Group 27

Software Evaluation of Reading Aptitude using Consumer-Grade Gaze Tracking Hardware

Preliminary Report – September 19th, 2014

Prepared By:

Paras Vora
Maeve Woeltje
David Young

Prepared For:

Dennis Barbour, M.D. Ph.D.
Jonathan Silva, Ph.D.
Wandhi Zhu, Ph.D. Student

Washington University in St. Louis Department of Biomedical Engineering
BME 401: Biomedical Engineering Design

Table of Contents

1.0 Background

Even in an era of increasing technology dependence, reading remains an essential daily task. It is critical in education and the work place, where knowledge is transmitted most often through text. In fact, a child's future economic success can be predicted by their reading proficiency as early as the fourth grade, and there is clear indication that students from low-income families have poorer reading abilities[1]. This shows a clear need for a cost-effective solution capable of monitoring how children are reading, in order to provide schools and caregivers the information they need to help a child develop into a successful reader.

While a plethora of reading assessment tests exist, they all include interacting with a subject and usually are given by trained professionals. Parents, teachers, students, and other interested consumers could benefit from a cost-effective, consumer friendly system of analyzing reading patterns, which can be used to analyze how well a user read a piece of text. This sort of analysis can be done through the use of eye tracking equipment alone. Modern commercial eye-tracking systems measure the point-of-gaze (POG) through the Pupil Center Corneal Reflection (PCCR) method[2]. Generally, an infrared camera is attached to a computer containing the appropriate image processing software, and infrared light from an LED near the camera is directed to the eye. This creates a corneal reflection that manifests itself as a "glint"[3]. The software can then create a vector between the pupil and reflection, and with further trigonometric calculation, measure POG.

POG could provide information useful in characterizing reading performance. For example, with POG data it is possible to determine the amount time the eyes are locked on one location, indicated by a relative lack of eye movement. This eye behavior is known as a fixation and tends to last around 225-250 milliseconds. Generally, humans perceive three to four letter spaces to the left and fourteen to fifteen letter spaces to the right of the point of fixation[4]. It is during these periods of fixation that a reader is absorbing information. Fixation duration can indicate difficulties in information extraction or how engaging an object is to the user[5]. Analyzing a reader's fixation provides a great deal of information on how well they are focusing on and understanding a piece of text.

Another prominent feature of eye movement is the saccade. Saccades are rapid eye movements between two points of fixation, which last about 25-60 milliseconds and travel seven to nine letter spaces between each fixation during reading. The distance between saccades indicates more meaningful cues amongst objects present[6]. Saccades are not always forward and linear. In fact, regressive saccades (backtracking) occur quite frequently and suggest processing (encoding) difficulty of text[7]. The overall pattern of saccadic movement indicates how a reader interacts with a piece of text.

Reading occurs via a combination of fixations and saccades. Through the positional measurements provided by an eye tracker, it is possible to determine a reader's fixation and saccade data. A variety of inferences can be drawn from these metrics, including a user's gaze pattern or behavior and the complexity of the material being read. Many of these inferences apply directly to reading skills, and thus can provide feedback on how an individual reads.

2.0 Need and Scope of Project

The current age of e-books, tablets and digital media allows the average person to read more from a screen than from a piece of paper. Simultaneously, the willingness of consumers to use software for training and self-assessment is increasing. Through eye-tracking technology, it will be possible to follow a person's gaze as they read on digital devices, and subsequently utilize the output data for analytics.

2.1 Project Need

There is a current need for a consumer grade system to provide a user with information about how they read. Reading analyses using eye-tracking hardware have been conducted in clinical settings for research purposes. However, there is no intuitive, affordable software that provides feedback on a subject's reading thoroughness to a teacher, parent, student, employer, or other interested individual. Thoroughness feedback, in this sense, is an analysis of reading patterns including notes on changes in speed, identification of apparent distraction, and a marking of segments where a reader skimmed, skipped, and re-read text. A software suite designed to measure performance through an analysis of reading patterns has the potential to improve the education process for interested consumers. Teachers would be able to identify whether or not children are actually reading their assignments. Employers could tell if employees thoroughly read a memo, set of instructions or contract. Furthermore, the thoroughness analysis could allow the interested party to identify portions of a document that readers are struggling with. Teachers, parents, and students would be able to track the improvement of a child's reading performance. With a consumer size user population, gaze tracking data could be collected in greater quantities than clinical trials, to be used for big data analysis. An affordable gaze tracking solution designed specifically for reading would address the need for personal diagnostics to break into the consumer reading market.

2.2 Project Scope

This project aims to create an affordable consumer-friendly software toolset that will work in conjunction with existing gaze tracking hardware to quantify and evaluate reading patterns as well as store user session histories. Using experimental metrics, it will provide instant feedback to the user outlining their reading performance. Reading performance will be defined by a user's reading speed, a breakdown of document sections skipped or skimmed, possible indications of distraction or boredom, and a list of sections or vocabulary words where the user somehow struggled (as indicated by prolonged fixation and redundant reading). This feedback will be presented in an intuitive package with the potential for a user to compare results to previous sessions or global averages. A user's progress history will be stored for recall at a later point in time, and potential use in big data analysis. This project will not include the design of any hardware, as there already exist eye-tracking devices on the market. The project will not serve to replace any existing clinical-grade diagnostics.

3.0 Design Considerations

Note that the quantitative analyses for the minimum required hardware sampling rate as well as the data storage size per sample are shown in Section 5.0: Analysis. These analyses helped define the hardware specifications shown in Section 3.1 below.

3.1 Hardware Specifications (Consumer Grade Eye Tracking)

Cost	< $100 (accessible to consumers)
Setup	Useable at home, external device (for customer's existing laptop or desktop), 1 cord, calibration in under 1 minute
Weight	Lightweight (<1 lb)
Size	Small (< 30cm x 5cm x 5cm)
Physical Connections	USB
Power Source	Powered over USB by Parent Device (attached computer)
Minimum Hardware Data Output	Spatial Gaze Coordinates (both raw and smoothed), Sample Timestamps
Latency	Functionally Real time
Sampling rate	Minimum 4Hz for reading, 12Hz for scanning…. or higher (Nyquist)
Working Distance from Device	1.5-3 feet
Compatability	Windows 7 and Mac OSX drivers

3.2 Software Specifications (Provided Hardware is Already Selected)

Useful software outputs	Heatmaps, Scan Paths and comparisons to known patterns, sentence/word (vocab) lists for skipped or re-read portions of text, time, historical log of trials, data storage friendly formats
Required Prior Training	None, User Intuitive step by step instructions, approachable interface and menu system, child friendly
Calibration	User intuititve step by step instructions and graphical interface
Software Adaptability	Compatible with provided input texts given in .txt file.
	Support for different monitor resolutions and aspect ratios
Safety	Display Backgrounds that reduce eye fatigue and strain
Storage Requirements	Average session data size <5MB
	Program size <500MB
Completion Date	December 9th 2014
Ease of use	Intuitive GUI and menu system

3.3 Preliminary Hardware Selection Criteria

The stated need calls for both software and gaze tracking hardware in a working package. As gaze tracking is not a new technology, something about the solution must be disruptive for it to successfully branch into the consumer market. Eye trackers have been present in clinical and research domains for decades. However clinical grade devices are unwieldy for personal use, exceedingly expensive and often lack simple or intuitive software support. In the last year inexpensive consumer grade devices have emerged from various manufacturers, introducing low cost development platforms for software engineers to explore new uses of eye-tracking. The feature set and performance of these new devices fall short of their more expensive counterparts, however their substantially lower cost means both hobbyists and professionals alike are adopting these products. Similar to the effects of the Nintendo Wii controller, a truly consumer grade eye tracker will spawn a new widely adopted method of computer interfacing. To avoid redundancy and considering limitations in time, aptitude and funding, the design team will outsource the hardware portion of the solution rather than designing an eye tracker.

Although software design will be somewhat universal, and potentially portable to more than one eye tracking unit, it is important to select a single development device that a user is expected to purchase. This will ensure software testing is consistent and representative of the consumer experience. Looking at the range of existing hardware, one can immediately cast out esoteric research or clinical grade devices from a cost standpoint alone. The goal is to create a package that will permit widespread use in a consumer household setting. To accomplish this, it is crucial to select the best performing device that the average consumer can afford: a process that requires careful analysis of the existing devices in the consumer eye tracking market.

Although new companies seem to pop up every day, there are four main competitors controlling the affordable consumer gaze-tracking market. Included in Appendix C is a complete comparison of advertised specifications for these devices. Manufacturer Gazepoint offers a product called the “GP3 desktop eye tracker” along with software suites that can collect data, as well as analyze and generate heap maps from user recordings. Unfortunately these software solutions are proprietary and cost over $1000. Combined with the near $500 price tag of the tracker itself, the Gazepoint solution is impractical for most consumers. Meanwhile, Microsoft has extended the functionality of its original Kinect with the Kinect 2.0 for Windows. Although marketed primarily for positional-tracking, the Kinect 2.0 offers hardware capable of gaze tracking. However, this aspect of the hardware functionality has yet to be fully explored. The previous Kinect was well received in millions of homes, indicating Microsoft knows how to produce a consumer grade product at an affordable price. They may have similar success with the Kinect 2.0 despite the device being more expensive than its predecessor at nearly $200. Manufacturer Tobii has been engineering eye-tracking devices for many years, but has just released its first consumer grade solution: the EyeX. At around $140, the EyeX has been marketed heavily toward the consumer crowd. Tobii aims to bring easy developer environments to programmers creating Windows applications that incorporate the EyeX. Lastly, Eye Tribe is a younger Danish company marketing an affordable consumer grade tracker. The unit is priced at $100, includes intuitive calibration and data acquisition software, user friendly APIs in popular programming languages and support for both Windows and Mac OS X operating systems. To help select from the previously mentioned devices, the team created a list of hardware specifications for the project so the appropriate tracker could easily be selected. This chart can be seen in detail in Section 3.2.

The Pugh chart provided in Appendix B shows how the team weighed the benefits and compromises of each device before selecting the Eye Tribe tracker for its price, existing APIs (Java), and consumer grade performance specs. Once the team selected the Eye Tribe tracker, it was possible to consider a few aspects of the software design process. The programming language Java was selected due to the developer team's shared Java experience, and existing APIs for the EyeTribe tracker. The team also narrowed the code framework for text mapping down to two possible architectures. The first being to code a grid from scratch using a unit dimension equal to the width of one character (in a monospaced font) allowing easy comparison between text coordinates and input gaze coordinates. The second would leverage some of the capabilities of the Oracle® documented TextLayout API to handle the shaping, ordering, and positioning text. The latter would permit more options in displaying text, but the former would make it significantly easier to map and document word hit detection. The final decision on text mapping will require further investigation through experimentation.

4.0 Existing Solutions

Recall the need is recognized to be the following: an affordable consumer grade gaze-tracking solution to intuitive and immediate feedback on the thoroughness of a user's reading session. Investigation indicates there is no available gaze tracking solution designed specifically for reading that takes the form of a consumer friendly, affordable or user intuitive package. Hardware that would permit such an affordable solution has only started emerging in the last year. Because the solution requires the integration of both hardware and software, there is still a great deal of low hanging fruit in the latter realm. A few software products exist with relevant goals, but they fall short of answering the stated need.

Before designing a new solution, it is important to examine all relevant existing solutions. Unfortunately, as is often the case with biotechnology, the existing products are used by a very select few. Although the market is now starting to see affordable eye-tracking hardware, the existing software solutions are kept from consumers because they are exceedingly expensive or proprietary products functionally limited to clinical and research professionals. Gaze pattern databases (from gaze behavior during reading tasks) are not being created from thousands of consumers. Instead such data is being collected from small quantities of disabled individuals or test subjects in research labs. Big data can be the best way to highlight new patterns, especially those that may have been unpredictable. Of the existing software available today, the least expensive products seem to target only marketing or advertising research for small businesses and web developers.

Taking a closer look at a few of the most relevant solutions, of those available, still leaves the stated need without an answer. Probably the most affordable piece of software out today is *Eyeproof* from the company Eye Tribe. Although still in its beta phase, *Eyeproof* isn't actually software one can purchase or download, instead it is a service. The *Eyeproof* service allows customers to upload a picture of a webpage or advertisement, after which Eye Tribe will perform internal data acquisition from users looking at the image. The customer will then receive a heat map overlay on top of the uploaded picture that represents the gaze data from Eye Tribe's testing. The fallbacks of this service are its lack of portability and versatility, the feedback limitations of a single (albeit powerful) graphic, and an intended customer base that doesn't include general consumers. A more powerful piece of software that consumers can purchase is the *Gazepoint Analysis Software* from Gazepoint. This is a $1495 proprietary software suite for use in conjunction with Gazepoint's own $495 GP3 eye tracker. The software can generate heat maps from gaze data, as well as playback videos of scan paths (cursor location). These are still glorified image overlays that do not link dynamically with the viewed medium. The combined price of Gazepoint's solution places it beyond the grasp of most consumers. Last up is the *EyeMetrix* software from Mirametrix for their S2 eye tracker. This proprietary software can calculate portions of total session time spent looking at selectable ads, buttons or sections of websites. Mirametrix provided a quote of $11,500 for the software and the S2 tracker. What all these existing solutions fail to address is the link between viewed medium and gaze pattern as well as analysis or feedback on the scan path beyond a graphical representation.

Some of these concerns have been thought of before, and addressed in patent applications. Few applications have actually been assigned patents and most fail to design a solution short of stating the desired result. No products have been created from these ideas. There are 4 published patent applications in particular that are worth investigating. These publications outline eye-tracker integration with reading text in dynamic ways (with prompted actions or responses). Hubertus Cortenrad described an immersive reading experience using a theoretical ebook with a built in eye-tracker[8]. After mapping the text of a novel to related sensory inputs, such as sounds, the tracker would determine what portion of text a user was reading and trigger the additional functions that provide immersion. For example, if a scene in a novel was on a beach, ocean sounds might be triggered when the user reads that section of text. This idea is clearly targeted at consumers as the described product is a luxury item. A related consumer product was described by Louis Rosenberg, but this time for reading text on any computer screen. The idea was to provide a graphical flag or marker at the location in the text where a user stopped reading, triggered by the user's gaze leaving the monitor[9]. This would enable users to quickly find their previous point of focus and pick up where they left off without searching through or re-reading text. Focusing less on consumer markets and more on liability issues in professional industry, Janice Gobert and Toto Ermal defined an instruction system based on adaptive eye-tracking scaffolding[10].

As the user read through an instructional document, the system would provide flags or notes if a user skipped an instruction or missed critical information. Lastly, Hans Kruse and Brooke Hallowell successfully acquired a patent for cognitive and linguistic assessment using eye tracking[11].Their solution was aimed primarily at clinical or academic research and although particularly relevant for its integration with text, the consumer need did not call for an assessment. Designing accurate assessments would require the participation of psychologists and linguists.

While the above patents cover some of interesting uses of eye tracking while reading text, these publications lack an existing product and also fail to mention process details for designing or creating software. We aim to incorporate some of the concepts described in these patents to engineer a working solution that employs eye tracking to provide more feedback to the average consumer.

5.0 Analysis

5.1 Minimum Refresh Rate

t_{avg} = average time between start point of 2 saccades while reading (ms)

v_{avg} = average reading speed ($\frac{\text{words}}{\text{min}}$)

v_{max} = maximum reading speed ($\frac{\text{words}}{\text{min}}$)

$R_{\text{sampling,avg}}$ = minimum sampling rate to map each average fixation (Hz).

$R_{\text{sampling,max}}$ = minimum sampling rate to map each minimum fixation (Hz).

According to Rubin and Turano, $v_{\text{avg}} =\sim 250\ \frac{\text{words}}{\text{min}}$. Such speeds can also be found by taking the reciprocal of t_{avg} = $\sim$ 240 ms (treating t_{avg} here as the time it takes per atomic reading action, which is approximately a word) and converting to $\frac{\text{words}}{\text{min}}$:

$$t_{\text{avg}} =\sim 240\ \frac{\text{ms}}{\text{word}}$$

$$\frac{1}{t_{\text{avg}}} = \frac{1}{240}\ \frac{\text{words}}{\text{ms}} \cdot \frac{1000\ \text{ms}}{1\ \text{second}} \cdot \frac{60\ \text{seconds}}{1\ \text{minute}} = 250\ \frac{\text{words}}{\text{min}}$$

The article also gave a $v_{\text{max}} =\sim 1600 \frac{\text{words}}{\text{min}}$.

In order to ensure that each fixation is logged during average reading speed, $R_{\text{sampling,avg}}$ must be at the very least equal to v_{avg} in Hz. Similarly for maximum reading speed, $R_{\text{sampling,max}}$ must be at the very least equal to the inverse of v_{max} in Hz.

Thus $R_{\text{sampling,avg}} = (v_{\text{avg}} \frac{\text{words}}{\text{min}}) \cdot (\frac{1}{60} \frac{\text{minutes}}{\text{sec}}) = \frac{250}{60}$ Hz $= 4.17$ Hz

Similarly, $R_{\text{sampling,max}} = 26.67$ Hz

Accounting for potential aliasing, one can use the Nyquist rate double that of the previously calculated sampling rates. This yields a design consideration of:

$$R_{\text{sampling,avg,nyq}} = 2 \cdot R_{\text{sampling,avg}} = 2 \cdot 4.17\ \text{Hz} = 8.34\ \text{Hz}$$

$$R_{\text{sampling,max,nyq}} = 2 \cdot R_{\text{sampling,max}} = 2 \cdot 26.67\ \text{Hz} = 53.34\ \text{Hz}$$

Given that the selected Eye Tribe tracker runs at fixed sampling rates of 30, 40, and 60 Hz and that data storage is trivial, the device will be run at the maximum sampling rate of 60 Hz for all data acquisition.

5.2 Data Storage

For each sample, the EyeTribe hardware sends a myriad of data to the computer. For the purposes of this project, the information that will require storage includes only 7 variables. Four of these variables are simply the raw and smoothed X and Y gaze coordinates given in pixels. The remaining three variables are the sample time stamp in milliseconds, a state variable describing whether the eye is fixated or not and the relative pupil size. Naturally the state of fixation variable will be represented with a boolean data structure, while the pupil size will be represented with a floating point number. The remaining variables will take the form of integers. This yields a total of 5 integers (each 4 bytes in Java), 1 float (4 bytes in Java) and 1 boolean (undetermined size, but will most likley be coded as a single byte for ease of parsing).

Given that the total storage size per sample is...

$(\text{\# of ints})(\text{size of int}) + (\text{\# of float})(\text{size of float}) + (\text{\# of boolean})(\text{size of boolean})$

$= 5 \cdot 4 + 1 \cdot 4 + 1 \cdot 4 = 25$ bytes

After determining the data storage requirements per sample, it is possible to calculate the storage requirements per unit time that the device is tracking gaze using the following formula:

$(\text{storage per sample in} \frac{\text{\# bytes}}{\text{sample}})(\text{sampling rate in Hz})$

For example, at a sampling rate of 60Hz...

$= (25\frac{\text{bytes}}{\text{sample}})(60\frac{\text{samples}}{\text{sec}}) = 1500\frac{\text{bytes}}{\text{sec}} = 1.46$ KiB/s $= 87.6$ KiB/min $= 5.13$ MiB/hour

Similarly, at a sampling rate of 40 Hz...

$= (25\frac{\text{bytes}}{\text{sample}})(40\frac{\text{samples}}{\text{sec}}) = 1000\frac{\text{bytes}}{\text{sec}} = 0.977$ KiB/s $= 58.6$ KiB/min $= 3.43$ MiB/hour

Similarly, at a sampling rate of 30 Hz...

$= (25\frac{\text{bytes}}{\text{sample}})(30\frac{\text{samples}}{\text{sec}}) = 750\frac{\text{bytes}}{\text{sec}} = 0.73$ KiB/s $= 43.8$ KiB/min $= 2.57$ MiB/hour

Given a single software user, sampling at 60Hz and reading for 1 hour a day, the required space to store all data recorded for the entire year is calculated as follows... $(5.13\frac{\text{MiB}}{\text{hour}})(1\frac{\text{hour}}{\text{day}})(365\frac{\text{days}}{\text{year}}) = 1872.45$MiB/year $= 1.83$GiB/year

Noting that the average consumer PC storage capacity today is quoted in TiB, and that the cost of hard disk storage is pennies per GiB and solid state storage is well bellow a dollar per GiB, these storage requirements are more than adequate from a cost and size perspective.

6.0 Organization

6.1 Team Member Responsibilities

To ensure adequate coverage over each distinct part of software design, each team member will have leadership over a specific programming domain, as specified in Figure 6.1.1. The other team members will still assist the leaders in each domain, allowing them to gain experience in every aspect of project development as well as ownership over their domain. In addition, the color-coded Gantt chart in Appendix A indicates the individual tasks required by each team member, including course assignments.

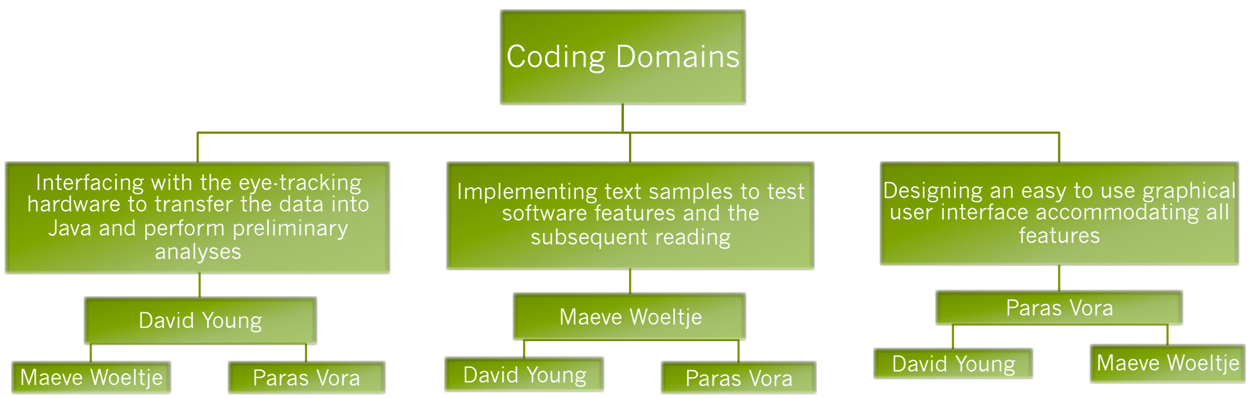


Figure 6.1.1: Division of Software Development Responsibilities

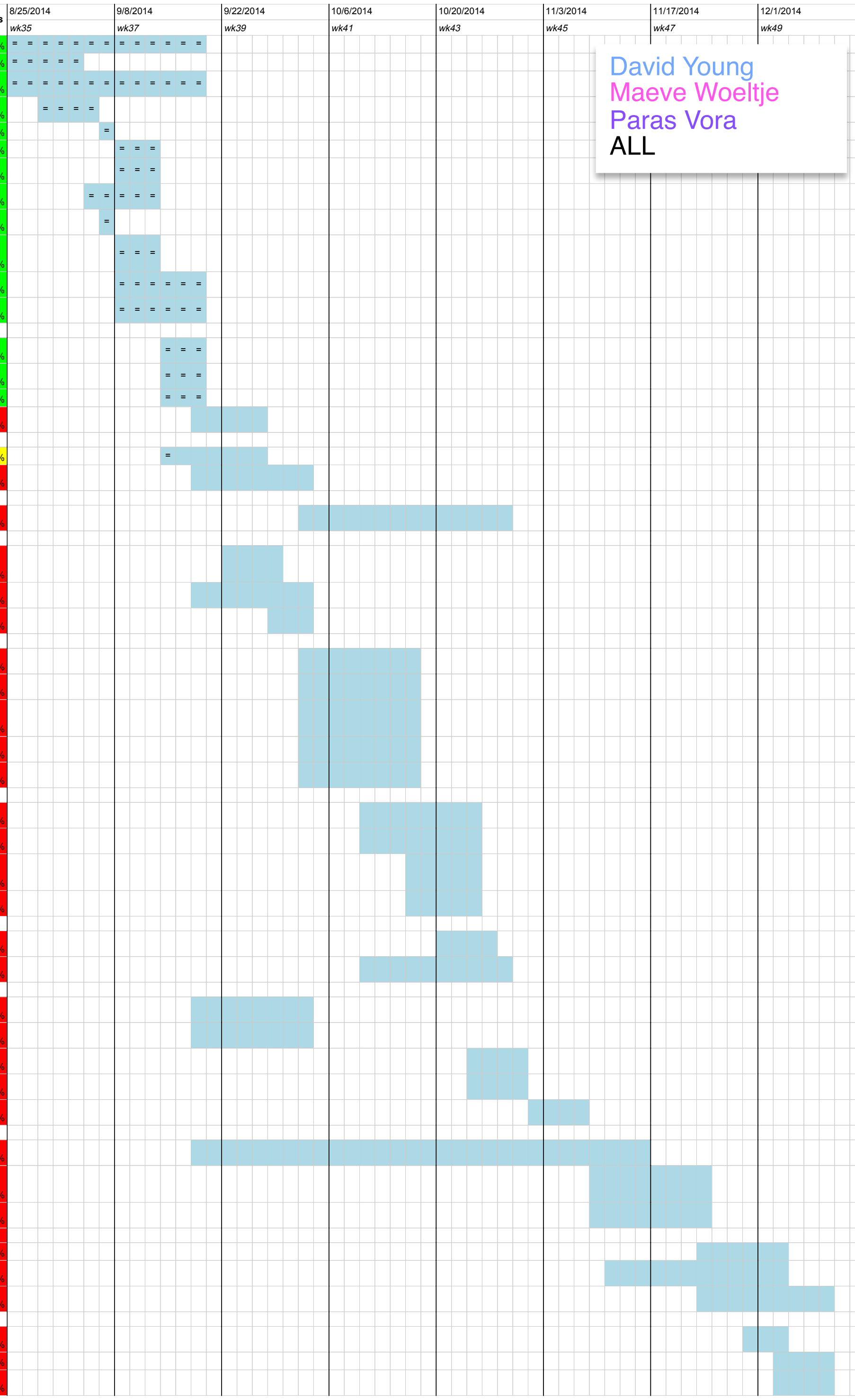
6.2 Design Schedule

The color-coded Gantt chart in Appendix A indicates the individual tasks, and their timeframes, required by each team member. These tasks not only include product development, but also include course assignments.

Appendix A

Design Schedule (Gantt Chart)

	Task description	Start date	Finish date	Progress
1	Preliminary Tasks	8/25/2014	9/19/2014	100%
2	Identify Problem	8/25/2014	9/2/2014	100%
3	Confer with Dr. Barbour about Project Direction	8/25/2014	9/19/2014	100%
4	Perform Background Research	8/30/2014	9/5/2014	100%
5	Define Project Scope	9/6/2014	9/7/2014	100%
6	Perform Patent Search	9/9/2014	9/12/2014	100%
7	Search for Existing Solutions	9/9/2014	9/12/2014	100%
8	Reach Design Specifications	9/4/2014	9/12/2014	100%
9	Organize Team Responsibilities	9/6/2014	9/7/2014	100%
10	Carry out Preliminary Analyses and Calculations	9/8/2014	9/12/2014	100%
11	Prepare Preliminary Report	9/8/2014	9/19/2014	100%
12	Prepare Preliminary Presentation	9/8/2014	9/19/2014	100%
13	Acquire Eye Tracking Hardware	9/14/2014	9/19/2014	100%
14	Connect and Calibrate Device	9/14/2014	9/19/2014	100%
15	Test Device Functionality	9/14/2014	9/19/2014	100%
16	Familiarize and get Acquainted With Java API	9/19/2014	9/26/2014	0%
17	Consult CS Faculty	9/14/2014	9/26/2014	10%
18	Outline Basic Architecture for Software on Paper	9/19/2014	10/3/2014	0%
19	Confer with Dr. Barbour About Project Direction	10/3/2014	10/28/2014	0%
20	Code Test for Device Communication and Data Acquisition	9/23/2014	9/28/2014	0%
21	Determine Method of Data Storage	9/19/2014	10/3/2014	0%
22	Test R/W Capabilities on Large Sets of Gaze Data	9/28/2014	10/3/2014	0%
23	Experiment with Different Forms of Text Mapping	10/3/2014	10/17/2014	0%
24	Experiment with Java TextLayout API	10/3/2014	10/17/2014	0%
25	Experiment with 2D Graphics & Text in Swing API	10/3/2014	10/17/2014	0%
26	Experiment with 2D Graphics & Text in SWT	10/3/2014	10/17/2014	0%
27	Experiment with Hit Detection Methods	10/3/2014	10/17/2014	0%
28	Architect and Code Basic GUI for Software	10/10/201	10/24/2014	0%
29	Create Basic Menu System	10/10/201	10/24/2014	0%
30	Create Basic Test for Text Mapping Using TextLayout or a Text Grid	10/17/201	10/24/2014	0%
31	Create Basic Test for Hit Confirmation	10/17/201	10/24/2014	0%
32	Prepare Progress Oral Report	10/20/201	10/27/2014	0%
33	Prepare Progress Written Report	10/11/201	10/29/2014	0%
34	Define All Data Output Forms of Feedback	9/19/2014	10/3/2014	0%
35	Outline Format for Output Data	9/19/2014	10/3/2014	0%
36	Design Methods for Creating Feedback	10/24/201	10/31/2014	0%
37	Code Methods for Creating Feedback	10/24/201	10/31/2014	0%
38	Test Feedback Generation	11/1/2014	11/8/2014	0%
39	Create or Select Text Samples to Test	9/19/2014	11/15/2014	0%
40	Test Device with Simulated Usage Scenarios	11/9/2014	11/23/2014	0%
41	Re-Evaluate Aspects of Execution	11/9/2014	11/23/2014	0%
42	Prepare Final Oral Report	11/23/201	12/3/2014	0%
43	Prepare Final Written Report	11/12/201	12/3/2014	0%
44	Prepare Poster Presentation	11/23/201	12/9/2014	0%
45	Create Demo Mode For Presentation	11/30/201	12/4/2014	0%
46	Ensure Code is Adaptable	12/4/2014	12/10/2014	0%
47	Consider Extra Functionality	12/4/2014	12/10/2014	0%



Appendix B

Existing Consumer-Grade Gaze-Tracking Manufacturer Specifications

Existing Consumer Grade (Priced) Eye-Tracking Hardware

Manufacturer	**SMI vision**	**Mirametrix**	**GazePoint**	**Microsoft**	**Tobii**	**Eyetribe**
Product	RED -OEM	S2	GP3 desktop	Kinect	EyeX	Tracker
Grade	clinical (comparison)	research (comparison)	consumer	consumer	consumer	consumer
Dimensions	15.4x2.5cm	35x4x3cm	-	-	32x2x1.5 cm	20x1.9x1.9cm
Weight	30g	300g			91g	70g
Operating Distance	50-70cm	-	-	137cm	40-90cm	45-75cm
Vert. Tracking Range	plus 20°/minus 40°	-	-	60°	-	-
Horz. Tracking Range	plus/minus 20°	-	-	70°	-	-
Gaze Pos. Accuracy	0.5°	0.5°	0.5° – 1°	-	-	0.5° – 1°
Spatial Resolution	0.1° (RMS)	-	-	-	-	0.1 degree (RMS)
Tracking Range	32x21cm @60cm away	-	-	-	-	-
Calibration Modes	2/5/9 pts	5 or 9 pts	5 or 9 pts	-	-	9, 12 or 16 pts
System Latency	<20ms @ 60Hz	-	-	20ms	-	<20ms @ 60Hz
Blink Recovery Time	16ms @ 60Hz	-	-	-	-	-
Works w/ glasses	yes	yes	-	-	-	yes
Max Monitor Size	28"	-	-	-	24"	24”
API/SDK	C functions	-	C based languages	C++, C#, Cx	C++	C++, C# and Java
OS Support	XP, Vista, 7 (32 & 64bit)	-	7,8	7,8	7,8,7.1	7,OSX
Sampling Rate	30-120Hz	60 Hz	60Hz	30 Hz	-	30, 40 or 60Hz
Connection	USB	-	-	USB 3.0	USB 3.0	USB 3.0
Price	$19,000.00	$4890-11500	$495.00	$199.00	$139.00	$100.00
Availability	available	available	available	available	available	available

Appendix C

Hardware Selection Design Considerations (Pugh Chart)

Pugh Chart

	weight	Tobii	Eye Tribe	Gaze Point	S2	SMI Vision	Kinect 2.0
Availability/Access	10	9	9	8	2	1	4
Cost	10	6	8	3	1	1	5
Safety	10	10	10	10	10	10	9
Programing Language	9	4	8	6	2	4	4
Consumer/User Friendly	8	9	8	3	2	1	10
Hardware Performance	7	6	6	8	5	7	8
Available support	6	7	6	4	2	4	3
Existing APIs	6	8	7	1	3	7	3
Physical size/ weight	6	5	7	5	2	9	1
Sampling rate	6	7	7	7	7	9	4
Customer support	5	10	8	7	5	5	7
Multiple Platform Usage	5	6	7	6	1	6	4
Data Input	4	5	5	5	5	5	5
Market dominance	3	6	4	1	1	2	5
Movement correction	3	5	5	8	1	7	8
Total		695	732	558	339	489	542

Availability/Access: current availability and ease of purchase
Cost: advertised or quoted price of device
Programming Language: how comfortable the development team is with compatible language
Consumer/User friendly: how easily or quickly a consumer can setup and use the hardware
Hardware performance: sampling rate, data format and storage, resolution
Available support: existing developer communities
Existing APIs: how well documented or user friendly the included API is
Physical size: ability to fit beneath a monitor, laptop or possibly tablet
Customer support: manufacturer customer support and responsiveness
Multiple Platform Usage: Parent Device or OS compatibility (such as Windows & OSX)
Data input: how easy it is to receive data from the hardware in a software solution
Market dominance: which device is more common/popular on the market
Movement correction: built in corrections for head tilting or shaking

References

1. Early Reading Proficiency in the United States. (2014, January 1). Retrieved September 15, 2014, from http://www.aecf.org/m/resourcedoc/aecf-EarlyReadingProficiency-2014.pdf

2. Goldberg, H. J., & Wichansky, A. M. (2003). "Eye tracking in usability evaluation: A practitioner's guide." In J. Hyönä, R. Radach, & H. Deubel (Eds.), *The mind's eye: Cognitive and applied aspects of eye movement research* (pp. 493-516). Amsterdam: Elsevier.

3. Poole, A. & Ball, L. J. (2006). "Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future Prospects." *In Ghaoui, Claude (Ed.). Encyclopedia of Human Computer Interaction*. Idea Group

4. Cusimano, Corey. "Eye-Tracking While Reading." *Kertz Lab*. Brown University, 08 June 2012. Web. 18 Sept. 2014. <https://wiki.brown.edu/confluence/display/kertzlab/Eye-Tracking+While+Reading>.

5. Just, M. A., & Carpenter, P. A. (1976). Eye fixations and cognitive processes. *Cognitive Psychology, 8,* 441-480.

6. Goldberg, J. H., Stimson, M. J., Lewenstein, M., Scott, N., & Wichansky, A. M. (2002). "Eye tracking in web search tasks: Design implications." *In Proceedings of the Eye Tracking Research and Applications Symposium* 2002. NY: ACM Press.

7. Sibert, J. L., Gokturk, M., & Lavine, R. A. (2000). "The Reading Assistant: Eye gaze triggered auditory prompting for reading remediation." *In Proceedings of the Thirteenth Annual ACM Symposium on User Interface Software and Technology* (pp. 101-107). NY: ACM Press.

8. Cortenrad, Hubertus M R. Immersive Reading Experience Using Eye Tracking. Koninkl Philips Electronics Nv, Philips Corp, Hubertus M R Cortenraad, assignee. Patent WO2006100645 A2. 28 Sept. 2006. Print.

9. Rosenberg, Louis B. Gaze-responsive Interface to Enhance On-screen User Reading Tasks. Outland Research, Llc, assignee. Patent US7429108 B2. 5 Nov. 2005. Print.

10. Gobert, Janice D., and Ermal Toto. Instruction System with Eyetracking-Based Adaptive Scaffolding. Worcester Polytechnic Institute, assignee. Patent US20130226845 A1. 23 Feb. 2012. Print.

11. Hallowell, Brooke, and Hans Kruse. Cognitive and Linguistic Assessment Using Eye Tracking. Patent EP2441386 A1. Oct.-Nov. 2008. Print.

12. The Eye Tribe. "The Tech Specs". *TheEyeTribe.com.* June 2014. Web.

13. Tobii. "The Eye Experience – Specifications". *Tobii.com.* September 2014. Web.

14. Gazepoint. "Products – GP3 Desktop Eye Tracker". *Gazept.com.* September 2014. Web.

15. Mirametrix. "Products – S2 Eye Tracker". *Mirametrix.com*. September 2014. Web.