Summary of Eye Gaze Tracking Using an RGBD Compared to RGB

Non-IR methods can be separated to:

1. Appearance based
2. Iris based
3. **Face-model based**

Appearance based

Build a regressor that maps eye appearance to screen coordinates

These assume appearance **only** changes due to pupil movement

Iris based

Detect iris through ellipse fitting, then calculates its normal to approximate gaze.

These methods are usually inaccurate due to iris occlusion, noise, and reflections.

**Face-model based**

**Locates facial landmarks, and their 3D locations (via stereo camera/generic face model). Then eyeball center location is estimated and refined through calibration.**

**These methods are robust to head movement, but small errors in 3D landmarks will cause large errors in gaze estimation.**

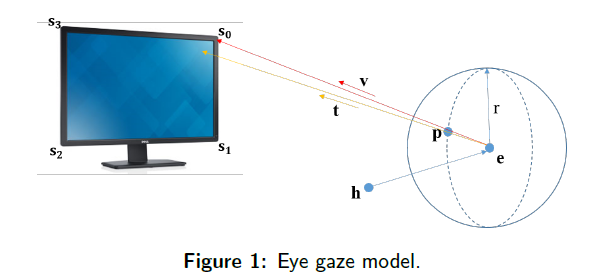
For Face-model based methods, two ways to extract depth information:

1. Depth camera
2. By minimizing projection error of person-specific 3D face model and tracked 2D landmarks.

The paper analyses Face-model methods, and:

1. A comparison of the two methods mentioned above
2. A method to measure lower bound of gaze error
3. Effects of pupil, and facial feature detection, on gaze estimation

**Overview**



P – pupil center

E – eyeball center

T – optical axis

V – visual axis == gaze direction

Alpha – horizontal diff. of V vs. T

Beta – vertical diff. of V vs. T

H – head coordinate system

The following is known for a user (through calibration): eyeball center, eyeball radius, alpha and beta.

After calibration, gaze direction can be computed:

1. Transfer eyeball center from head coordinate to world coordinate (see head pose section)
2. Calculate optical axis == a normalized vector from eyeball center to pupil center (see iris detection)
3. Remove head rotation factor
4. Calculate visual axis (gaze direction) by rotating alpha and beta deg.

**System calibration**

Transform depth camera, color camera, and monitor screen coordinate system into world coordinate system (color camera system?)

Depth to camera: described in [11]

Screen to camera: using auxiliary camera and calibration pattern (like 9 dots) [12]

**Head pose estimation**

Calculated using one of two methods: using depth camera, or the boring way

Depth camera way

1. Track 49 facial landmarks on RGB (using supervised descent)
2. Read the corresponding 3D coordinates from depth camera
3. To track head pose, construct 3D face model
4. Head pose is measured relative to reference model – Head rotation matrix and translation vector are obtained.

If depth data is missing for a point: perform local neighborhood search.

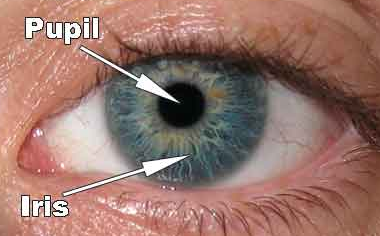
Remove points with fitting error more than two STD from mean, and return to minimization problem (4).

**Iris Detection**

Without IR data (which Realsense should have), iris detection is replaced with pupil detection.

Iris modelled as ellipse. Pupil center inferred from center of ellipse.

1. Crop eye region
2. Perform histogram equalization to increase contrast
3. Construct binary image using thresholding on mean pixel
4. Run Connected component analysis to remove reflections
5. Run Gaussian blur
6. “Emit” 30 rays (horizontally, to ignore up down eyelids)
7. Point with highest gradient on ray is considered to be on the ellipse
8. 3D pupil center can be then calculated



**Personal parameter calibration**

Required to compute eyeball center, alpha and beta.

User is asked to look at 9 predefined points on the screen. Initial parameters are set to human average.

Calibration is achieved by minimizing sum of angles between predicted gaze direction and ground truth.

COBYLA algorithm is used [7]

**Experiments**

Iris detection is racist

From images - Occluding eyelids (or when the iris is at the extreme)