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

Lech Świrski* University of Cambridge
Andreas Bulling† University of Cambridge
Neil Dodgson‡ 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 Concensus (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 the performs a novel RANSAC-based ellipse fitting to robustly refine this approximation.

*e-mail: lech.swirski@cl.cam.ac.uk
†e-mail: andreas.bulling@acm.org
‡e-mail: neil.dodgson@cl.cam.ac.uk

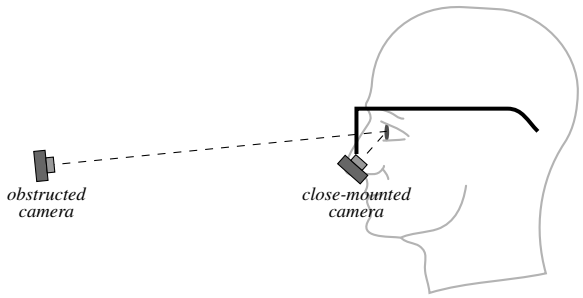


Figure 1: *Glasses in front of the eyes obstructs the view of normal eye-trackers. A camera mounted between the glasses and the eye can see the pupil, albeit at a large angle to the viewing axis.*

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 naïvely, 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 inner square, so the feature is unlikely to be centred on the pupil.

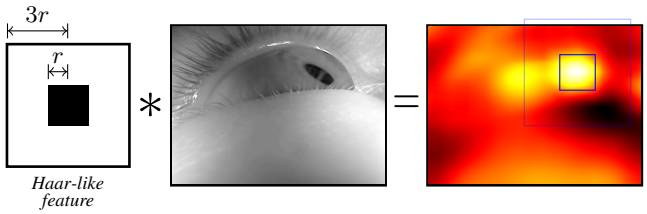
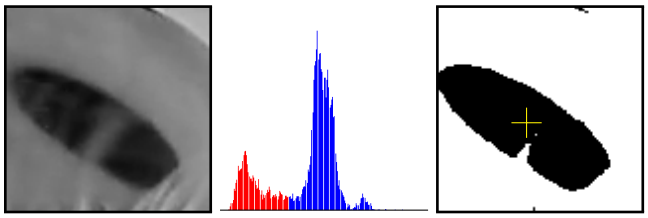


Figure 2: *To find the approximate pupil region, the eye image is convolved with a Haar-like centre surround feature of radius r. The pupil region is centred on the location of the maximal response over all pixels and radii.*



(a) *Pupil region* **(b)** *Region histogram* **(c)** *Segmented pupil*

Figure 3: *The pupil region (a) is segmented using k-means clustering of its histogram (b). The largest black region in the resulting segmented image is assumed to be the pupil (c).*

Hence, in the next stage, we approximate the pupil location within this region (figure 3).

A common approch 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.

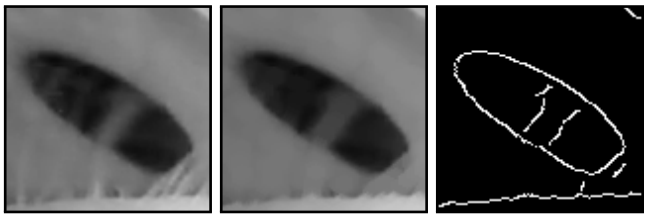
We choose to consider this as an intensity-based image segmentation problem. Our approach is to segment the image histogram into two clusters, corresponding to pupil and background intensity values. We use k-means clustering on the histogram of the pupil region to find two clusters: dark and light (figure 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].



(a) *Pupil region* **(b)** *Morphological open* **(c)** *Canny edges*

Figure 4: *The refined pupil region (a) is preprocessed using a morphological 'open' operation, which removes small occlusions and noise (b). This opened image is then passed through a Canny edge detector (c).*

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.

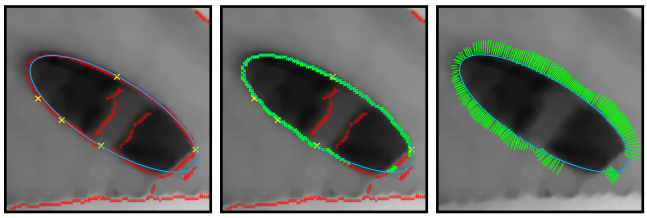
We then find the boundary between pupil and iris using a Canny edge detector (figure 4c). We used thresholds of 30 and 50 as parameters to the edge detector, although we found that any thresholds within a similar order of magnitude gave equally good results across all the datasets we tried, due to the 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 are exhaustive, but computationally expensive. Searching-based methods instead test a subset of possible ellipses, and select the best. A classic example is RANSAC, which is a generic model-fitting approach. RANSAC finds the best model for a set of data by repeatedly minimally sampling the data, fitting a model to the sample, and calculating the support for that model. The resulting best fit is the model with maximal support.

Our technique uses RANSAC to fit an elliptical model to the edge points, and we introduce a novel image-aware support function which reduces the support of ellipses not aligned with the pupil in the image. For each RANSAC iteration, we use the direct least squares method [Fitzgibbon et al. 1999] on a minimal sample of



(a) *Sample ellipse fit* **(b)** *Inlier ellipse fit* **(c)** *Inlier gradients*

Figure 5: *In each RANSAC iteration, we sample 5 random edge points and fit an ellipse to them (a). We then find inliers to this ellipse fit, and refit the ellipse (b). The quality of the fit is found by finding the image gradient at each inlier (c), and summing the magnitude of the gradients in the direction orthogonal to the ellipse.*

five points (figure 5a). The standard RANSAC support function then finds inliers to the model fit using a threshold on some error function — in this case, a set of points which are sufficiently close to the boundary of the ellipse (figure 5b) — and calculates support as the size of the set of inliers. Finding the Euclidean distance of a point from the boundary of an ellipse is a non-trivial operation, involving solving a quartic equation; instead, we use an approximation. We represent the ellipse by its conic equation,

$$Q(x, y) = Ax^2 + Bxy + Cy^2 + Dx + Ey + F \qquad (1)$$

where the ellipse is the isocontour at 0. The error function we use is EOF_2 from Rosin's survey [1996], defined as

$$\text{error}(Q, x, y) = \alpha \frac{Q(x, y)}{|\nabla Q(x, y)|} \qquad (2)$$

where α normalises the values of error(x, y) so that the error of being one pixel away from the minor axis of the ellipse is set to 1. The set of inliers is thus defined as:

$$\text{inliers} = \{\, (x, y) \mid \text{error}(Q, x, y) < \varepsilon \,\} \qquad (3)$$

This is the approach taken by most ellipse fitting algorithms. However, we notice that we want our ellipse to lie on a boundary from dark pixels to light pixels, and hence wish to prefer such ellipses. Furthermore, strong image edges are more likely to be part of the pupil contour, therefore we wish to prefer ellipses lying along strong edges. We therefore wish to have an image-aware support function which takes these into account.

We define our support function as

$$\text{support}(Q, I, \text{inliers}) = \sum_{(x,y)\in\text{inliers}} \frac{\nabla Q(x, y)}{|\nabla Q(x, y)|} \cdot \nabla I(x, y) \qquad (4)$$

This support function still has a preference for large sets of inliers. However, it adds weight to inliers where the direction of the ellipse gradient $\nabla Q(x, y)$ agrees with the image gradient $\nabla I(x, y)$, and adds negative weight if the gradients oppose. Furthermore, it strengthens this weight where the magnitude of the image gradient is large (figure 5c).

We make three additional changes to the RANSAC algorithm. Firstly, we add an early rejection step for the initial five point sample, if the five points' ellipse gradients do not agree with the image gradients. Secondly, we noticed that an ellipse fit to a poor sample will still find sufficient inliers to provide a good fit. Hence, we iterate the ellipse fitting and inlier selection step, which increases the likelihood that a given sample will result in a good fit, and allows us to perform fewer

Algorithm 1 Our image-aware approach to fit an ellipse to a set of points in an image, using N iterations and an inlier threshold of ϵ.

procedure RANSAC-ELLIPSE-FIT(*points*, *image*, N, ϵ)
  best-ellipse $\leftarrow$ **null**
  best-support $\leftarrow -\inf$

  // Perform N RANSAC iterations
  repeat N **times**
    sample $\leftarrow$ RANDOM-SAMPLE(*points*, 5)
    ellipse $\leftarrow$ FIT-ELLIPSE(*sample*)

    // Early sample rejection
    if $\exists(x, y) \in$ *sample* **where**
        ∇*ellipse*$(x, y) \cdot \nabla$*image*$(x, y) \leq 0$ **then**
      continue *// reject sample, skip this iteration*
    end if

    // Iteratively refine inliers (we use M = 2)
    repeat M **times**
      inliers $= \{\, (x, y) \in$ *points* $\mid$ error(*ellipse*, $x, y) < \epsilon \,\}$
      ellipse $\leftarrow$ FIT-ELLIPSE(*inliers*)
    end repeat

    // Calculate the support of the ellipse
    support $\leftarrow$ support(*ellipse*, *image*, *inliers*)
    if *support* $>$ *best-support* **then**
      best-ellipse $\leftarrow$ *ellipse*
      best-support $\leftarrow$ *support*
    end if

    // Early termination for $\geq 95\%$ inliers
    if $|$*inliers*$| \geq 0.95 \cdot |$*points*$|$ **then**
      break
    end if
  end repeat

  return *best-ellipse*
end procedure

RANSAC iterations. Finally, as is common in RANSAC algorithms, we perform an early termination if our inlier set is sufficiently large — we used 95% of the size of the input point set.

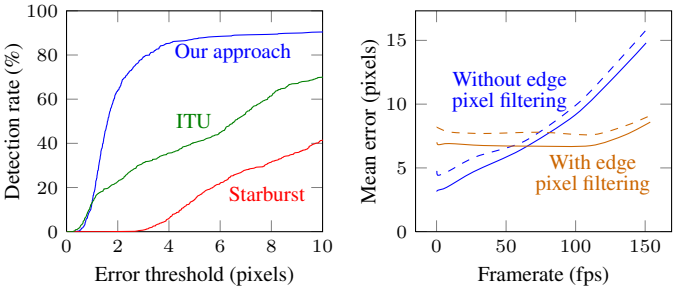
Our final robust, image-aware ellipse fitting algorithm is described in algorithm 1. Internally, we still use the direct least-squares approach to ellipse fitting, however performing this ellipse fitting on only inliers ensures that the overall fit is robust.

2.4 Evaluation

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 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

[1] http://www.cl.cam.ac.uk/research/rainbow/projects/pupiltracking/



(a) *Detection rate* **(b)** *Framerate of our approach*

Figure 6: *(a): detection rate of our approach compared to existing approaches. (b): mean error vs. framerate of our approach, with image-aware support (solid lines) and without (dashed lines).*

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 consistently achieves a lower mean error than using the number-of-inliers support, regardless of whether we include the edge pixel selection step. Furthermore, our approach has a clear trade-off between framerate and accuracy. For webcam-based systems, with framerates of 30–60 fps, or offline systems with lower framerate, our approach (blue) outperforms the edge pixel filtering variation (orange). This is because we can find "difficult" ellipses where the edge point selection fails to select inlier points. For extremely high framerate systems, above 60 fps, the edge pixel filtering variation achieves a lower mean error than our approach alone, as it filters out a large number of outliers, increasing the probability that a given sample will consist of only inliers.

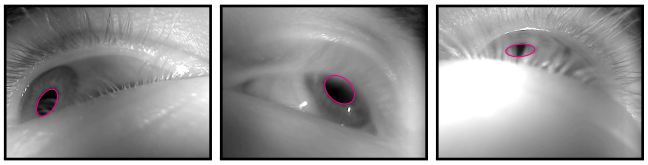


Figure 7: *Example results of our approach on difficult images from our evaluation data set.*

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 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

BOYKOV, Y. Y., AND JOLLY, M.-P. 2001. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In *Proc. ICCV*, 105–112.

CHANG, F., CHEN, C.-J., AND LU, C.-J. 2004. A linear-time component-labeling algorithm using contour tracing technique. *Computer Vision and Image Understanding 93*, 2, 206–220.

CHAU, M., AND BETKE, M. 2005. Real Time Eye Tracking and Blink Detection with USB Cameras. Tech. rep., Boston University Computer Science.

FISCHLER, M. A., AND BOLLES, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM 24*, 6 (June), 381–395.

FITZGIBBON, A., PILU, M., AND FISHER, R. B. 1999. Direct least square fitting of ellipses. *IEEE TPAMI 21*, 5, 476–480.

HANSEN, D. W., AND JI, Q. 2010. In the eye of the beholder: a survey of models for eyes and gaze. *IEEE TPAMI 32*, 3, 478–500.

LI, D., WINFIELD, D., AND PARKHURST, D. J. 2005. Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches. In *Proc. IEEE Vision for Human-Computer Interaction Workshop*, 1–8.

ROSIN, P. L. 1996. Analysing error of fit functions for ellipses. *Pattern Recognition Letters 17*, 14, 1461–1470.

SAN AGUSTIN, J., SKOVSGAARD, H., MOLLENBACH, E., BARRET, M., TALL, M., HANSEN, D. W., AND HANSEN, J. P. 2010. Evaluation of a low-cost open-source gaze tracker. In *Proc. ETRA*, 77–80.

VIOLA, P., AND JONES, M. 2001. Rapid object detection using a boosted cascade of simple features. In *Proc. CVPR*, I–511–I–518.