scene images. The system was tested on the same database as that of Ref. [16]. The authors reported no improvement in the detection and false alarm rate but mention that the algorithm is an order of magnitude faster than those in Refs. [16,17].

In Ref. [18] a support vector machine (SVM) is trained for face detection. SVM training is computationally quite expensive, but it results in finding the optimum boundary between different classes; for a detailed account of the SVM the reader is referred to Ref. [19]. Pre-processing was identical to that in Ref. [17]. Detection was carried out using an exhaustive hierarchical multi-scale filtering of the input image with the SVM classifier. Experiments were conducted on two sets of images and a comparison of SVM with the system of Sung and Poggio [17] was performed. Results showed an improvement over Ref. [17] for the first set, but no improvement for the second set.

Sung and Poggio [17] used an example-based learning approach. The training set consisted of a 19 × 19-pixel face image. The pre-processing was identical to that in Ref. [16]. The face set is modeled by dividing it into six Gaussian clusters, each with a mean and a covariance matrix, by elliptical $k$-means clustering. For the non-face class they determined a negative region in face space by “bootstrapping”; the misclassified faces are also modeled by a set of six clusters. The normalized Mahalanobis distance is used as the metric on a subset of 75 eigenvectors. Two distance measures are defined for each of the six face and six non-face clusters; one is the projective distance from the cluster mean, while the other is the orthogonal distance. For each test image, 12 pairs of distance measures are determined and these 12 pairs are fed to a set of three classifiers. Experiments were performed on two datasets, with results of 96.7 and 84.6% success reported using a single-layer perceptron as the classifier.

In Ref. [16] a neural network-based upright face detection system is outlined. A 20 × 20-pixel face was used to train the network. Pre-processing steps included using an oval mask to eliminate the corners, a brightness plane fit to remove brightness variations, and finally histogram equalization. Training is done using multiple face images obtained by minor variations in scale and rotation of an original and mirrored set of face images. For the non-face set, general scenes are initially used. During the training phase a bootstrap technique is used to augment the non-face region with misclassified faces. In the testing phase a brute force hierarchical pyramid scheme is used to check for faces at every location and every scale. After initial filtering with the face filter, the results are post-processed using another neural network. Experiments were conducted on two different datasets. Test Set 1 (consisting of 130 images with 507 frontal faces) yielded a performance of 92.5%. Test Set 2 was the FERET database, which yielded a 99.5% detection rate.

Ref. [20] is an extension to Ref. [16]. By augmenting the algorithm with a router network, which determines the angle of rotation, the authors were successful in detecting frontal face images in various rotated orientation. The new system was able to detect 76% of the faces in two test sets, with very few false alarms.

A common thread in most of the face detection work described up till now has been the fact that scale information cannot be obtained from a grayscale image; therefore, a brute force search has to be performed at every location at all possible scales to detect faces. Color images provide the needed ingredient to make initial segmentation and scale selection for faces possible. The remaining part of this section reviews work done using color images for segmentation and detection of faces.

In Ref. [21] a face detection and tracking method is outlined using normalized color information. The flesh region of the face is extracted and its color distribution is determined. Normalized red and green values are used to compensate for brightness. Along with this color information, motion is used to determine an area in the image that is flesh-colored and is moving. Combining these two information sources a head tracking system was developed using a neural network. Testing was done on two sequences of 1860 and 460 frames, yielding 96 and 100% accuracy, respectively.

Ref. [22] describes a lip and face real-time tracker using the same color distribution technique as in Ref. [21]. For tracking, Kalman filtering was used. A hidden Markov model was used to classify head motion as well as facial expression. A near-perfect detection rate was reported.

In Ref. [23] a color facial region template-based approach is used to detect faces in color images. The face is divided into six rectangular regions; the forehead, two eye regions, two cheek regions, and the mouth region. A priori constraints on chromatics and smoothness associated with each of these regions are used for their
selection by assigning to each region an invariant measure. A multiresolution search is performed on the image using a fixed-size template. Although the template and its regions do not give accurate feature detections, the face region is claimed to be robustly detected, even under varying pose.

2.2. Facial feature detection

Nixon [24] used the Hough transform for eye detection. The eye was modeled by a circle for the iris and a “tailored” ellipse for the sclera boundary. The tailoring was necessary because of the pointed nature of the corners of the eyes. The Sobel gradient operator was applied to the initial image. Thresholding the resulting gradient image produced an edge image. Further constraints were placed on the positions of the eyes in the image (the search was performed on the upper half of the image). Testing was done on a set of six images taken under similar illumination conditions. Results of the testing indicated a small difference between the experimental position and the measured position of the iris center.

Yuille et al. [5] extracted facial features using deformable templates. These templates are allowed to translate, rotate and deform to fit the best representation of their shape present in the image. Preprocessing is done to detect the peaks and valleys in the initial intensity image. The template for the eye has 11 parameters consisting of the upper and lower arcs of the eye; the circle of the iris; the center points; and the angle of inclination of the eye. This template is fit to the image in an energy minimization sense. Energy functions of valley potential, edge potential, image potential, peak potential and internal potential are determined. Coefficients are selected for each potential and an update rule is employed to determine the best parameter set. In the experiments it was found that the starting location of the template is critical for determining the exact location of the eye. When the template was started above the eyebrow, the algorithm failed to distinguish between the eye and the eyebrow. Another drawback to this approach is its computational complexity. Generally speaking, template-based approaches to feature extraction are more logical. The problem lies in the description of the templates. Whenever analytical approximations are made to the image, the system has to be tolerant to certain discrepancies between the template and the actual image. This tolerance tends to average out the differences that make individual faces unique.

A statistically motivated approach to detecting and recognizing the human eye in an intensity image with the constraint that the face is in a frontal pose is described by Hallinan [25]. A template-based approach to detecting the eye in an image is presented. The template is depicted as having two regions of uniform intensity. The first is the iris region and the other is the white region of the eye. The approach constructs an “archetypal” eye and models various distributions as variations of it. For the “ideal” eye a uniform intensity is chosen for both the iris and white. In an actual eye certain discrepancies from the ideal are found which hamper the uniform intensity choice. These discrepancies can be modeled as “noise” components added to the ideal image. For instance, the white region might have speckle (spot) points depending on scale, lighting direction, etc. Likewise, the iris can have “white” spots within it. The author uses an $\alpha$-trimmed distribution for both the iris and the white. A “blob” detection system is developed to locate the intensity valley caused by the iris enclosed by the white. Using $\alpha$-trimmed means and variances and a parameter set for the template of the blob, a cost function is determined for valley detection. A deformable human eye template is constructed around the valley detection scheme. The search for candidates uses a coarse-to-fine approach. Minimization is achieved using the steepest descent method. After locating the candidate a goodness-of-fit criterion is used for verification purposes. The inputs used in the experiments were frontal face intensity images. In all, three sets of data were used. One consisted of 25 images used as a testing set, another had 107 positive eyes, and the third consisted of images with most probably erroneous locations which could be chosen as candidate templates. For locating the valleys the author reported as many as 60 false alarms for the first data set, 30 for the second and 110 for the third. An increase in hit rate was reported when using the $\alpha$-trimmed distribution. The overall best hit rate reported was 80%.

Reisfeld and Yeshurun [26] use a generalized symmetry operator for finding the eyes and mouth in a face. Their motivation stems from the almost symmetric nature of the face about a vertical line through the nose. Subsequent symmetries lie within features such as the eyes, nose and mouth. The symmetry operator locates points in the image corresponding to high values of a symmetry measure discussed in detail in Ref. [26]. They indicate their procedure’s superiority over other correlation-based schemes like that of Baron [27] in the sense that their scheme is independent of scale or orientation. However, since no a priori knowledge of face location is used, the search for symmetry points is computationally intensive. The authors mention a success rate of 95% on their face image database, with the constraint that the face occupies between 15 and 60% of the image.

In an extension to their “eigenface” method, Ref. [28] uses a modular “eigenspace” for detecting features in the face. They construct principal component projective spaces for the eyes, nose and mouth and call them “eigeneyes”, “eigennose” and “eigennemouth”, respectively. Detection of these features involves a hierarchical search in scale and space for a region in the image whose projection matches the signature of the feature in question. Detection rates of 94, 80, and 56% were reported for
the eyes, nose and mouth, respectively, on a large dataset of 7562 images.

Gabor [29] filters are used to locate features of the eyes. These features include the corners of the eyes and the iris. In a model-based approach, steerable Gabor filters are used to locate and track the corners of the eyes as well as finding the iris. A sequential search is employed by first finding a particular edge, e.g., the left corner of the iris, then using steerable filters to track the edge of the iris. A similar method is used to find the corners of the eyes. Experiments were performed on 100 high resolution face images with diffuse illumination. The authors reported an overall iris detection error rate of 2.35%. On the remaining images the outer eye corners were found successfully while a 91.6% success rate was reported for the inner eye corners.

Most of the literature reviewed here involved brute-force, hierarchical search for the detection of faces in gray-scale images. Color images, along with flesh-tone information, provide a means of estimating the size of the face. We will augment the face detection method described in Ref. [21] and use a filter-based approach to locate the eyes for face detection verification as well as facial feature detection. More specifically, we will describe two filter-based approaches, one using Gabor wavelet-based linear filters for eye detection, and the other using non-linear filters for eye corner detection. As we will see, the linear method has good detection performance, but gives many false alarms; this led us to develop the non-linear method.

Our methods will be tested on the Aberdeen, UK face image database, as well as on video frames collected in our laboratory. The methods described in the literature have been tested on different datasets; this makes it difficult to compare the performance of different methods. At the algorithm level, our approach has significantly less complexity and should perform better than those in the literature.

3. Face detection

In this section we describe in detail the method we have used to detect and delineate the face region from the background. Our method is closely related to that described in Ref. [21].

3.1. Color segmentation

Let \( \mathcal{S} \) represent a color image with RGB values. This image is normalized to \( r = R/(R + G + B) \) and \( g = G/(R + G + B) \) which we call \( \mathcal{S}_n \) [21]. We determine the flesh tone color space by manually selecting flesh regions in color images and histogramming their \( r \) and \( g \) values (both rescaled to 100), giving \( H_i(r, g) \), where subscript \( i \) is the image number. The average histogram is \( H = 1/n \sum_{i=1}^{n} H_i(r, g) \); it is shown as a 2D scatter plot in Fig. 1. In Ref. [21] motion information was used to decide if a region of interest was a face or not. We do not have motion information, as our database consists of single images. We assume that the face is the most prominent flesh-tone region in the image, and we verify our assumption with the aid of detecting internal features such as eyes.

Many methods of analyzing histograms can be found in the literature [30]. We can model the scatter plot by either a single or a mixture of Gaussian distributions. By examining the scatter plot of the average histogram we found that a single 2D Gaussian was an adequate model. The fitted Gaussian had the following parameters:

\[
\begin{align*}
\mu &= \begin{bmatrix} 42.91 \\ 32.28 \end{bmatrix} \\
\Sigma &= \begin{bmatrix} 19.28 & -5.45 \\ -5.45 & 8.55 \end{bmatrix}
\end{align*}
\]

Let \( V \) be the two-dimensional vector of \( r \) and \( g \), with \( \mu_r \) and \( \mu_g \) the respective means and \( \Sigma \) representing the covariance. The transformation

\[
\hat{V} = AV + E[V],
\]

where

\[
A = B^{-1/2}
\]
(with $B$ and $\Gamma$ being the eigenvectors and eigenvalues of $\Sigma$, respectively) will give uncorrelated $r$ and $g$ values. This procedure transforms the $r$ and $g$ values to the axes of the Gaussian cluster. We found that the resulting rotation was negligible and a simple rectangular bounding box around the cluster was sufficient for our experiments.

This analysis gives us $D_{\text{face}}$, which is the flesh tone color space. $D_{\text{face}}$ is used to segment the image into regions of interest and background. This binary image $S_{\text{roi}} = S_{\mu}(r, g) \in D_{\text{face}}$ is further processed by connected component labeling, resulting in the formation of blobs of flesh tone.

Seventy-two color images were taken from the Aberdeen, UK database. These images were collected under slightly varying illumination and scale. All of these images have a frontal view of the face. The original images can be seen in Figs. 2 and 3 and the results of flesh tone filtering are shown in Figs. 4 and 5. A bounding rectangle is used to enclose the face blob. As can be seen from the results of the flesh tone filtering, illumination changes, especially highlights, tend to affect the color of the flesh region. This is apparent from the large number of “holes” in the face blob. Although there exist features (regions) in the face blob which should deviate from the flesh tone, for instance, the eyes, nostrils and in some cases the lips, we cannot rely on this being the case in general.

It should be noted that the normalized color space for flesh tone is dependent on the color of the illumination source. This creates a need to initialize the flesh tone for a particular acquisition scenario. Once initialization has been done, the flesh tone is relatively immune to individual variations in flesh tone, even across different races [21].

As a secondary database we used a sequence of 21 (nonconsecutive) video frames acquired in our laboratory. This sequence was collected in a semi-controlled
Fig. 3. Images 37–72 of the Aberdeen database.

environment. The background is more cluttered than the one for the Aberdeen images. Histogram analysis gave us the following parameter values:

$$\mu = \begin{bmatrix} 52.66 \\ 29.99 \end{bmatrix}$$

$$\Sigma = \begin{pmatrix} 68.94 & -30.89 \\ -30.89 & 18.11 \end{pmatrix}.$$

As in the case of the Aberdeen images we used a simple rectangular bounding box to specify the flesh tone region. It should be noted that the parameter values for this database are different from those for the Aberdeen database.

A sample image from the frame sequence and the result of flesh tone filtering are shown in Figs. 6(a) and (b), respectively. The region of interest in Fig. 6 and its connected components of flesh tone are shown in Fig. 7.

3.2. Scale selection

The literature reviewed in Section 2.1 indicates the difficulty of determining the scale of the face in an intensity image without brute-force search. This task is somewhat easier when color images are used and pre-screening is done using flesh-tone color information, but even in a color segmentation the exact scale (or size) of the face is not directly evident. This is clear from the color segmentation results in Figs. 4 and 5. We have used a rectangular bounding box to determine the face area. This area may contain part of the neck as well as some hair (if the hair color matches the flesh color).

Our goal is to determine the size of the eyes (orbits). If the mandible size (Fig. 8) is known then it is very easy to
determine the orbit size using the reference values in Ref. [31]. The face blob has the face region as a subset, since the neck and sometimes the hair (or lack thereof) also show up as flesh tone. There is no easy way to determine the exact size of the mandible. However, we can approximate the mandible size by the square root of the area of the face blob. Using reference values from Ref. [31] we determine the mean and variance of the ratio between the blob size and the orbit size. This ratio can be used to estimate the size of the eye. We have experimentally found this estimate to correctly determine the size of the eye in the majority of our images. Minor exceptions occur when long hair matching the flesh tone is present. Examples of this can be seen in Figs. 4 and 5.

In summary, after the flesh-tone colored blobs are segmented, an estimate of the orbit size is computed using anthropometric ratios. We will now describe two methods of detecting the eyes using this information.

4. Eye detection

In this section we describe two methods of detecting the eyes in the face image. The first method, which is described in greater detail in Ref. [32], uses linear filters to detect the eyes in the gray level image of the face, as described in Section 4.1. This method gave good detection results, but it also gave many false alarms when applied to our face databases. We therefore developed a second method in which nonlinear filters were used to detect the corners of the eyes, as described in Section 4.2. This method also gave good detection results, with no false alarms.

4.1. Eye detection by linear filtering

Our linear eye detection filter is constructed out of oriented Gabor wavelets which form an approximation
Fig. 5. Delineated flesh regions for the images of Fig. 3.

Fig. 6. (a) Sample image from our sequence. (b) Result of flesh-tone filtering.

to the eye. Gabor wavelets have been used in a variety of feature extraction and face recognition algorithms; see Refs. [29,33,34]. One of the most attractive reasons for using Gabor wavelets is their energy localization property in both the space and frequency domains. In Ref. [29] steerable Gabor wavelets were used to track the left and right corners of the eyes.

Fig. 7. Region of interest in Fig. 6 and its connected components.
The general shape of the eye is elongated with the long axis horizontal. Fig. 9 is a sketch of an eye as a 2D object. The size of the eye is not constant even if the distance of the face from the camera is fixed. An exact matched filtering technique will run into problems due to these variations. Fig. 9 suggests finding the eyes by detecting the right and left sclera regions as well as the iris region.

There is a disparity in intensity values between the flesh region of the face and the sclera. We use Gabor wavelets to detect the edges of the eye away from the corners of the eye. At an appropriate scale and orientation, filtering with a single Gabor wavelet gives a prominent response to the corresponding pattern of intensity variations in the image. If we look at the real and imaginary parts of the wavelet (Figs. 10(a) and (b)) we notice that the real part is akin to a spot detector and the imaginary part resembles a step edge detector. Our approach utilizes the ability of Gabor wavelets to find oriented edges and applies it to detect the edges of the eye's sclera.

Each eye is modeled by four Gabor wavelets in a prescribed formation (Fig. 11). Although most of the work done on Gabor wavelet filtering for detection of features involves the real part [33], we have found that the imaginary part is ideally suited for detecting the boundary between the flesh region of the face and the sclera. The imaginary part of the Gabor wavelet is asymmetrical in the direction of the sinusoid propagation. To capture the sclera, we orient the positive parts of the wavelets toward the inside of the eye. This yields a high response if all four wavelet filters find the eye edges.

Fig. 11 shows the orientations of the Gabor wavelets that maximize the detection of all four intensity transition regions of the eye. In addition to these four filters we also use a Gaussian filter to detect the dark circle of the iris. Fig. 12 is a graphical rendition of the combined wavelet and Gaussian filter.
A tradeoff is needed to optimize the selectivity of the filter and its immunity to illumination variations. This led us to decompose the filter into three components which we call the left opposing wavelets (Fig. 13(a)), the right opposing wavelets (Fig. 13(b)), and the central Gaussian for iris detection (Fig. 13(c)).

The image is convolved with these three filters separately. To select a feature point, threshold values are selected at the mean plus 1.5 times the standard deviation for each of the three convolutions. A point in the image is classified as a feature if all three convolutions at that point exceed their thresholds.

The scale parameter is set to be 55% of the size of the orbit that is determined by the scale selection algorithm (Section 3.2). The horizontal and vertical displacements are kept at a ratio of 3:1 in favor of the horizontal direction. The angular parameters are selected as $\pm 30^\circ$ for the left and right opposing wavelets, respectively.

To eliminate false alarms and validate correct detections, we employ two stages of post-processing. The first stage acts as a pre-screener using one-dimensional intensity variations to verify the eye hypothesis, and the second stage uses scale information to pair these screened responses. We first describe the verification stage in some detail, because it is also used in our nonlinear filtering method of eye detection; we then briefly describe the pairing stage.

The rationale behind the verification stage is to obtain independent confirmation for the detection of the eye. Instead of performing two-dimensional spatial pattern matching we project a portion of the image onto one dimension and analyze the projection to determine if the image has “eyeness” properties. “Eyeness” can be described by the light to dark to light intensity variation from the left part of the sclera to their ist to the right part of the sclera.

A compact light source tends to produce differing intensity values across the image. This was encountered in the flesh tone classification method used for face detection as well, where it produced holes in the face blob, as seen previously. A compact light source creates a highlight on the iris. The position of this highlight is dependent on (a) the direction of the light source and (b) the gaze direction of the eye. The eye is a spherical object and will reflect the light source at the point where the normal to the eye bisects the angle between the light source and the camera. This highlight manifests itself as a high intensity value on the iris. A simple light to dark to light
approach to describing “eyeness” will fail due to this false bright intensity value at a point where the dark iris should be present. This highlight could be modeled as spot or specular noise, and the first thought might be to use a smoothing filter to reduce its effect. However, since the spatial resolution of the eye is very small, any smoothing operation will result in corrupting the natural variations of the sclera and iris that we are trying to detect.

At every point where an eye has been hypothesized we use the size of the estimated orbit to define a region in the image centered at that point. Let this region be $e$ and its size be $m \times n$ (in rows and columns). $e$ is projected onto four vectors of $n$ elements by taking the minimum, maximum, average and median values of each column (see below for the rationale for these four projected spaces). The vectors are thus given by

$$t_{\text{min}}(j) = \min(e(i,j), i = 1, \ldots, m),$$

$$t_{\text{max}}(j) = \max(e(i,j), i = 1, \ldots, m),$$

$$t_{\text{avg}}(j) = \text{mean}(e(i,j), i = 1, \ldots, m),$$

$$t_{\text{median}}(j) = \text{median}(e(i,j), i = 1, \ldots, m).$$

Examples of these projections are shown in Fig. 14.

The original “eyeness” criterion would have given rise to a “U” shaped pattern, with the left sclera forming one peak, the right sclera forming the other peak, and the iris forming a trough between these two peaks. When a highlight is present in the iris, the “U” turns into a “W” (see Fig. 15).

We found that the highlight on the iris sometimes creates a problem for $t_{\text{max}}$. Slightly offset detection of the eye sometimes causes the dark area of the eyelashes to be included in $e$, causing $t_{\text{min}}$ to give erroneous responses. Using $t_{\text{avg}}$ sometimes fails when the eye is rolled to the side and $t_{\text{median}}$ sometimes gives erroneous values when the eye detection is not on the central horizontal axis. To compensate for these individual shortcomings of $t_{\text{min}}, t_{\text{max}}, t_{\text{avg}}$, and $t_{\text{median}}$, we used all of the projections and constructed a consensus-based decision criterion.

Each of the projected vectors $t_i$: $i \in \{\text{min, max, avg, and median}\}$ is divided into two equal halves $t_i$ representing the left and right halves of the eye. We locate the maximum point in each of the halves; let $x_{i_{\text{max}}} = \max(t_i)$ be the maximum for the left half and $x_{i_{\text{max}}} = \max(t_i)$ the maximum for the right half. For the iris we determine the minimum point between $x_{i_{\text{max}}}$ and $x_{i_{\text{max}}}$ and call it $x_{i_{\text{min}}}$. Consensus is obtained by using a voting strategy. We allow a tolerance of one position and the four projected vectors are equally weighted. Voting is done by the left half and right half maxima of every projected vector. A decision is reached if any point on the vector has more than two candidates voting for it and the ratio between the mean value of these two points in the maximum projection and the minimum point in the minimum projection is above a set threshold.

![Fig. 14.](image-url)
Finally, we briefly describe the pairing stage. The eyes in a frontal image of the face almost always come in pairs. Since we have the scale information for the face, we can impose spatial distance and orientation conditions on the responses that have been passed by the verification stage. Pairing is achieved by looking for responses that are within a distance tolerance of $d_{\text{scale}}$ and have an angular deviation of at most $\pm \alpha$ degrees from the horizontal; $\alpha$ was set at 7%.

Tables 1 and 2 show the results of applying the linear filter, followed by the verification and pairing post-processing stages, to the Aberdeen database. These tables show the image number in the first column, the number of responses after applying the linear filter in the second column, the number of points which satisfy the “eyeness” criterion in the third column, and the final number of pairs that are selected as eye candidates in the fourth column. We see that the linear filtering approach provides adequate eye detection, but its false alarm rate is rather high.

An examination of the results leads to the conclusion that this approach is best suited for uniform illumination situations where the eyes are wide open, closely resembling the figure of the eye as seen in Fig. 9. A reason for missing the detection of an eye is that, due to varying illumination across the face, thresholding (based on the statistics of the filter responses) may produce good detections on the side of the face closest to the illumination source and poor detections on the side away from the source. In such a situation only one eye will be detected, and since we are enforcing a pair of eyes condition for the final selection, this results in a missed detection. Lowering the threshold decreases the missed detection rate but increases the number of false alarms. Another factor responsible for false alarms is detections that are close to each other.

The linear filtering method was also applied to our frame sequence. The offset $m$ was again kept at 1.5 with the scale parameter $s$ set at 65% of the orbit size determined by the scale selection procedure. $d_{\text{scale}}$ is again set

![Eye-size image around feature point](image)

![Average value projection](image)

Fig. 15. Average value projection (b) shows “W” caused by the highlight on the iris (a).

<table>
<thead>
<tr>
<th>Image no.</th>
<th>Initial centers</th>
<th>Verified centers</th>
<th>Paired centers</th>
</tr>
</thead>
<tbody>
<tr>
<td>im1</td>
<td>14</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>im2</td>
<td>30</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>im3</td>
<td>28</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>im4</td>
<td>40</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>im5</td>
<td>24</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>im6</td>
<td>55</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>im7</td>
<td>52</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>im8</td>
<td>53</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>im9</td>
<td>44</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>im10</td>
<td>70</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>im11</td>
<td>54</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>im12</td>
<td>61</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>im13</td>
<td>57</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>im14</td>
<td>50</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>im15</td>
<td>70</td>
<td>13</td>
<td>6</td>
</tr>
<tr>
<td>im16</td>
<td>61</td>
<td>11</td>
<td>7</td>
</tr>
<tr>
<td>im17</td>
<td>80</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>im18</td>
<td>58</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>im19</td>
<td>48</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>im20</td>
<td>32</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>im21</td>
<td>37</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>im22</td>
<td>41</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>im23</td>
<td>148</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>im24</td>
<td>158</td>
<td>13</td>
<td>2</td>
</tr>
<tr>
<td>im25</td>
<td>48</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>im26</td>
<td>67</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>im27</td>
<td>70</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>im28</td>
<td>75</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>im29</td>
<td>79</td>
<td>14</td>
<td>2</td>
</tr>
<tr>
<td>im30</td>
<td>95</td>
<td>16</td>
<td>5</td>
</tr>
<tr>
<td>im31</td>
<td>64</td>
<td>11</td>
<td>2</td>
</tr>
<tr>
<td>im32</td>
<td>51</td>
<td>13</td>
<td>6</td>
</tr>
<tr>
<td>im33</td>
<td>43</td>
<td>12</td>
<td>8</td>
</tr>
<tr>
<td>im34</td>
<td>55</td>
<td>16</td>
<td>8</td>
</tr>
<tr>
<td>im35</td>
<td>106</td>
<td>14</td>
<td>3</td>
</tr>
<tr>
<td>im36</td>
<td>117</td>
<td>17</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 1

Intermediate and final results of linear filtering for images 1–36
by the scale selection and the rotation tolerance $\alpha = 7^\circ$. Table 3 shows the results of applying the various stages to the 21 frames of the sequence. The eyes in these images closely resemble the ideal eye depicted in Fig. 9. This leads to better detection rates, as seen from the results. False alarms, however, are still present.

Tables 1–3 show that our linear filtering method, followed by postprocessing, had an 80% detection rate on the Aberdeen database and a 95% detection rate on the frame sequence, but is suffered from many false alarms. The high false alarm rate of the linear filtering method led us to develop a non-linear approach based on eye corner detection, as described in Section 4.2.

4.2. Eye detection by nonlinear filtering

The linear filtering-based method described in Section 4.1 was applied to gray-level images; color was used only to detect the face and determine the scale for the filter. Color images provide additional information which we can also use in detecting the eyes. This information is present in the form of semi-distinct normalized color distributions of the sclera and the flesh. The semi-distinctness can be seen in Fig. 16 and its reason is explained below.

The eyes can be thought of as islands in a sea of flesh tone. They have a distinct spatial pattern at each corner. This pattern resembles a wedge of sclera color surrounded by flesh tone. It should be noted here that we are ignoring the iris color, because it has a large degree of variability across individuals.

In this section we define non-linear filters to detect the left and right corners of the eyes in a color image. These filters are constructed to take into account the variability of squint in the eye, a factor we were unable to account for in our linear filtering method. The design and implementation of the filters is described in Section 4.2.1.

Since our filters are designed to detect left and right corners separately, their responses provide rich information for detecting the eyes. We will describe the use of this information in Section 4.2.2. Experimental results of testing the non-linear filtering-based method on the same images that were used in Section 4.1 are given in Section 4.2.3.
4.2.1. Filter design and implementation

Our filters are applied to color images and make use of information about the normalized color distribution of the flesh tone in the face [21]. In the same way that the face region’s color distribution is determined, we also determine the color distribution of the sclera region. Let the normalized color distribution for the face be $D_{\text{face}}$ (Fig. 16(a)) and that for the sclera be $D_{\text{eye}}$ (Fig. 16(b)) (these normalized distributions are empirically determined from $r$ and $g$ as in Section 3.1). The sclera region can contain some flesh-tone-like characteristics due to the presence of blood vessels in it. In addition to this, highlights on the face tend to push the flesh-tone toward the eye-tone. These factors cause the color distribution of the sclera region to overlap that of the flesh-tone region, as can be seen in Fig. 16. Discrimination between the flesh region and the sclera region has to take this overlap into account. This is accomplished by experimentally determining a boundary in the normalized ($r$, $g$) space.

Given the flesh-tone and eye-tone distributions in the normalized color space, we can construct non-linear filters to detect the left and right corners of the eye. We know that the eye corners are surrounded by flesh regions, and in turn surround the sclera of the eye. Our filters are tuned to look for positions in the image which have this characteristic.

At every location in the face region we apply left and right eye corner filters. These “wedge filters” are made up of three wedge-shaped parts as shown in Fig. 17. The largest wedge looks for flesh tone and the smallest wedge looks for sclera tone. There is a “don’t care” region between the sclera wedge and the flesh wedge; this allows for possible squinting of the eye. A corner, either right or left, is detected if the average value of the pixels in the flesh wedge satisfies $D_{\text{face}}$ and the average value of the pixels in the sclera wedge satisfies $D_{\text{eye}}$. (We also tested approaches not based on the average, e.g., determining the numbers of flesh-colored and sclera-colored pixels in the wedge. We found that this approach was much less tolerant to noise than the averaging approach.)

A lot of work has been done on corner detection in gray-scale images [35–38]. Some methods make use of variable wedge-shaped neighborhoods [39,40]. Our method differs because we use color images and we know the approximate orientation and scale of the corners that we are interested in. We also use “don’t care” regions,
which are a standard idea in edge detection (for example
the zeros in the Sobel operator). Non-linear feature de-
tectors, especially for lines, are discussed extensively in
Ref. [41]. The discussion in Ref. [41] provides a frame-
work for defining such detectors for other types of fea-
tures, such as corners. We have also made use of the
 distinction in Ref. [41] between non-linear and semi-
linear filters.

Our corner filters, like their linear counterparts in
Section 4.1, are scale-dependent. The filters shown in Fig.
17 have a radius of 15 pixels. The inner wedge has an
angular size of 20°, and the outer wedge has an angular
size of 70°. The don’t care region thus has two parts of
25° each. In rescaling to take into account the size of the
face, the only change needed is in the radius of the filter.

The extracted face subimage (Section 3.1) is treated as
the input to the filtering process. Scale determination
(Section 3.2) allows us to select the radius of the filters.
Although scale has some effect on the performance of the
filters, it is not as critical as in the case of the linear filter.
For this reason we determined the average scale of our
data set and used a single scale in the filtering process.

Let the filter radius be \( r \) pixels. Let \( \mathcal{W}_o \) and \( \mathcal{W}_i \) be
the outer and inner left corner wedges, and let \( \mathcal{W}_a \) and
\( \mathcal{W}_b \) be the corresponding right corner wedges. It should
be noted that these are binary filters. The filter response
is an “ANDING” of an image region corresponding to
the size of the wedge with the wedge, resulting in selection
of all the image values at locations where the wedge has
value 1. This process is done for all four wedges at every
possible image location.

Let \( \mathcal{F}_\mathcal{L}_o \) and \( \mathcal{F}_\mathcal{L}_i \) be the normalized \( r \) responses of
filtering with the outer and inner left corner wedges, and
let \( \mathcal{F}_\mathcal{R}_o \) and \( \mathcal{F}_\mathcal{R}_i \) be the normalized \( r \) responses using
the outer and inner right corner wedges. Then

\[
\begin{align*}
\mathcal{F}_\mathcal{L}_o(i,j) &= \text{mean} (\mathcal{I}_g(i-n:i+n, j-n:j+n) \land \mathcal{W}_o), \\
\mathcal{F}_\mathcal{L}_i(i,j) &= \text{mean} (\mathcal{I}_g(i-n:i+n, j-n:j+n) \land \mathcal{W}_i), \\
\mathcal{F}_\mathcal{R}_o(i,j) &= \text{mean} (\mathcal{I}_g(i-n:i+n, j-n:j+n) \land \mathcal{W}_a), \\
\mathcal{F}_\mathcal{R}_i(i,j) &= \text{mean} (\mathcal{I}_g(i-n:i+n, j-n:j+n) \land \mathcal{W}_b),
\end{align*}
\]

where \( \mathcal{I}_g \) is the extracted image of normalized \( g \) image
values as determined in Section 3.1. Similarly, the responses for
the normalized \( g \) image are

\[
\begin{align*}
\mathcal{F}_\mathcal{L}_o(i,j) &= \text{mean} (\mathcal{I}_g(i-n:i+n, j-n:j+n) \land \mathcal{W}_o), \\
\mathcal{F}_\mathcal{L}_i(i,j) &= \text{mean} (\mathcal{I}_g(i-n:i+n, j-n:j+n) \land \mathcal{W}_i), \\
\mathcal{F}_\mathcal{R}_o(i,j) &= \text{mean} (\mathcal{I}_g(i-n:i+n, j-n:j+n) \land \mathcal{W}_a), \\
\mathcal{F}_\mathcal{R}_i(i,j) &= \text{mean} (\mathcal{I}_g(i-n:i+n, j-n:j+n) \land \mathcal{W}_b).
\end{align*}
\]

Eqs. (5) and (6) determine four pairs of values of every
possible pixel location in the image. To determine if an
image location qualifies as either a left corner or a right
corner we need to check its normalized color values
against the flesh tone and sclera tone values that we have
determined above. Let \( L \) indicate a left corner detection
image and \( R \) indicate a right corner detection image; then

\[
\begin{align*}
L(i,j) &= (\mathcal{F}_\mathcal{L}_o(i,j), \mathcal{F}_\mathcal{L}_i(i,j)) \in \mathcal{D}_\text{face} \\
&\land (\mathcal{F}_\mathcal{R}_o(i,j), \mathcal{F}_\mathcal{R}_i(i,j)) \in \mathcal{D}_\text{eye}, \\
R(i,j) &= (\mathcal{F}_\mathcal{R}_o(i,j), \mathcal{F}_\mathcal{R}_i(i,j)) \in \mathcal{D}_\text{face} \\
&\land (\mathcal{F}_\mathcal{L}_o(i,j), \mathcal{F}_\mathcal{L}_i(i,j)) \in \mathcal{D}_\text{eye}.
\end{align*}
\]

Using mean values in the filter application tends to
increase the number of responses. These responses tend
to form clusters. We use pruning operations to reduce the
effect of this clustering. The simplest operation would be
to perform connected component labeling and use the
center of mass of each cluster as the pruned response.
However, we are using wedge-shaped filters for both left
and right corner detection, and we know the approxim-
ate orientation of the eyes. This justifies the following
pruning heuristic: For the left corner clustered responses
we use the mean value in the vertical direction but the
minimum value in the horizontal direction. Likewise, for
the right corner clustered responses we use the mean
value in the vertical direction but the maximum value in
the horizontal direction.

Figs. 18(a) and (b) show the clustering of the left and
right corner responses, respectively. Figs. 18(c) and (d) are
the responses after the pruning operations described
above. We can see the effect of using these pruning
operations by the results shown in Fig. 18. The left corner
operation looks for the leftmost midpoint of the cluster,
thus effectively picking up the farthest left response.
Similarly, the right corner operation looks for the rightmost
midpoint of the cluster. Averaging the masked values
would introduce responses that are displaced away from
the actual corners, e.g., a left corner response would be
displaced to the right, into the sclera region. Our pruning
operations effectively negate this artifact.

Figs. 18(c) and (d) still show a lot of false responses.
This demonstrates the need to do further post-process-
ing. It should be noted that the responses are identified as
right or left responses, and we also have scale informa-
tion from the face detection and scale selection operation.
Combining this information gives us more flexibility than
we had in the linear filtering approach. In the next
section we will describe the post-processing that we per-
form after non-linear filtering, using the corner detection
results as a starting point.

4.2.2. Post-processing

The corner detection algorithm gives responses for left
corners and right corners. We also have scale information
about the face, specifically, an estimate of the orbit size. Using reference values we can easily find the inter-ocular distance from the estimates of the orbit size [31]. In this section we describe post-processing methods that make use of this geometric information.

Fig. 19 presents a block diagram of our post-processing procedure. We pair the left and right responses in three different ways. The first one (indicated by the LR channel in Fig. 19) pairs a left response with the corresponding right response using the orbit size as the pairing criterion. This pairing is termed LR pairing. We have found that this simple pairing alone does not effectively eliminate the false alarms. The second and third pairings are shown in the LL RR channel of Fig. 19. In this case we use the interocular distance as the pairing criterion. We pair left responses that satisfy this criterion and term the result LL, and we similarly pair right responses and term the result RR. We then use the “eyeness” verification procedure described in Section 4.1 to reduce the number of pairings. Finally, we combine the surviving LR, LL and RR pairs into final decisions. We will describe the all of these steps in detail in the subsections that follow.

4.2.2.1. Pairing responses. The first pairing that we do is for left and right corner responses that satisfy the orbit size criterion. We are looking for the left and right corners of an eye. Let the left corner responses be \( l(i), i = 1, \ldots, k \) where \( l(i) \) is the coordinate pair for the \( i \)th response and \( k \) is the total number of left responses. Similarly, \( r(j), i = 1, \ldots, m \) represents the \( i \)th right corner response and \( m \) is the total number of right corner responses. Let \( v_l \) be the \( k \times m \) matrix joining each left corner response to every right corner response. Let \( d_{l, r} \) be the \( k \times m \) distance matrix representing the \( L_2 \) distance between the left and right corner responses. Also, let \( \theta_{l, r} \) be the \( k \times m \) slope matrix representing the slope of the line connecting each left corner response with each right corner response. Then

\[
d_{l, r}(i, j) = \|v_l(i, j)\|, \quad \forall (i, j) \in \mathbb{N}^k \times m,
\]

\[
\theta_{l, r}(i, j) = \langle v_l(i, j) \rangle, \quad \forall (i, j) \in \mathbb{N}^k \times m.
\] (8)

Let the estimated orbit size range be \( \partial b_{\text{min}} \) to \( \partial b_{\text{max}} \). Since we are dealing with frontal upright faces, we do not anticipate significant rotation. For this reason we have a tight bound on the allowable slope value of the line connecting the two corners of the eye. Let this bound be \( \beta_{l, r} \). A pair of left and right corner responses is said to form an LR pair if the following conditions hold:

\[
d_{l, r}(i, j) \leq \partial b_{\text{min}},
\]

\[
|\theta_{l, r}(i, j)| \leq \beta_{l, r}.
\] (9)

Left corner response \( l(i) \) then forms an LR pair with right corner response \( r(j) \). Fig. 20 shows the resulting LR pairs formed from the left and right responses shown in Figs. 18(c) and (d).

The second type of pairing joins left (or right) eye corners that could be the left (or right) corners of two eyes. We denote a left/right pairing by LL and a right/right pairing by RR. Using the same notation for left and right responses we denote by \( v_l \) and \( v_r \) the \( k \times k \) and \( m \times m \) vectors joining each left corner with every other left corner and each right corner with every other
right corner, respectively, \(d_{ll}\) is the \(k \times k\) distance matrix for LL pairings with the corresponding slope matrix \(\theta_{ll}\). Similarly, \(d_{rr}\) is the \(m \times m\) distance matrix for RR pairings with the corresponding slope matrix \(\theta_{rr}\). These matrices are determined by

\[
d_{ll}(i, j) = ||v_{ll}(i, j)|, \quad \forall i \neq j \in \mathbb{N}^k \times k;
\]

\[
\theta_{ll}(i, j) = \angle(v_{ll}(i, j)), \quad \forall i \neq j \in \mathbb{N}^k \times k,
\]

\[
d_{rr}(i, j) = ||v_{rr}(i, j)|, \quad \forall i \neq j \in \mathbb{N}^m \times m;
\]

\[
\theta_{rr}(i, j) = \angle(v_{rr}(i, j)), \quad \forall i \neq j \in \mathbb{N}^m \times m.
\]

(10)

Let the inter-ocular distance range between \(\delta c_{\min}\) and \(\delta c_{\max}\) and the let the slope tolerance for LL and RR pairings be \(\beta_{itr}\). An LL pairing is formed between \(l(i)\) and \(l(j)\) if

\[
\delta c_{\min} \leq d_{ll}(i, j) \leq \delta c_{\max},
\]

\[
|\theta_{ll}(i, j)| \leq \beta_{itr}.
\]

(11)

are satisfied. Similarly, an RR pairing is formed between \(r(i)\) and \(r(j)\) if

\[
\delta c_{\min} \leq d_{rr}(i, j) \leq \delta c_{\max},
\]

\[
|\theta_{rr}(i, j)| \leq \beta_{itr}.
\]

(12)

are satisfied. Fig. 21(a) shows all the LL pairs in Fig. 20 while Fig. 21(b) shows the RR pairs.

4.2.2.2. Verifying “Eyeness”. Once we have the LR, LL and RR pairings, we proceed to the next phase where screening for “eyeness” is done. “Eyeness” refers to the characteristic “W”-shaped patterns that are expected to be present in the eyes. The “eyeness” verification process is the same as the one described in Section 4.1, except that the extracted regions are different; in fact, these regions differ even among the three classes themselves.

The pairing procedure results in three sets of pairs, one each for the LR, LL and RR cases. Each set of pairs corresponds to two points on the image. We will analyze the “eyeness” property for each of these sets individually. Since the main difference among the three sets is the extracted region (or regions) that is to be analyzed, we first describe these regions.

LR: Let \(lx(i), ly(i)\) denote the left and \(rx(i), ry(i)\) the right coordinates for the \(i\)th LR pair. Then the extracted region \(e_{lr}\) is the region with left and right boundaries at \(lx(i)\) and \(rx(i)\), respectively, and upper and lower boundaries determined by the vertical scale of the eye (determined by the aspect ratio of the eye) centered on the horizontal line joining the mean values of \(ly(i)\) and \(ry(i)\). The aspect ratio is the mean ratio between the horizontal and vertical extents of the eye. We use the ratio of 3:1 in favor of the horizontal axis.

LL: Let \(llx(i), lly(i)\) be the left and \(lx(i), ly(i)\) the right coordinates for the \(i\)th LL pair. In this case we expect an eye to be to the right of each of these points, and we use the estimated orbit size \(\hat{d}b\) to determine the size of the extracted region. Let \(e_{ll}(l)\) and \(e_{ll}(r)\) be the extracted regions for the left and right eyes, respectively. The left boundary for \(e_{ll}\) is \(llx(i)\) while the right boundary is determined by using \(\hat{d}b\). For the upper and lower boundaries, we center the eye on the horizontal line, take the mean of \(lly(i)\) and \(ly(i)\), and then use the vertical extent of the eye (determined by \(\hat{d}b\) and the aspect ratio of the eye) to determine these boundaries. Similarly, for \(e_{ll}(r)\) we use \(lx(i)\) as the left boundary and follow the procedure described for \(e_{ll}(l)\).

RR: Let \(rlx(i), rly(i)\) be the left and \(rx(i), ry(i)\) the right coordinates for the \(i\)th RR pair. Here we expect the eyes to be to the left of each of these points. We select two regions \(e_{rr}(l)\) and \(e_{rr}(r)\) in a symmetrical manner to the regions extracted for the LL pairs.

Once we have obtained the extracted regions, we perform the “eyeness” test as described in Section 4.1. For the case of LR pairs this becomes a test of the single region \(e_{lr}\). After performing the test on all LR pairs we keep only those pairs that pass the test. Fig. 22 shows only those LR pairs that passed the “eyeness” test.
Combining the paired LR, LL and RR responses. Then the next section will discuss a natural method of RR responses that passed this test. Fig. 23 shows only those pairs of LL and RR pairs. For the case of LL and RR pairs in which the corners are the same, i.e., \( llx(k) = lrx(m) \) and \( rrx(n) \).

Similarly, we test the pairs of extracted regions for LL and RR pairs. Fig. 23 shows only those pairs of LL and RR responses that passed this test.

We see from Figs. 22 and 23 that some false alarms still remain. The next section will discuss a natural method of combining the paired LR, LL and RR responses.

### 4.2.2.3. Combining verified pairs of responses.

In the example that we have been treating so far we have a left and right corner detection for each eye, but we also have some false alarms that need to be suppressed. This example illustrates one reason for using corner detection filters. In many cases we can use partial corner detection responses to make useful decisions. For instance, suppose we have both corner responses for the left eye but have only the left corner response for the right eye. After the pairing and verification phase we get an LR pair and an LL pair. Individually these pairings do not provide very strong evidence for detecting a pair of eyes. However, if we combine the two responses, we arrive at stronger support for the hypothesis of a pair of eyes.

Using this rationale we have developed a weighted decision procedure that depends on the number of pairs that contribute to the eye hypothesis. The procedure is based on grouping pairs of responses; it results in a non-linear decision space. This space is divided into seven regions which we number 0, 1, 1.5, 2, 3, 3.5 and 4, and denote by \( d_0, d_1, d_{1.5}, d_2, d_3, d_{3.5} \) and \( d_4 \). These regions are defined as follows:

- \( d_0 \): No LR, LL or RR pairs are present in the image.
- \( d_1 \): One LR pair is present.
- \( d_{1.5} \): Either one LL pair or one RR pair is present.
- \( d_2 \): Two LR pairs or an LL pair and an RR pair are present, conforming to the criteria for a pair of eyes.
- \( d_3 \): One LR pair, coinciding with either an LL pair or an RR pair, is present.
- \( d_{3.5} \): Two LR pairs are supported by either one LL pair or one RR pair, or an LL pair and an RR pair are supported by one LR pair.
- \( d_4 \): Two LR pairs are supported by one LL pair and one RR pair.

It should be noted that the subscripts for the decision regions are numerically in the same orders as the degrees of confidence we would assign to the decisions.

The decision is reached in two stages. The first stage pairs up LR pairs and LL and RR pairs. For the case of LR pairing, let \( k \) and \( m \) represent the indices of two LR pairs. \( k \) and \( m \) will form a \((k, m)\) pair of pairs if

1. \( llx(k) \approx lrx(m) \) and \( rrx(n) \).
2. \( lly(k) \approx lly(m) \) and \( rrx(n) \).
3. \( lly(k) \approx lly(m) \) and \( rrx(n) \).

The first condition ensures that the two pairs are in an approximate straight horizontal line. The second condition enforces an ordering requirement on the corners of the two eyes. This condition ensures that the left corner and right corner of the left eye and the left corner and right corner of the right eye occur in that order, from left to right. When both of these conditions are satisfied we say that \((k, m)\) forms a pair of LR pairs.

For the case of the \( p \)th LL pair and the \( n \)th RR pair we look for the following conditions:

1. \( llx(p) \approx lrx(n) \) and \( lrx(p) \approx rrx(n) \).
2. \( lly(p) \approx lrx(n) \) and \( rry(n) \approx rrx(n) \), and \( rly(n) \approx lry(p) \).

If these conditions are satisfied then we say that the \( p \)th LL pair and the \( n \)th RR pair form a \((p, n)\) pair of pairs.

The second stage looks for those pairs of \((p, n)\) pairs which are supported by other pairs of pairs. This means that if we have a \((k, m)\) pair of LR pairs and a \((p, n)\) pair of LL and RR pairs in which the corners are the same, i.e., \((llx(p), lly(p)) = (lx(k), ly(k)) \) and \((lrx(p), rry(n)) = (lx(m), ly(m)) \) and \((llx(n), lry(n)) = (rx(k), ry(k)) \) and \((rrx(n), rry(n)) = (rx(m), ry(m)) \), then we have full support for the pair of pairs, with the result falling in the decision region \( d_4 \). An example of \( d_4 \) is shown in Fig. 24(d).

The decision region \( d_{3.5} \) is obtained when we have a \((k, m)\) pair of LR pairs which is supported by one pair of either LL or RR, or we have a \((p, n)\) pair of LL and RR pairs which is supported by one LR pair. Fig. 24(c) is an example of this decision.
For this decision region we can denote a left-eye and an RR pair, then we are in decision region \( d_{3} \). For instance, if we have the left and right corners of the left eye and an LL pair, then we are in decision region \( d_{3} \). For this decision region we can define a binary 4-tuple which indicates the missing corner; in the example just given, the binary string 1110 indicates the absence of the right corner of the right eye. Fig. 24(b) shows an instance of this decision with the binary string 1011.

The decision region \( d_{2} \) indicates that a pair of pairs has been found but there is no cross-support. This indicates that either we have a \((k, m)\) pair of LR pairs or we have a \((p, n)\) pair of LL and RR pairs.

\( d_{1} \) is the presence of one LL or one RR pair, indicating a possible match for two eyes. \( d_{1} \) is the presence of only one LR pair, indicating one eye’s presence, as shown in Fig. 24(a). \( d_{0} \) is the null decision region in which no eyes have been found.

\[ d_{3} \]

\( 4.2.3. \) Experimental results

Our eye corner detection algorithm was tested on the same two sets of images as in Section 4.1. In the Aberdeen database we set the radius of the filter at 17 pixels and kept it constant, as described in Section 4.2.1. The normalized and rescaled flesh-tone region \( D_{\text{face}} \) and eye-tone region \( D_{\text{eye}} \) were as follows:

\[ D_{\text{face}} = (r, g) \in 39 \leq r \leq 48, \quad 30 \leq g \leq 35, \]  

(13)

\[ D_{\text{eye}} = (r, g) \in 32 \leq r \leq 38, \quad 30 \leq g \leq 36. \]  

(14)

For pairing responses, \( d_{1} \), \( d_{2} \) and \( d_{0} \) were determined by the scale selection method, while the rotation tolerances were set at \( \beta_{r} = 10^\circ \) and \( \beta_{l} = \beta_{r} = 7^\circ \).

Tables 4 and 5 show the results of all the steps in applying non-linear filtering and post-processing to the Aberdeen database. The first column gives the image number; the second and third columns give the numbers of right corner and left corner responses, respectively; the fourth, fifth and sixth columns give the numbers of LR, LL and RR pairings, respectively. The “Verified LR and LL/RR” columns give the numbers of these pairs that pass the “eyeness” criterion. The “Final” column indicates the final number of points obtained by combining the verified pairs of responses, and the “d” column gives the decision region that the image lies in; this indicates the level of confidence.

In the introduction to this section we alluded to the fact that detecting left and right corners of the eye provides rich information for the subsequent post-processing. As can be seen in Tables 4 and 5, we can make useful decisions even when all four corners of the two eyes are not detected. An example is provided by the decision region \( d_{3} \) obtained for image “im14” by combining one LL pair with one LR pair. The binary string associated with this decision was 1011. The non-linear filtering process failed to find the right corner of the left eye; this is seen from the binary string associated with the decision. This additional information is lacking in the linear filtering method and is a contributing factor in the better performance of the non-linear filtering method.

When we applied the non-linear filtering procedure to the frame sequence the radius for the filter was set at 11. The parameter values for \( D_{\text{face}} \) and \( D_{\text{eye}} \) are also different; they are

\[ D_{\text{face}} = (r, g) \in 46 \leq r \leq 55, \quad 29 \leq g \leq 34, \]  

(15)

\[ D_{\text{eye}} = (r, g) \in 35 \leq r \leq 44, \quad 30 \leq g \leq 36. \]  

(16)

For pairing responses, \( d_{1} \), \( d_{2} \) and \( d_{0} \) were determined by the scale selection procedure and the values of \( \beta_{l}, \beta_{r} \) and \( \beta_{l} \) were the same as the ones used in the first dataset.

Table 6 gives the results of all the intermediate steps in applying non-linear filtering and post-processing to the frame sequence, and Fig. 25 shows the 21 frames with the final responses (having confidence levels 1.5 or greater) marked by dots. In this sequence the subject is moving her head as well as her eyes. Analyzing the results shows an evolving change in the number of detected corners as the iris is rolled from one side to the other. From the detections observed in the latter part of the sequence (corresponding to images “im17”–“im21” in

![Fig. 24. Detection of eyes according to confidence measure.](image)
Table 4
Intermediate and final results of non-linear filtering for images 1–36

<table>
<thead>
<tr>
<th>Image no.</th>
<th>Right</th>
<th>Left</th>
<th>LR</th>
<th>LL</th>
<th>RR</th>
<th>Pass LR</th>
<th>Pass LL/RR</th>
<th>Final</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>im1</td>
<td>10</td>
<td>7</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>im2</td>
<td>13</td>
<td>12</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>im3</td>
<td>15</td>
<td>14</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>im4</td>
<td>9</td>
<td>10</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im5</td>
<td>12</td>
<td>12</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im6</td>
<td>23</td>
<td>16</td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im7</td>
<td>24</td>
<td>19</td>
<td>5</td>
<td>4</td>
<td>10</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im8</td>
<td>22</td>
<td>18</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>im9</td>
<td>19</td>
<td>17</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im10</td>
<td>8</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im11</td>
<td>9</td>
<td>10</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>im12</td>
<td>9</td>
<td>9</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im13</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im14</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>im15</td>
<td>6</td>
<td>13</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im16</td>
<td>6</td>
<td>13</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im17</td>
<td>8</td>
<td>17</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im18</td>
<td>12</td>
<td>20</td>
<td>8</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im19</td>
<td>10</td>
<td>15</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im20</td>
<td>11</td>
<td>7</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im21</td>
<td>6</td>
<td>11</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im22</td>
<td>7</td>
<td>10</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im23</td>
<td>6</td>
<td>9</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>im24</td>
<td>9</td>
<td>10</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>im25</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im26</td>
<td>9</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im27</td>
<td>9</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im28</td>
<td>18</td>
<td>11</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>im29</td>
<td>29</td>
<td>44</td>
<td>12</td>
<td>15</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im30</td>
<td>24</td>
<td>16</td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im31</td>
<td>25</td>
<td>22</td>
<td>9</td>
<td>1</td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im32</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im33</td>
<td>7</td>
<td>8</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im34</td>
<td>10</td>
<td>13</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>im35</td>
<td>5</td>
<td>7</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>im36</td>
<td>8</td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 6), it is evident that as the irises move towards the left corners of their respective eyes we are able to detect the opposing (right) corners. If we consider this as a single image the confidence measure would be very low, 1.5; however, considering each image as part of the sequence, we are able to reach valuable conclusions about the movement of the iris. Such qualitative analyses are of great importance when tracking of eye gaze direction is required.

5. Conclusions

Our method of eye corner detection involves nonlinear filtering using color-based wedge-shaped filters. This approach proved to be superior to the one using Gabor wavelet-based linear filtering due to the fact that it was not sensitive to squint, and it also provided rich information, in the form of right and left corner responses, for post-processing. For the Aberdeen database a detection rate of 90% was achieved with no false alarms, when setting the confidence threshold at 2. We found that most of the failures were due to incorrect estimation of the orbit size from the face blob. For the frame sequence we set the confidence threshold at 1.5 and achieved a detection rate of 90% with no false alarms. We used a lower confidence threshold in this sequence to allow for the eye gaze variations. As we see from Fig. 25, when the iris is shifted to either the right or left corner of the eye, only the opposite corner is detected.
Table 5
Intermediate and final results of non-linear filtering for images 37–72

<table>
<thead>
<tr>
<th>Image no.</th>
<th>Right</th>
<th>Left</th>
<th>LR</th>
<th>LL</th>
<th>RR</th>
<th>Pass LR</th>
<th>Pass LL/RR</th>
<th>Final</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>im37</td>
<td>6</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1.5</td>
</tr>
<tr>
<td>im38</td>
<td>4</td>
<td>9</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>im39</td>
<td>6</td>
<td>16</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>im40</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>im41</td>
<td>7</td>
<td>10</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3.0</td>
</tr>
<tr>
<td>im42</td>
<td>7</td>
<td>11</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>im43</td>
<td>4</td>
<td>12</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im44</td>
<td>14</td>
<td>18</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>im45</td>
<td>12</td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im46</td>
<td>5</td>
<td>11</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im47</td>
<td>10</td>
<td>13</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>im48</td>
<td>23</td>
<td>13</td>
<td>7</td>
<td>2</td>
<td>12</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im49</td>
<td>9</td>
<td>10</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im50</td>
<td>10</td>
<td>9</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im51</td>
<td>10</td>
<td>14</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im52</td>
<td>7</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im53</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im54</td>
<td>13</td>
<td>9</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im55</td>
<td>27</td>
<td>28</td>
<td>10</td>
<td>7</td>
<td>9</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td>im56</td>
<td>24</td>
<td>23</td>
<td>9</td>
<td>6</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im57</td>
<td>23</td>
<td>24</td>
<td>9</td>
<td>6</td>
<td>8</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im58</td>
<td>13</td>
<td>21</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im59</td>
<td>11</td>
<td>9</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im60</td>
<td>17</td>
<td>11</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im61</td>
<td>7</td>
<td>12</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3.0</td>
</tr>
<tr>
<td>im62</td>
<td>5</td>
<td>11</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td>im63</td>
<td>13</td>
<td>9</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>im64</td>
<td>8</td>
<td>7</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im65</td>
<td>25</td>
<td>22</td>
<td>7</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im66</td>
<td>23</td>
<td>19</td>
<td>7</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im67</td>
<td>9</td>
<td>11</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im68</td>
<td>17</td>
<td>13</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im69</td>
<td>18</td>
<td>13</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im70</td>
<td>10</td>
<td>10</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im71</td>
<td>9</td>
<td>11</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im72</td>
<td>7</td>
<td>17</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
</tbody>
</table>

Table 6
Intermediate and final results of non-linear filtering for frames 1–21

<table>
<thead>
<tr>
<th>Image no.</th>
<th>Right</th>
<th>Left</th>
<th>LR</th>
<th>LL</th>
<th>RR</th>
<th>Pass LR</th>
<th>Pass LL/RR</th>
<th>Final</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>im1</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im2</td>
<td>7</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im3</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im4</td>
<td>8</td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im5</td>
<td>4</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im6</td>
<td>6</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>im7</td>
<td>7</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>3.5</td>
</tr>
<tr>
<td>im8</td>
<td>7</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im9</td>
<td>9</td>
<td>10</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im10</td>
<td>8</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im11</td>
<td>9</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im12</td>
<td>9</td>
<td>9</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3.0</td>
</tr>
<tr>
<td>im13</td>
<td>8</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3.0</td>
</tr>
<tr>
<td>im14</td>
<td>7</td>
<td>10</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1.5</td>
</tr>
<tr>
<td>im15</td>
<td>7</td>
<td>8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3.0</td>
</tr>
<tr>
<td>im16</td>
<td>9</td>
<td>7</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>im17</td>
<td>10</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1.5</td>
</tr>
<tr>
<td>im18</td>
<td>11</td>
<td>7</td>
<td>3</td>
<td>0</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3.0</td>
</tr>
<tr>
<td>im19</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1.5</td>
</tr>
<tr>
<td>im20</td>
<td>6</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1.5</td>
</tr>
<tr>
<td>im21</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1.5</td>
</tr>
</tbody>
</table>
In summary, the non-linear filtering method had a higher level of performance in detecting the eyes than traditional linear filtering approaches. As a by-product, detection of the corners of the eyes provides landmark points that can be used in eye gaze tracking.

Acknowledgements

The support of the Office of Naval Research under Grant N00014-95-1-0521 is greatfully acknowledged. The authors also thank Prof. Rama Chellappa for his guidance in the development of the linear filtering method.

References

About the Author

SAADA SIROHEY received his B.Sc. with highest honors in electrical engineering from King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia in 1990, and his MS and Ph.D. degrees from the University of Maryland, College Park, also in electrical engineering. He is currently working at General Electric Medical Systems in medical imaging. His research interests are signal/image processing and image understanding. He has done research in automated face detection/recognition and tracking with emphasis on eye gaze tracking. He is currently working on bio-medical imaging.

About the Author

AZRIEL ROSENFELD is a tenured Research Professor, a Distinguished University Professor, and Director of the Center for Automation Research at the University of Maryland in College Park. He also holds affiliate professorships in the Departments of Computer Science, Electrical Engineering, and Psychology. He holds a Ph.D. in mathematics from Columbia University (1957), rabbinic ordination (1952) and a Doctor of Hebrew Literature degree (1955) from Yeshiva University, and honorary Doctor of Technology degrees from Linköping University, Sweden (1980) and Oulu University, Finland (1994).

Dr. Rosenfeld is widely regarded as the leading researcher in the world in the field of computer image analysis. Over a period of 35 years he has made many fundamental and pioneering contributions to nearly every area of that field. He wrote the first textbook in the field (1969); was founding editor of its first journal (1972); and was co-chairman of its first international conference (1987). He has published over 30 books and over 600 book chapters and journal articles, and has directed over 50 Ph.D. dissertations. In 1985 he served as chairman of a panel appointed by the National Research Council to brief the President’s Science Advisor on the subject of computer vision; he has also served (1985–1988) as a member of the Vision Committee of the National Research Council. In honor of his 65th birthday, a book entitled “Advances in Image Understanding — A Festschrift for Azriel Rosenfeld”, edited by Kevin Bowyer and Narendra Ahuja, was published by IEEE Computer Society Press in 1996.

He is a Fellow of the Institute of Electrical and Electronics Engineers (1971), and won its Emanuel Piore Award in 1985; he is a founding Fellow of the American Association for Artificial Intelligence (1990) and of the Association for Computing Machinery (1993); he is a Fellow of the Washington Academy of Sciences (1988), and won its Mathematics and Computer Science Award in 1988; he was a founding Director of the Machine Vision Association of the Society of Manufacturing Engineers (1985–1988), won its President’s Award in 1987 and is a certified Manufacturing Engineer (1988); he was a founding member of the IEEE Computer Society’s Technical Committee on Pattern Analysis and Machine Intelligence (1965), served as its Chairman (1985–1987), and received the Society’s Meritorious Service Award in 1986, its Harry Goode Memorial Award in 1995, and became a Golden Core member of the Society in


1996; he received the IEEE Systems, Man, and Cybernetics Society’s Norbert Wiener Award in 1995; he received an IEEE Standards Medalion in 1990, and the Electronic Imaging International Imager of the Year Award in 1991; he was a founding member of the Governing Board of the International Association for Pattern Recognition (1978–1985), served as its President (1980–1982), won its first K.S. Fu Award in 1988, and became one of its founding Fellows in 1994; he received the Information Science Award from the Association for Intelligent Machinery in 1998; he was a Foreign Member of the Academy of Science of the German Democratic Republic (1988–1992), and is a Corresponding Member of the National Academy of Engineering of Mexico (1982).