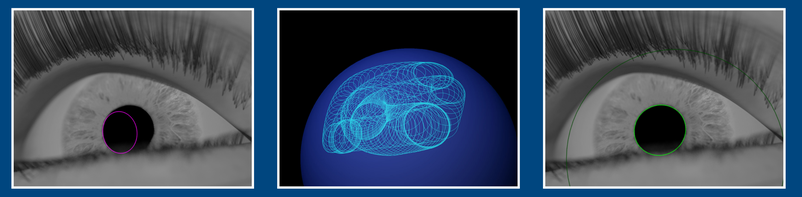
Automatic Temporal Eye Model Fitting

Lech Świrski & Neil Dodgson

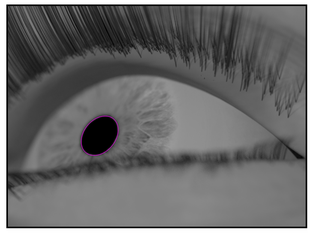


Motivation

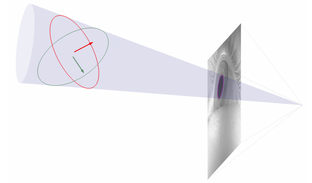
We wanted to do gaze estimation on a head-mounted eye-tracker without requiring the user to do any calibration. We developed a temporal approach which, given a set of eye images taken over some time period, automatically fits and refines a simple 3D eye model. We then find the gaze using this model. Our approach only relies on pupil shape, and does not require glints, controlled lighting, multiple cameras or user input.

Overview

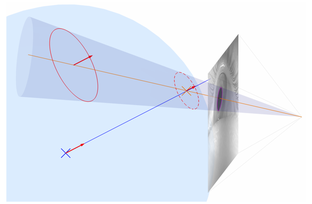
Our approach uses an existing frame-independent pupil detection to find an initial 2D pupil location estimate. We individually unproject these, and combine them to initialise a 3D model. We then refine the 3D model by optimising the eye position, gaze angle and pupil size to best fit the original image data.



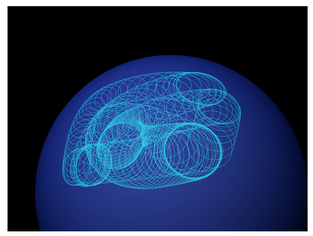
Initial pupil detection



Pupil ellipse unprojection



Eye model initialisation



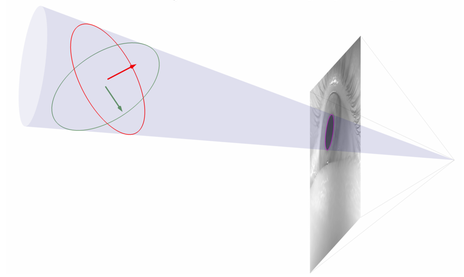
Model optimisation

Pupil Detection

We use our existing pupil detector which finds a best fit for a pupil ellipse calculated from image edges in the pupil region

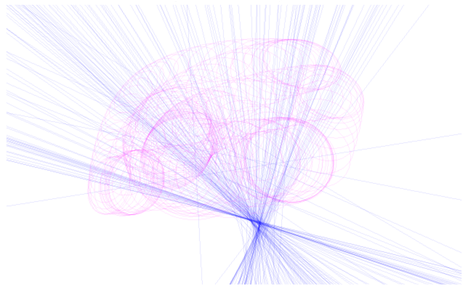
Pupil Ellipse Unprojection

Initially, we individually unproject each 2D pupil ellipse into a fixed-radius circle in 3D, by finding circular intersections of a cone going through the ellipse. This gives two possible circles, symmetric about the major axis of the pupil ellipse.



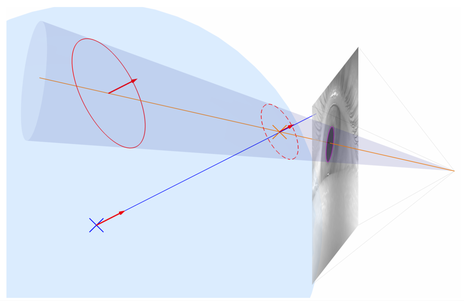
Eye Centre initialisation

We intersect the minor radii of the pupil ellipses to find a projected eye centre, and unproject that at a fixed distance. We then resolve the two-circle ambiguity for each pupil by picking the circle whose normal points away from the centre



Eye Radius initialisation

We know that the pupil circles all lie tangent to a sphere, so the eye radius is the distance from eye centre to pupil centre along the pupil normal. For each pupil, we find this distance by intersecting the pupil centre projection line with a line parallel to the pupil normal from eye centre. The eye radius is initialised as the average of these distances, and the pupils circles are re-projected to lie tangent to the surface of the eye sphere.

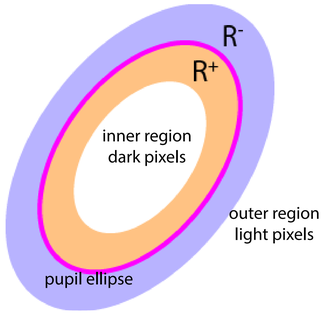


Optimisation

The previous steps only give an initial estimate of the eye and pupil position. We refine this by parametrising the eye position and pupil positions on the eye, and projecting the pupil circles back as ellipses onto the eye images. We want these ellipses to best fit the image data, while still constraining the pupil circles to all lie on the same sphere. We considered two metrics for calculating how well the ellipses fit the images.

Pupil-Contrast Optimisation

We know that the inside of the pupil is dark, and the outside is light. Therefore, for a pupil ellipse, we consider pixels within a certain distance inside and outside of the boundary.

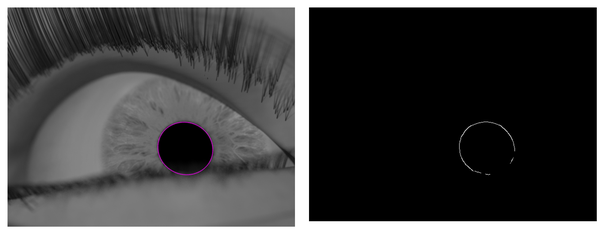


The inner and outer regions of the pupil ellipse, where we want the inner region pixels to be dark and the outer region pixels to be light

We want the pixels inside to be dark, and the pixels outside to be light, so we use gradient descent to maximise the difference of the mean value of the outer and inner regions for each eye image. This maximises contrast along the ellipse boundary

Pupil Edge Distance Optimisation

Our pupil detection gives, as output, the edge pixels which were used to find the pupil ellipse.

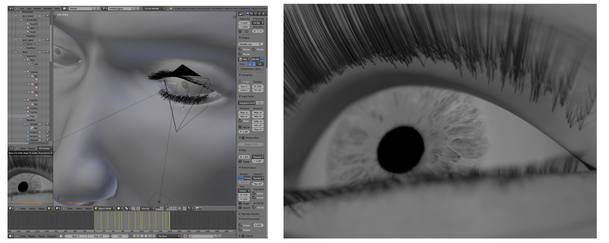


The pupil ellipse detected by our pupil tracker, and the corresponding edge pixels which were used to calculate that ellipse.

We assume that these edge pixels are correct, and we therefore want our projected pupils to be close to these edge pixels. We use a least-squares solver over all the eye images, minimising the sum of squared distances of the edge pixels from the corresponding projected pupil ellipse.

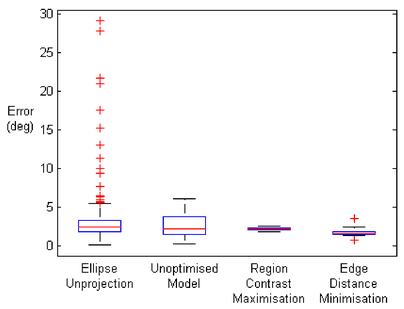
Evaluation

We evaluated our algorithm on a novel realistic rendered dataset. We used an existing model of a head from an online database, added eyelashes, and retextured it to approximate the appearance of an eye under IR illumination. We then rendered this using Blender's path-tracing. This model gives us full control over eye gaze, pupil radius, camera and illumination, and we could therefore use it to obtain an absolute ground truth.

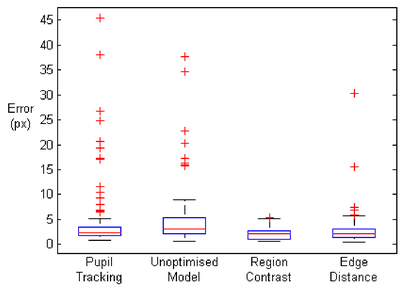


The head model in Blender, with visible camera and two spot light sources, and the resulting rendered eye image

We found that our algorithm gives much higher gaze accuracy than naive unprojection, but lower accuracy than calibrated approaches. We believe that we are approaching the limit of gaze accuracy obtainable from a pupil ellipse, and that further improvements must use additional data, such as calibration.



The angular gaze error of the naive pupil ellipse unprojection, our initial model, and our two optimisations



The ellipse reprojection error of the initial pupil tracker input, our initial model, and our two optimisations

EPSRC
Engineering and Physical Sciences Research Council

http://www.cl.cam.ac.uk/research/rainbow/projects/eyemodelfit

UNIVERSITY OF CAMBRIDGE
Computer Laboratory
RAINBOW RESEARCH GROUP