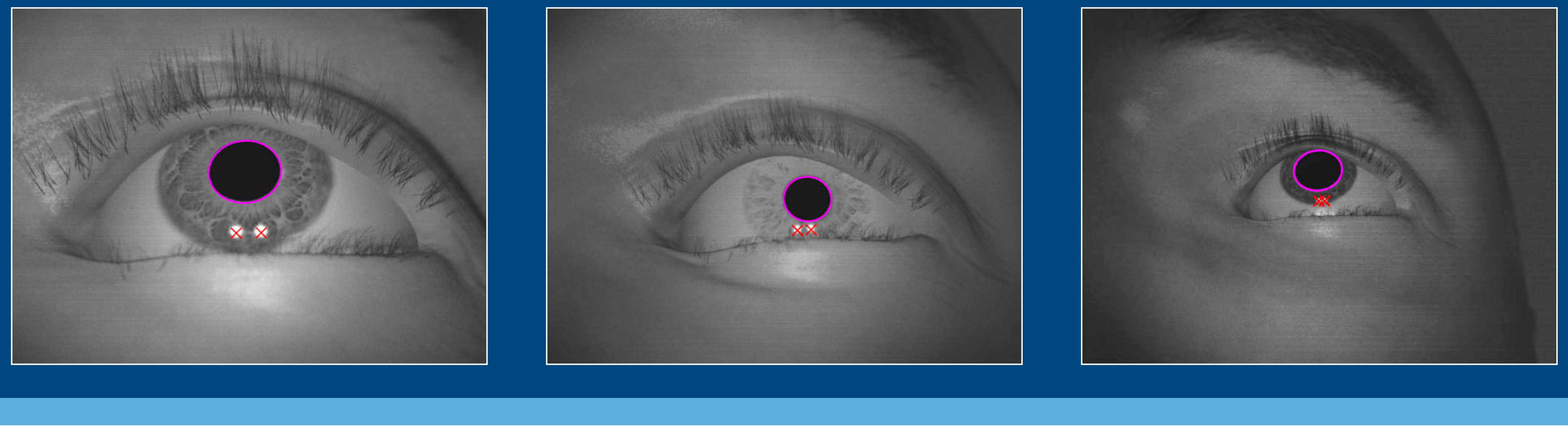
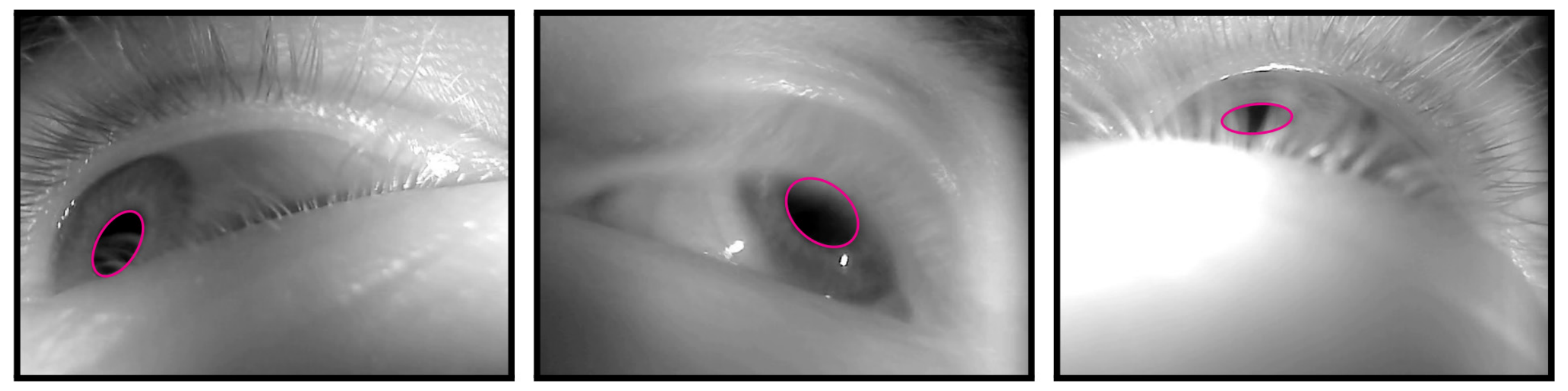
Rendering ground truth eye images

Lech Świrski & Neil Dodgson



Motivation

When evaluating eye tracking algorithms, a recurring issue is what metric to use and what data to compare against. For a proper evaluation, the results of the algorithms should be compared against a ground truth dataset, however it is not obvious where this data should come from, or how true it really is.



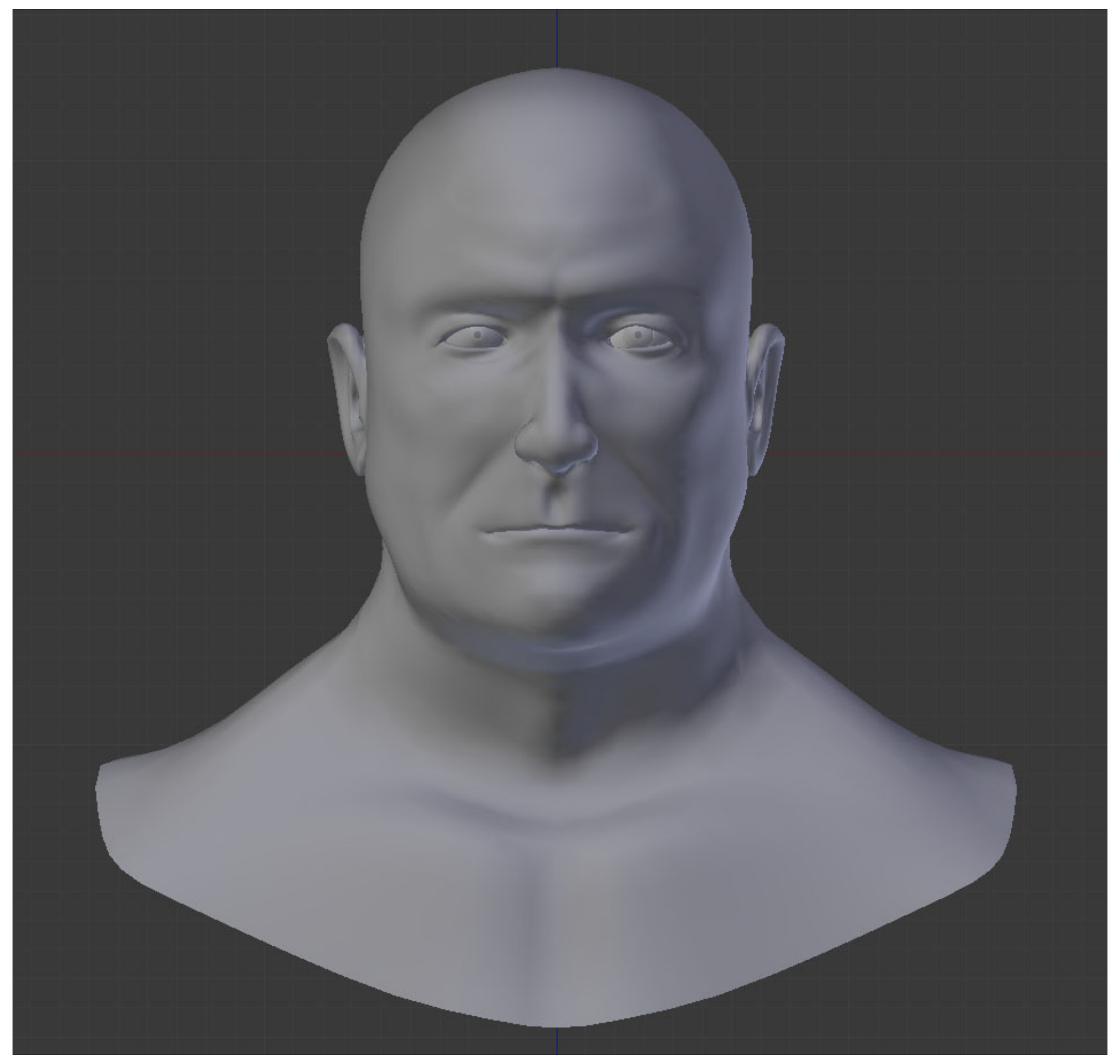
A pupil tracking algorithm will output ellipses, but it is not obvious how to evaluate this output in an easy and unbiased way.

We wanted to create a system for obtaining ground truth data which can produce a large amount of high-quality ground-truth data, which allows full control over the parameters of the system, and which doesn't require much effort to use.

Using real images does not offer enough control over the system, and requires manual labelling, which is time consuming and may include human bias. However, current simulation approaches, such as artificial eyes or rendered eyes, do not model the parts of the head surrounding the eye, such as the eyelids, eyelashes, and skin, which limits their use when evaluating image processing algorithms.

Our solution

To create a more realistic simulation, we built a head model which includes the eye and surrounding facial structure. We then used Blender to render this model with photorealistic lighting, reflections and refractions.

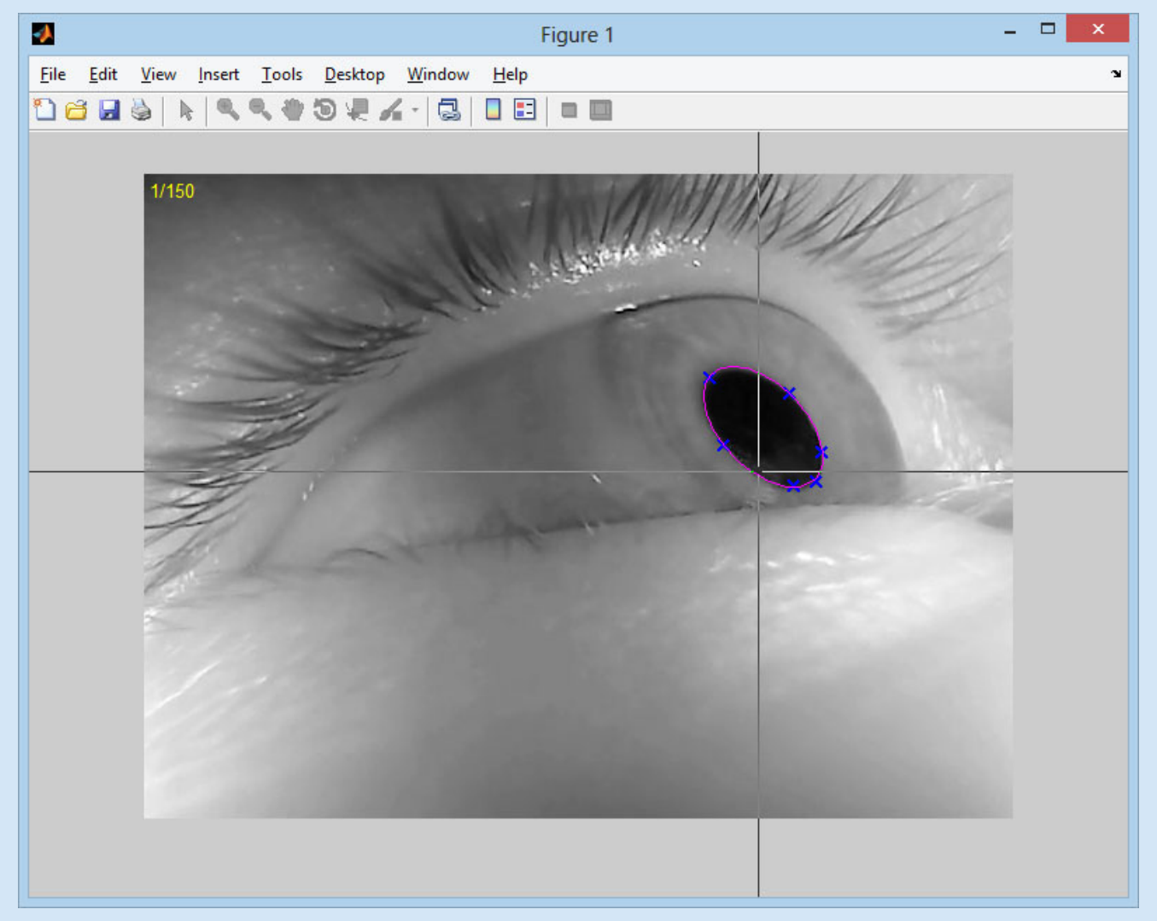


Our head model, which includes eyes and surrounding facial features.

Using a modern computer graphics system allows us to create realistic image input for a gaze estimator, while still allowing the user to have full control over parameters such as the gaze direction, camera position, and lighting.

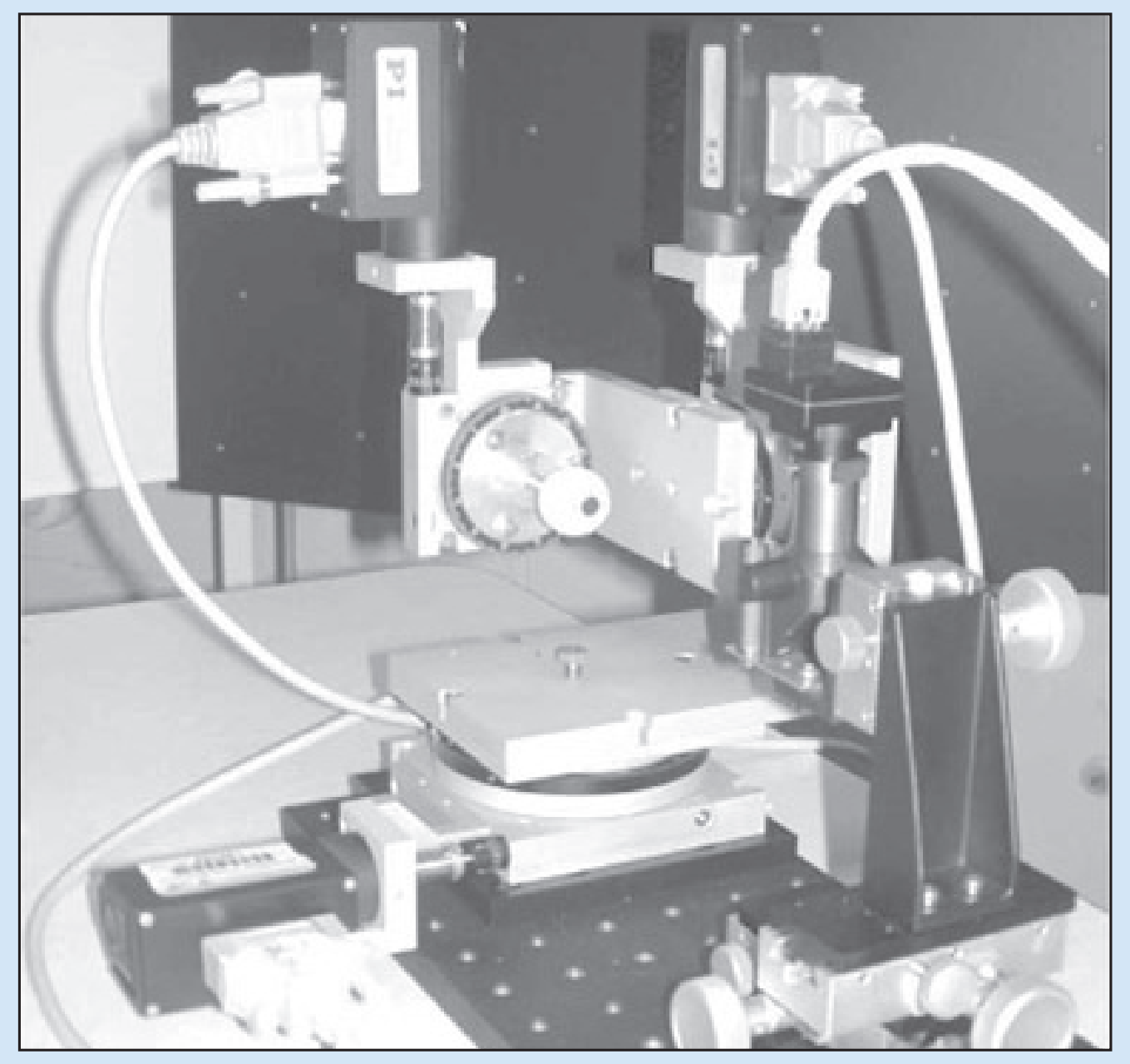
Existing Work

There have been many approaches to obtaining ground-truth data for eye tracker evaluation, however these can mostly be categorised into one of the following four categories, each with their own advantages and disadvantages:



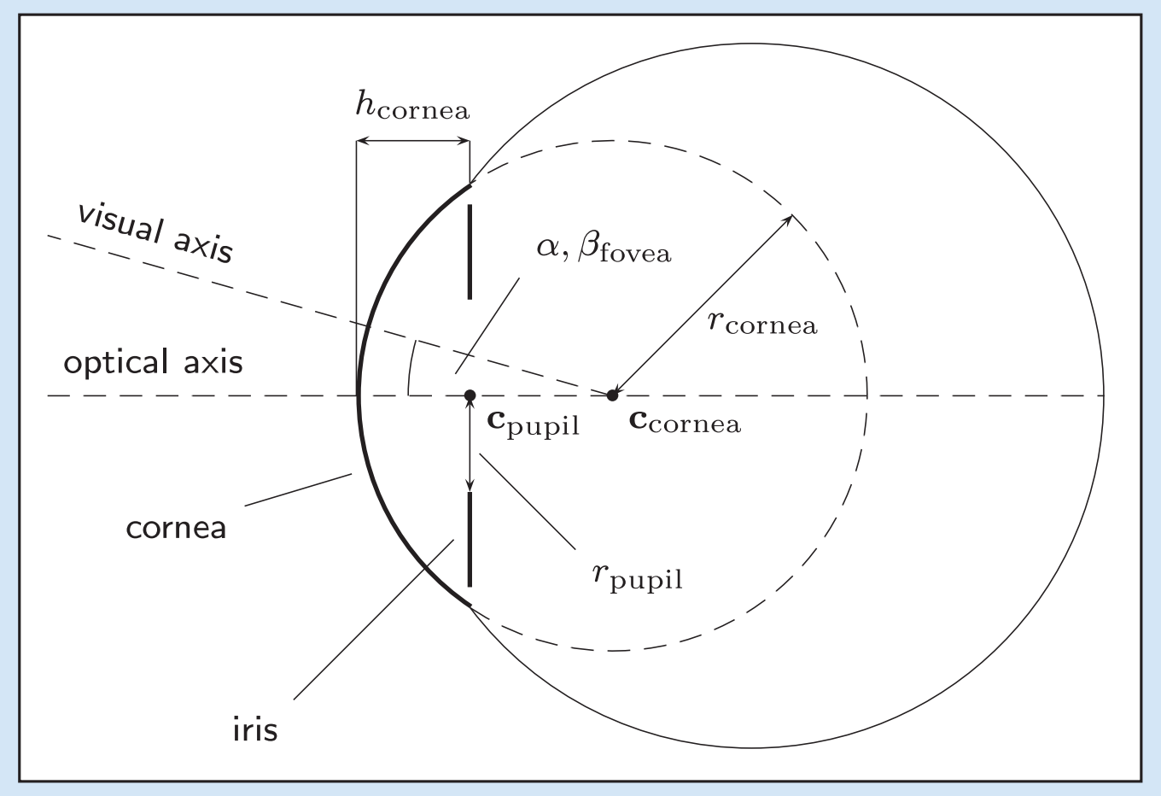
Manual labelling

+ Uses images of real eyes
– Takes time and effort to collect, may include labeller's bias



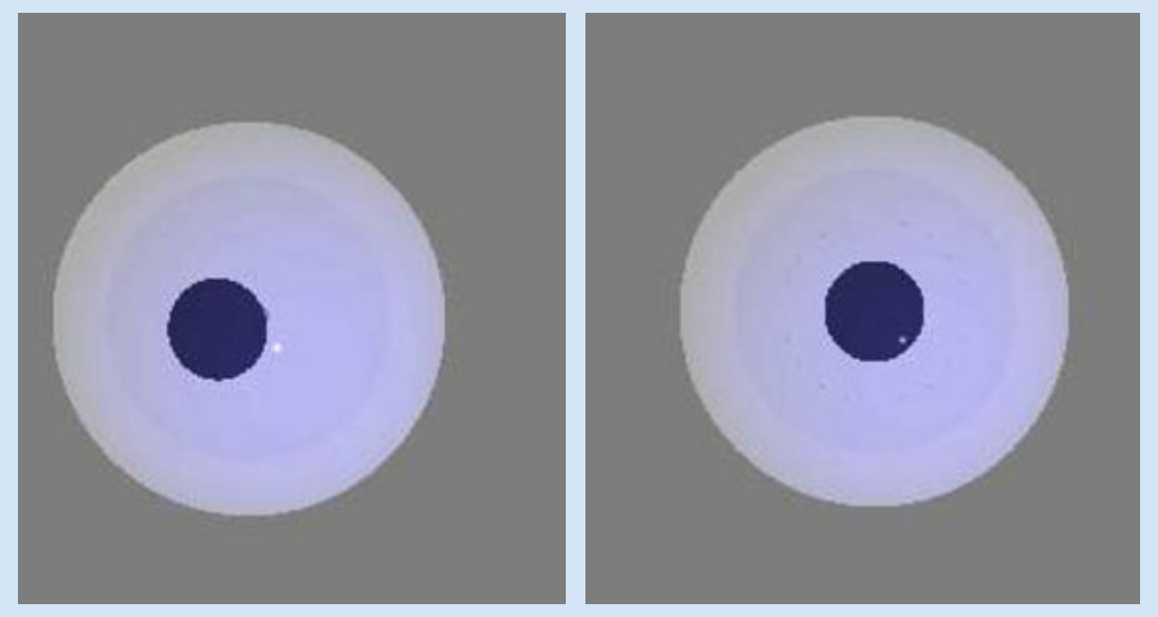
Artificial eyes

+ Control over position, real camera
– Specialised equipement, difficult to make realistic



Geometric models

+ Full control
– Doesn't help evaluate image-analysis algorithms

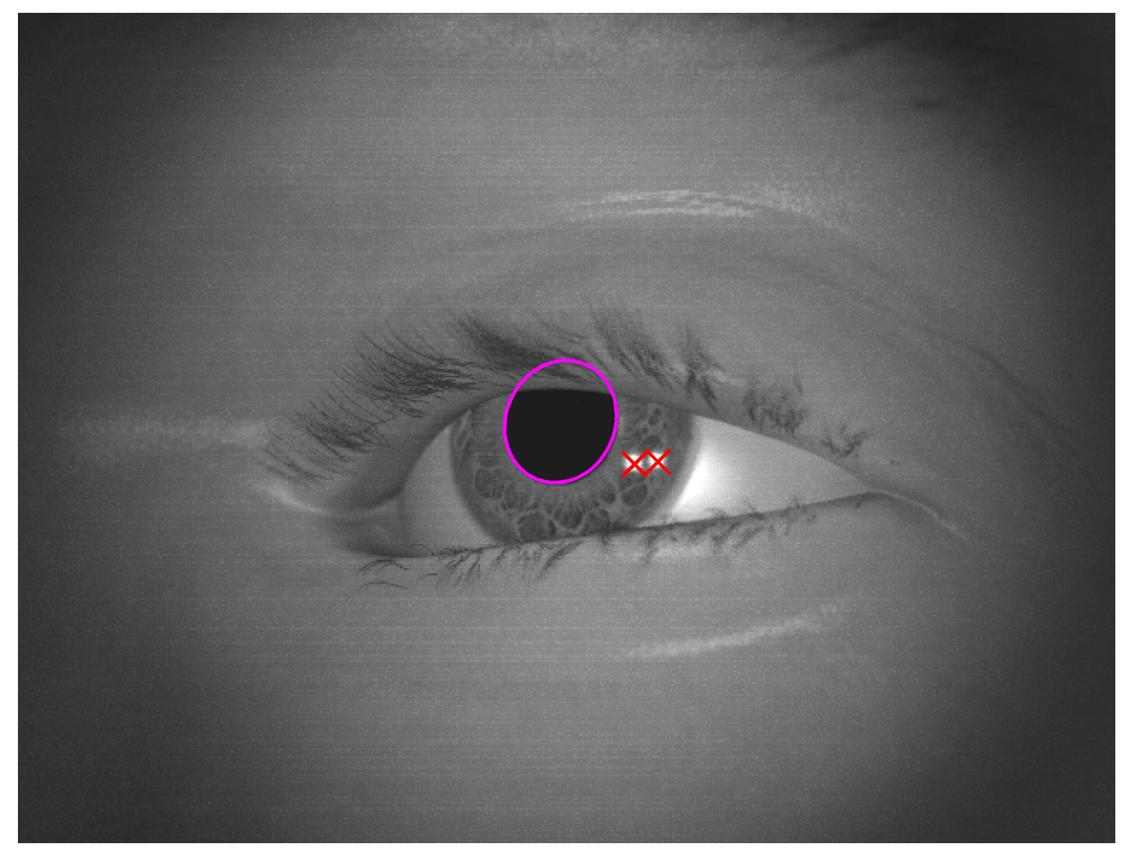


Rendered eyes

+ Full control
– Difficult to make realistic

Scripting

We provide a simple Python scripting interface to our system, which allows the user to easily render large amounts of ground-truth data with varying parameters with minimal effort. All the images in this poster were generated using this interface.



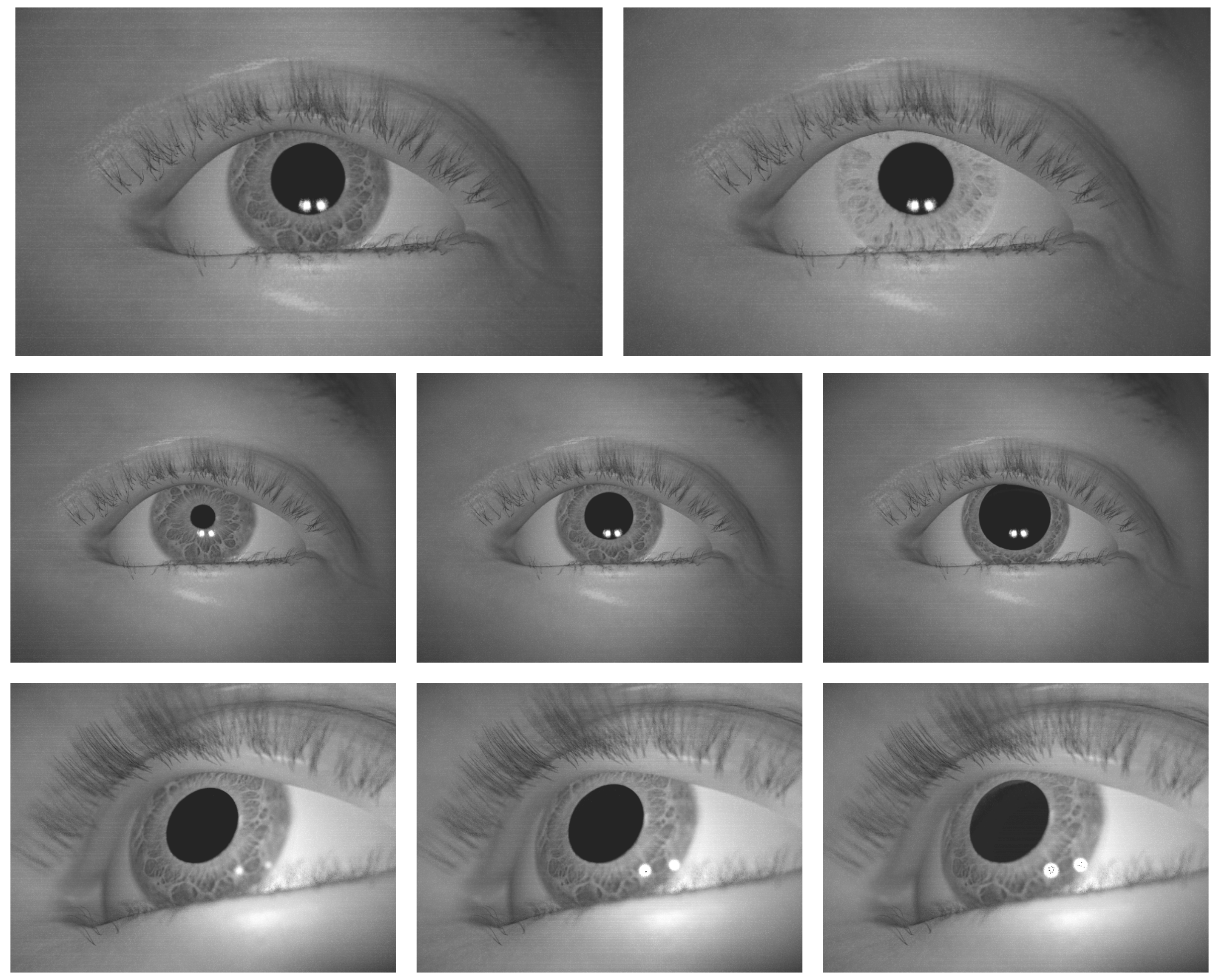
```
import eyemodel

with eyemodel.Renderer() as r:
    r.eye_target = [0, -1000, 0]
    r.camera_position = [20, -50, -10]
    r.camera_target = [0, -r.eye_radius, 0]
    r.eye_closedness = 0.2

    r.lights = [
        eyemodel.Light(
            location = [15, -50, -10],
            target = r.camera_target)
        eyemodel.Light(
            location = [25, -50, -10],
            target = r.camera_target)
    ]

    r.render("example.png")
```

An image of a partially closed eye with two lights shining on it, rendered by our system (left) and the code used to render it (right)

Parameters

Our interface gives the user control over many parameters:

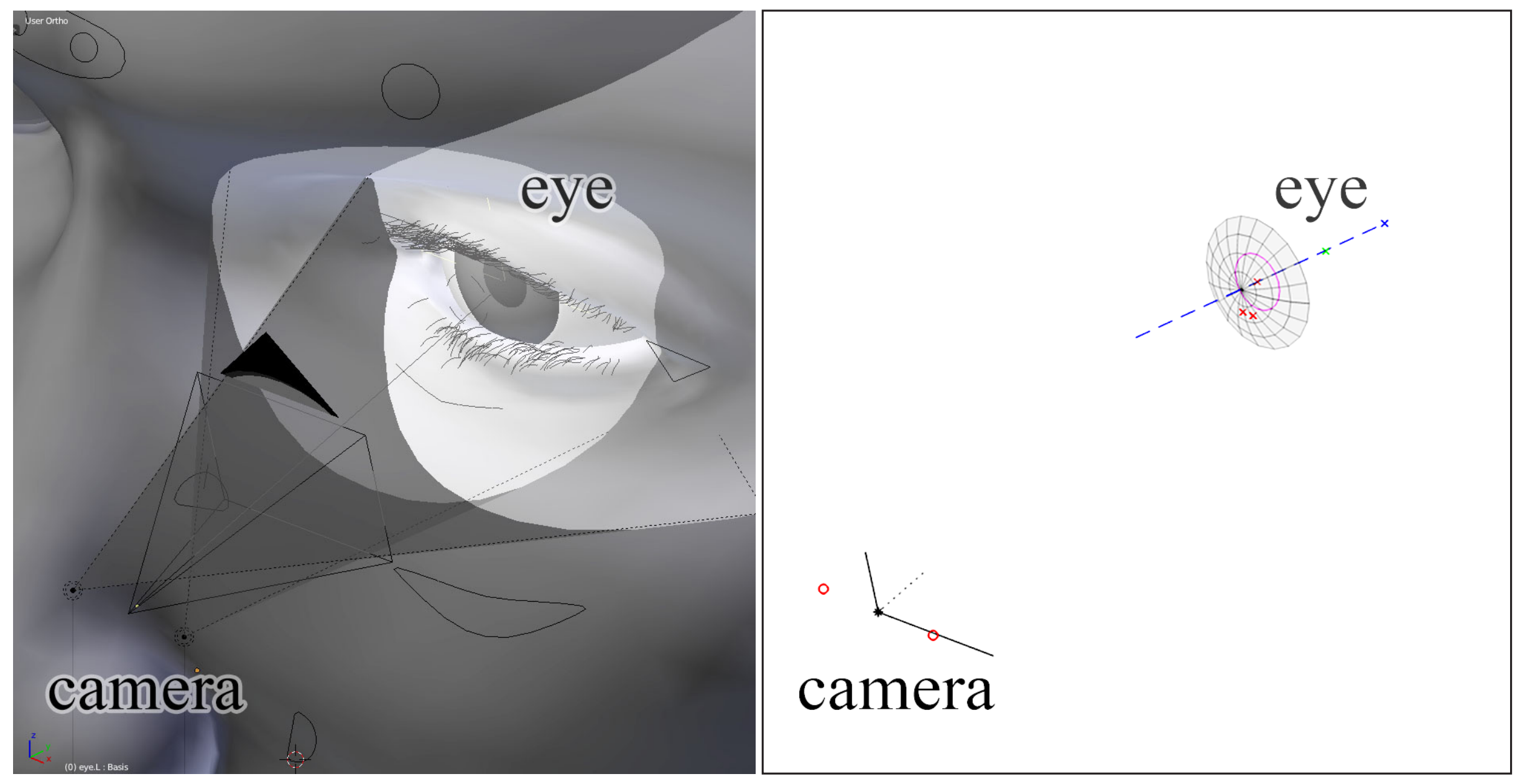
- Pupil size
- Cornea refractive index
- Eye radius, position and orientation
- Eyelid position (how closed the eye is)
- Iris texture
- Cornea refractive index
- Pupil dilation
- Camera position and orientation
- Camera focal length (in pixels)
- Camera F-number (for depth of field)
- Image size (in pixels)
- Lights (position, orientation, view angle, size and intensity)



Images created by our system showing varying parameters, such as iris texture, eyelid position, pupil radius and corneal index of refraction.

Ground truth calculation

Our model uses similar parameters to Böhme et al.[1], which means that we can pass our parameters into their MATLAB simulation framework.

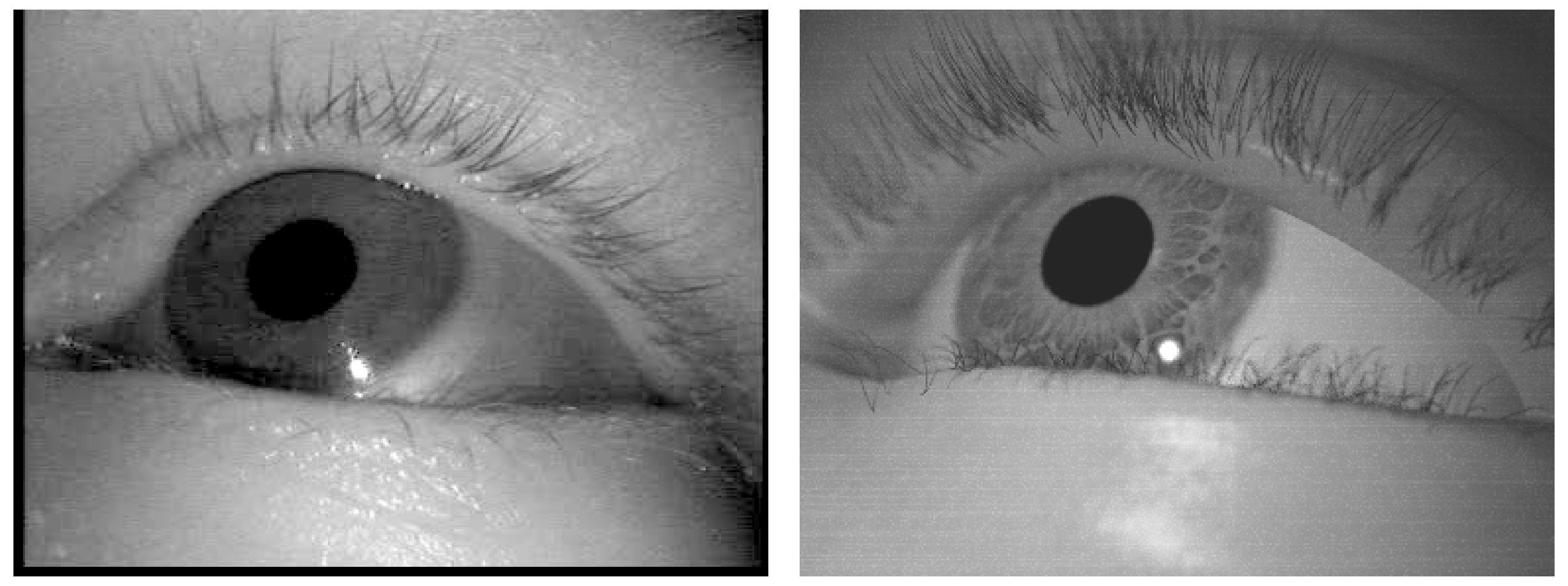


Our model (left), and the same parameters in Böhme et al.'s MATLAB simulation framework (right)

Using the same parameters as Böhme et al. allows us to use an existing, proven geometric model, to provide subpixel-accurate locations for the pupil contour and glints as the ground truth data for our rendered images. All image annotations in this poster were calculated using Böhme et al.'s simulation framework, with the parameters that were passed into our system.

Discussion

We have created a system which creates realistic, synthetic ground truth images for eye tracker evaluation, which is compatible with an existing geometric simulation framework.



An image of a real eye (left), and an image rendered using our model (right). The model parameters were set manually to match the real image.

Our system gives the user full control over many parameters which are important when evaluating eye tracking algorithms, but achieves a level of realism beyond that of previous simulation work. While we do not claim to be indistinguishable from real images, we believe that we have achieved a level of realism sufficient for convincing evaluation of image-based algorithms.

References

[1] Böhme, M., Dorr, M., Graw, M., Martinetz, T., and Barth, E. 2008. A Software Framework for Simulating Eye Trackers. In Proc. ETRA

EPSRC
Engineering and Physical Sciences Research Council

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

UNIVERSITY OF CAMBRIDGE
Computer Laboratory
RAINBOW RESEARCH GROUP