Mobile Stereoscopic Eye Tracking

Iowa State University ECpE Senior Design Team May 13-20
Justin Derby, Kristoffer Scott, Tyler Burnham, Arjay Vander Velden, Will Bryan, Scott Connell
Advisor: Daji Qiao
Client: Stephen Gilbert, VRAC

Introduction

Project Eyeris is an embedded, mobile, real-time eye tracking system. The goal of this project is to create a viable solution to handle 3D environments as well as real-time video streaming to observers. There is no current product that handles both of these requirements. Our solution is primarily to be used for virtual reality applications at Iowa State's Virtual Reality Application Center. Project Eyeris is the work of a six member team including an Electrical Engineer, Software Engineers and Computer Engineers over the course of two full semesters.

Design Requirements

- Track both eyes with minimal error (less than 1.5 degrees)
- Can be used with active stereo shutter glasses.
- All data and outward-facing video collected from the glasses shall be transmitted wirelessly and will be able to be viewed in real time
- No more than 10 second delay
- Unobtrusive and non-invasive
- Glasses: no more than 2 pounds and no wider than 10 inches
- Backpack: no more than 5 pounds
- Viewer's environment: 720p color video, field of view of 56 x 40 degrees
- Eye tracking: 30 frames per second

Workflow/Components

Cameras

Logitech C270
Outward-facing camera records what the user is looking at.

Microsoft HD 6000
Inward-facing IR cameras to capture the motion of each eye.

Algorithm Board

Gumstix DuoVero
The cameras are connected to an embedded board that runs an ellipse-fitting algorithm, which converts an image of an eye into data useful for the eye-tracking algorithm on the server. This data is sent over the serial port to the main board.

Main Board

Pandaboard ES
The main embedded board combines the outward-facing camera and the eye data and streams it over Wifi to the server.

Server

VRAC Cluster
The server analyzes the data from both eyes, and computes the direction and depth of where the user is looking. The server then overlays this information on the outward-facing video, and streams it to all connected clients.

Clients

Observers
The clients receive the video from the server with the painted point of focus from the outward-facing camera. They can also send control messages to fine-tune the algorithms.

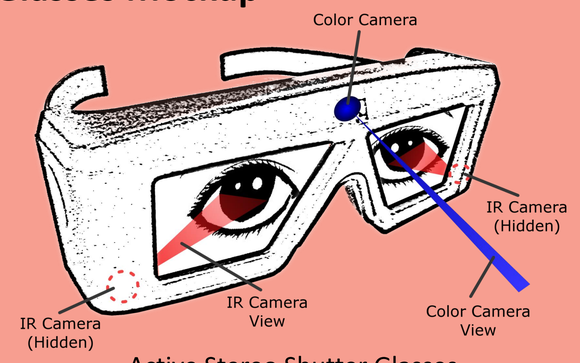
Technical Details

Programming languages used:
- C/C++
- Libraries used:
- Boost
- OpenCV
- Intel Threaded Building Blocks
- GStreamer
- Qt Framework

Development tools used:
- Netbeans C/C++
- Qemu
- Qt Creator

Environments used:
- Ubuntu (GNU/Linux)

Glasses Mockup

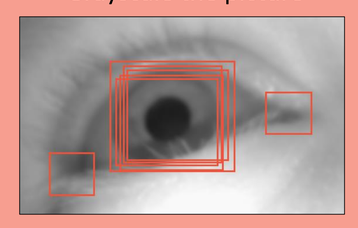


Active Stereo Shutter Glasses

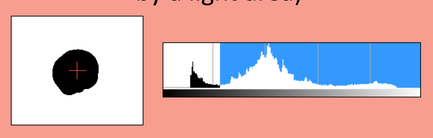
Pupil Fitting Algorithm

Grabs a picture from the inward-facing webcam

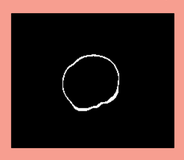
Grayscale the picture



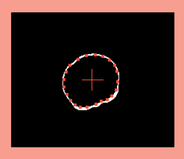
Use Haar-like feature detection to find the general area of the pupil (which is a dark area surrounded by a light area)



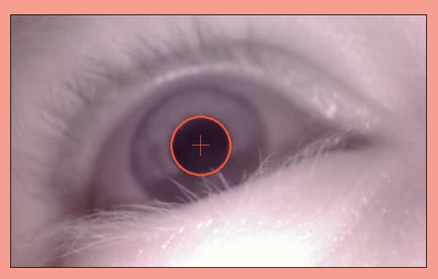
Threshold the image to get the general area of the pupil. The center weight of the black area is the center of the pupil



Manipulate the image to get rid of eye lashes, etc., and do edge-detection on the pupil



Find points from the center to the edges and perform an ellipse-fitting algorithm



The ellipse has been found and used for any other calculations (e.g. gaze, distance)

Testing

- Testing was done on local machines, and also on the hardware for the project
- We have tested our software using integration testing, and system testing. Independent modules were tested as a single unit, then tested as a whole with many other modules.