James Laskey
CSCI1970-HCI with Prof. Jeff Huang

WebGazer.js - A Webcam Real Time Eye Tracking Platform

The growing field of eye tracking is enabling new quality of life and analysis tools. Eye tracking assists motor impaired individuals, enables sophisticated user interaction capabilities, and captures more detailed attention statistics. Current state of the art eye tracking solutions involve expensive and specialized hardware. This prevents large scale adoption and usage of the technology outside of lab environments. To alleviate this issue, I worked with Alexandra Papoutsaki, a PhD student on a new web based framework called WebGazer.js. This library makes use of widespread consumer webcams to drive its predictions. Written in javascript, WebGazer.js runs entirely in the client's browser without needing to send any data to a server. This makes the tool ideal for user studies on Amazon Mechanical Turk where the user may not agree to a full recording of the interactions. We utilized this setup in our paper, *WebGazer: Scalable Eye Tracking Using User Interactions*[1], by sending studies to a large body of Amazon Mechanical Turk users.

WebGazer.js employs a well structured design which separates out various concerns intrinsic to eye tracking. The library splits out the face and eye tracking components into a 'tracker' module which detects and crops out the eyes of the participant. Because this operates as a separate module implementing a common interface, the tracker can be swapped during operation. This allows different tradeoffs in performance and accuracy to be made at the discretion of developers using WebGazer.js. Once the eye components are cropped, WebGazer.js passes this data to a 'regression' module which will make the x and y screen coordinate predictions. The currently implemented regression modules utilize the machine learning ridge regression algorithm. To train ridge regression we capture mouse click and mouse movement events from the browser. These movements are known to be strongly correlated with where the user is looking on the screen. Thus they make good enough ground truth training examples. From just a few of these training examples, ridge regression is able to begin making informed predictions about where the user is looking regardless of mouse location. The regression module is also swappable during operation. In future work this could allow for different regression models depending on the number of training examples collected. Since these modules are well separated from the rest of WebGazer.js we hope that it will be simple to extend the library with new modules that increase accuracy or improve performance. As an example of this, I wrote a new regression module that moves the regression training to a web worker greatly increasing speed.

In addition to building WebGazer.js, we have run several studies that employ the library to understand how users interact with search results. In the upcoming paper [1] we examined ran three studies which examined the differences eye tracking data could illuminate versus mouse only data. These studies required setting up a webserver to feed hundreds of users

[1] WebGazer: Scalable Eye Tracking Using User Interactions will be published and available at the IJCAI 2016 conference

specific search pages as well as writing code to log and extract all gaze and mouse data. The resulting data demonstrated a reasonably expected amount of similarity, but also some differences. These results are detailed in the paper.

To promote WebGazer.js, we built a website[2] showcasing the library and its capabilities. The source code for the project will also be made available on github[3]. We hope that the program will be useful in a variety of domains where cheap and web scale gaze tracking can provide new, exciting data.

[2] https://webgazer.cs.brown.edu
[3] https://github.com/brownhci/WebGazer