Robust real-time pupil tracking in highly off-axis images

Lech Świrski∗ Andreas Bulling† Neil Dodgson‡
University of Cambridge University of Cambridge University of Cambridge

Abstract

Robust, accurate, real-time pupil tracking is a key component for online gaze estimation. On head-mounted eye-trackers, existing algorithms that rely on circular pupils or contiguous pupil regions fail to detect or accurately track the pupil. This is because the pupil ellipse is often highly eccentric and partially occluded by eyelashes. We present a novel, real-time dark-pupil tracking algorithm that is robust under such conditions. Our approach uses a Haar-like feature detector to roughly estimate the pupil location, performs a k-means segmentation on the surrounding region to refine the pupil centre, and fits an ellipse to the pupil using a novel image-aware Random Sample Consensus (RANSAC) ellipse fitting. We compare our approach against existing real-time pupil tracking implementations, using a set of manually labelled infra-red dark-pupil eye images. We show that our technique has a higher pupil detection rate and greater pupil tracking accuracy.

1 Introduction

Although historically research has focused on remote eye-tracking, there has been a recent trend in algorithms specifically designed for cheap, webcam-based, head-mounted eye-trackers [Chau and Betke 2005; San Agustin et al. 2010]. While head-mounting simplifies many tasks such as compensating for head movement, low-cost systems are likely to have a lower build quality, and therefore algorithms cannot rely on hardware invariants, such as positions of lights for glints or calibration of pairs of cameras. Additionally, the proximity of the cameras in a head-mounted tracker means that they have to be positioned at a large angle to the visual axis, so as not to block the user’s gaze. The closer the camera is to the eye, the larger this angle has to be, which creates novel challenges in detecting the pupil: the pupil ellipse becomes increasingly eccentric and eyelashes become increasingly obstructive.

Despite these issues, it can be desirable to mount cameras very close to the eye. An example is eye-tracking on systems which place an obstruction in front of the eyes, such as glasses or head-mounted displays, where the eye camera must be positioned between the obstruction and the eye (figure 1). In such cases, standard pupil detection and tracking algorithms fail to find the pupil.

We present a real-time dark-pupil tracking algorithm designed for low-cost head-mounted active-IR hardware. Our algorithm is robust to highly eccentric pupil ellipses and partial obstructions from eyelashes, making it suitable for use with cameras mounted close to the eye. It first computes a fast initial approximation of the pupil position, and performs a novel RANSAC-based ellipse fitting to robustly refine this approximation.

2 Our pupil tracking algorithm

Our approach works in three stages:

1. Approximate the pupil region using a fast, simple feature detection, to reduce the search space in the following stages.
2. Use a k-means histogram segmentation to refine the approximation of the pupil region, and find an initial approximation to the pupil centre.
3. Refine the pupil centre and find its elliptical outline, using a novel ellipse fitting algorithm.

2.1 Initial region estimation

Our initial region estimation assumes that the pupil region, either the dark pupil itself or the combination of pupil and iris, can roughly be described as “a dark blob surrounded by a light background”, and is the strongest such feature in the image. To find the pupil, we use a Haar-like feature detector, similar to the features used in cascade classifiers [Viola and Jones 2001].

The core idea of the feature detector can be explained in terms of convolution. To find possible pupil regions, we convolve the image with a Haar-like centre-surround feature of a given radius (figure 2). We repeat this for a set of possible radii, between a user specified minimum and maximum, and find the strongest response over the 3D space of \((x, y)\) and radii. The \((x, y)\) location of this strongest response is assumed to be the centre of the pupil region, with the size of the region determined by the corresponding radius.

Although such a convolution is a slow operation if performed naively, we optimise this by first calculating the integral image [Viola and Jones 2001]. Using this integral image, we can find the response of a pixel to a Haar-like feature in constant time, only needing to sample 8 pixel values, one for each corner of the two squares, thereby making this step linear in the number of pixels and possible radii.

2.2 Kmeans pupil segmentation

The initial region estimation is unlikely to be accurately centred on the pupil. The Haar-like feature is square, and so is only an approximation to the elliptical pupil shape. Furthermore, the magnitude of the response will be similar regardless of where the pupil is in the
We use \( k \) with thresholding. We could have used other approaches, such as which adapts to such changes. This is a common approach in real-time pupil detection. It is critical to the performance of the pupil tracker, but it is often simply a free parameter of the algorithm, and hence affected by changes in illumination or camera settings. Instead of this manual parameter setting, we wish to have a fully automatic threshold calculation, which adapts to such changes.

We choose to consider this as an intensity-based image segmentation problem. Our approach is to segment the image histogram into two clusters, corresponding to pupil and background intensity values. We use \( k \)-means clustering on the histogram of the pupil region to find two clusters: dark and light (figure 3b). The dark cluster is then assumed to correspond to the pupil pixels, and we create a segmented binary image of the pupil region by thresholding any pixels above the maximum intensity in the dark cluster.

Finally, we find connected components in the segmented image [Chang et al. 2004], and select the largest to be the pupil. The centre of mass of this component approximates the pupil position.

This \( k \)-means segmentation is a fast and simple approach to segment the pupil region, chosen for its simplicity and natural equivalence with thresholding. We could have used other approaches, such as fitting Gaussian mixture models (GMMs) to the histogram, or using graph cuts [Boykov and Jolly 2001], however, we found that such techniques did not offer a sufficient improvement on the position estimate to justify their increased computational cost. Furthermore, although such techniques could offer a more accurate segmentation where the pupil is visible, the presence of occlusions would still require the position estimate to be further refined.

inner square, so the feature is unlikely to be centred on the pupil. Hence, in the next stage, we approximate the pupil location within this region (figure 3).

A common approach in real-time pupil detection is to assume that the pupil is the darkest element in the image, and find it by applying intensity thresholding to the image. The value of the threshold is critical to the performance of the pupil tracker, but it is often simply a free parameter of the algorithm, and hence affected by changes in illumination or camera settings. Instead of this manual parameter setting, we wish to have a fully automatic threshold calculation, which adapts to such changes.

We choose to consider this as an intensity-based image segmentation problem. Our approach is to segment the image histogram into two clusters, corresponding to pupil and background intensity values. We use \( k \)-means clustering on the histogram of the pupil region to find two clusters: dark and light (figure 3b). The dark cluster is then assumed to correspond to the pupil pixels, and we create a segmented binary image of the pupil region by thresholding any pixels above the maximum intensity in the dark cluster.

Finally, we find connected components in the segmented image [Chang et al. 2004], and select the largest to be the pupil. The centre of mass of this component approximates the pupil position.

This \( k \)-means segmentation is a fast and simple approach to segment the pupil region, chosen for its simplicity and natural equivalence with thresholding. We could have used other approaches, such as fitting Gaussian mixture models (GMMs) to the histogram, or using graph cuts [Boykov and Jolly 2001], however, we found that such techniques did not offer a sufficient improvement on the position estimate to justify their increased computational cost. Furthermore, although such techniques could offer a more accurate segmentation where the pupil is visible, the presence of occlusions would still require the position estimate to be further refined.

2.3 Pupil ellipse fitting

The final stage of our algorithm refines the pupil position estimate using an ellipse-fitting approach. Ellipse fitting is a common refinement method in pupil tracking techniques [Hansen and Ji 2010]; our approach is inspired in particular by Starburst [Li et al. 2005].

We once again consider only the pupil region, centred around the current pupil position estimate. We find the pupil in this region by fitting an ellipse to the boundary between the pupil and the iris. To do this, we preprocess the image to create an edge image and robustly fit an ellipse to the edge points while ignoring any outliers.

2.3.1 Image preprocessing

To remove features such as eyelashes and glints, we first perform a morphological ‘open’ operation, which closes small bright gaps in the otherwise dark pupil region, without significantly affecting the pupil’s contour (figure 4b). Although morphological operations would be prohibitively computationally expensive if done on the entire image, performing them on the pupil sub-region is acceptable.

We then find the boundary between pupil and iris using a Canny edge detector (figure 4c). We used thresholds of 30 and 50 as parameters to the edge detector, although we found that any thresholds within a similar order of magnitude gave equally good results across all the datasets we tried, due to the high contrast between pupil and iris.

2.3.2 Robust ellipse fitting

In the final stage, we fit an ellipse to the edge pixels. An ellipse can be fitted to any set of five or more points using a direct least squares method [Fitzgibbon et al. 1999], however there are likely to be pixels in the edge image which do not correspond to the pupil boundary, due to image noise, occlusions, or other strong features such as eyelids and the limbus. A least-squares technique will be strongly affected by such outliers, so we require a technique which will fit an ellipse to the pupil edge while being robust to outliers.

There are two main methods of robustly fitting ellipses to data: voting-based methods, such as the Hough transform, and searching-based methods, such as Random Sample Consensus (RANSAC) [Fischler and Bolles 1981]. Voting-based methods use a parameter accumulator: every pixel in the image votes for the parameters of all ellipses that it can be part of. Maxima in the accumulator correspond to discovered ellipses. This is accurate, but computationally expensive, as it requires a 5D accumulator for the five ellipse parameters.

Searching-based methods instead test possible ellipses, and select the one with greatest support. A classic example is RANSAC, which
We define our support function as

\[ \alpha Q(x, y) = \frac{[\nabla Q(x, y)]^2}{|\nabla Q(x, y)|} \]

where the ellipse is the isocontour at 0, which reduces the support of ellipses not aligned with the pupil in the image. This is the approach taken by most ellipse fitting algorithms. How-ever, we notice that we want our ellipse to lie on a boundary from being one pixel away from the minor axis of the ellipse is set to 1.

The resulting best fit is the model with maximal support.

We evaluate the pixel accuracy of the pupil ellipse fit by comparing against ground-truth data. For the ground truth, we use 600 hand-labelled eye images. The eye images were obtained as a uniformly random subset of left and right eye videos from two people, collected using a head-mounted camera system. These were labelled by fitting

**Algorithm 1**

```plaintext
Procedure RANSAC-ELLIPSE-FIT(points, image, N, \epsilon)
  best-ellipse ← null
  best-support ← − inf
  repeat N times
    sample ← RANDOM-SAMPLE(points, 5)
    ellipse ← FIT-ELLIPSE(sample)
    // Early sample rejection
    if \exists (x, y) ∈ sample where \nabla \text{ellipse}(x, y) \cdot \nabla \text{image}(x, y) ≤ 0 then
      continue // reject sample, skip this iteration
    end if
    // Iteratively refine inliers (we use M = 2)
    repeat M times
      inliers = \{ (x, y) ∈ points | error(ellipse, x, y) < \epsilon \}
      ellipse ← FIT-ELLIPSE(inliers)
    end repeat
    // Calculate the support of the ellipse
    support ← support(ellipse, image, inliers)
    if support > best-support then
      best-ellipse ← ellipse
      best-support ← support
    end if
    // Early termination for ≥ 95% inliers
    if |inliers| ≥ 0.95 \cdot |points| then
      break
    end if
  end repeat
  return best-ellipse
```
an ellipse to 5 or more manually selected points along the pupil boundary, discarding any images where the pupil is not visible. This data set is publically available\(^1\).

To compare our ellipse fit to the ground truth, we used the Hausdorff distance between the two ellipses. The Hausdorff distance between two ellipses finds the maximum Euclidean distance of one ellipse to any point on the other ellipse; in our case, we discretely approximate this by selecting 100 evenly spaced points along each ellipse. Note that we used the exact Euclidean pupil–ellipse distance rather than the approximate Euclidean distance used in our algorithm (eq. 2).

We also compared our approach to the publicly available implementation of Starburst [Li et al. 2005], and our own re-implementation of the pupil tracker from the ITU Gaze Tracker [San Agustin et al. 2010]. The latter technique finds a pupil region, and returns the centre of mass of the region as the pupil location—we extended this to provide a pupil ellipse by fitting an ellipse to the second moments of the region. Both of these techniques use a threshold parameter—in Starburst it is the edge threshold, in the ITU Gaze Tracker it is an intensity threshold. We adjusted the value of the threshold for each image sequence, to optimise the result.

We compare these three approaches by calculating the pupil detection rate, for various ellipse error thresholds. Figure 6a shows the result of this comparison. Our approach has a much higher detection rate, of over 87% on our data set within an error threshold of 5 pixels. To compare, the ITU Gaze Tracker has a detection rate of 40%, and Starburst less than 15%.

To evaluate the trade-off between framerate and accuracy, we ran our approach adjusting the number of RANSAC iterations. More iterations are more likely to give a better fit, however also increase the execution time, decreasing framerate.

We ran the evaluation both on our approach as described, and on an implementation which replaced the image-aware support function (eq. 4) with the standard number-of-inliers support function. Additionally, we ran the evaluation on an implementation which added an edge pixel selection step (similar to Starburst’s) between the image processing and RANSAC stages, again both with image-aware support and with number-of-inliers support. The framerate was calculated from the execution time on video data, using a C++ implementation on a quad-core 2.80 GHz CPU.

Figure 6b demonstrates this comparison. Our novel image-aware support function (solid lines) consistently achieves a lower mean error than using the number-of-inliers support (dashed line), regardless of whether we include the edge pixel selection step. Furthermore, our approach has a clear trade-off between framerate and accuracy.

---

\(^1\)http://www.cl.cam.ac.uk/research/rainbow/projects/pupitracking/

---

3 Conclusion

We have presented a novel pupil-tracking algorithm which is robust to occlusions such as eyelashes, and to the highly elliptical pupil shape caused by mounting a camera close to the eye (figure 7). Our major contributions are:

- A fast pupil position approximation using Haar-like features.
- Using a \(k\)-means segmentation approach for automatically selecting a pupil threshold.
- A novel formulation of RANSAC ellipse fitting, which robustly fits an ellipse to a set of 2D points using not only the 2D point positions but also image data.
- A publically available data set and ellipse distance metric for evaluating pupil ellipse fitting.

We evaluated our approach using a hand-labelled ground truth data set, and we have shown that our approach is robust and accurate at real time framerates, and can increase its robustness and accuracy for offline analysis.

References


