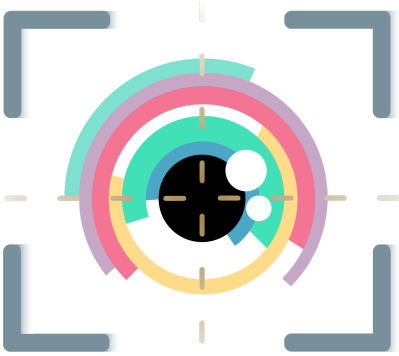
iTracking (Eye Tracking System)

Final Report

Prepared By:
Alan Wilms
Jefferson Kim
Will Lee
Pam Karwowski
Minh Chung
Matthew Kernen



VANDERBILT®
Peabody College

Presented in Fulfillment of the Requirements for

EECE 4951 Spring 2019

April 19, 2019

Project Summary

Technical Abstract

The project aimed to design an eye-tracking system that can be employed in a middle school classroom setting to collect data. In collaboration with Vanderbilt's Peabody College, the iTracking team worked to create a low-cost and robust eye tracking system by using two cameras to track the scene and subject's eyes. Current eye tracking systems provide extremely restricted freedom of movement and are expensive, creating a need for an affordable yet efficient eye-tracker.

The eye tracking system developed is capable of recognizing and localizing eyes in the camera field of view and analyzing the direction of the gaze using a cascading classifier with a thorough database. The device offers a cheap alternative to current standard devices in a research setting. The device handles various lighting conditions and user-motion moderately well.

Potential Commercial Applications

There are multiple eye tracking devices on the market, but this specific project aims to compete with affordable costs and a more customizable software to enable a more configurable user experience for researchers. The leading eye-tracking system producer, Tobii [1], offers products with significant capabilities but high cost of $1000+ - $10,000+ [2]. In addition, the eye-tracking system for this project is not intrusive, meaning that the subject need not wear any peripherals. The moderate robustness for changing lighting conditions and dynamic movement make it especially enticing for subjects of lower age.

This would serve to fill the niche for commercial as well as research aims with less stringent requirements for resolution and exact data. In addition, the cheap, portable eye tracking device would promote more research studies and further the pace for existing studies. This system is also easily scalable and integrable to any platform due to its open-source structure, which will attract researchers and corporations alike. A goal is to provide advanced data metrics along with standard gaze tracking to fit research needs.

The lower cost of this eye tracker may allow corporations who produce electronics, such as Microsoft or Intel, to more confidently integrate eye tracking into their systems. As of now, a major benefit of integrating eye tracking directly into the hardware of a system is providing face recognition to unlock one's computer. Adding this feature, however, raises the price of the computer which turns away customers. With this system, perhaps businesses will be comfortable pioneering the eye tracking field.

The government can distribute this system for research in classrooms of any age, ranging from kindergarten to college students. This system can efficiently track the eye movement of middle schoolers, so only a little work is needed to expand the system to

other classroom settings. By collecting data throughout the entire country, the potential information from data analyses will be a great benefit in creating better approaches in educating students.

Acknowledgements

The tireless efforts of our sponsor, Dr. Daniel Levin; our design professor, Dr. Ralph Bruce; and design teaching assistant, Charlie Gerrity, have been invaluable in the completion of this project. The iTracking team sincerely thanks these individuals who put in countless hours and served as our mentors throughout the process.

Table of Contents

Part 1: Project Execution

A. Statement of Work

Original System Requirements:

1. Accurate, real-time tracking of gaze
2. Reliable performance in differing settings
3. Friendly and simple user interface
4. Durable
5. Portable
6. Field of view camera
7. Moving eye-focused camera
8. Infrared illumination and sensing

Final System Scope:

1. Real time localization of eye
2. Semi-accurate calculation of gaze
3. Data collection interface
4. Laptop camera
5. Shape and light intensity based pupil detection

The outcome of the project differs greatly from the proposed device due to many factors. The work required to create an eye tracking system proved to be much more difficult and time consuming. Pieces of the project expanded as work continued which consumed more time and effort to accomplish. Additionally, the priorities of the project changed and requirements that were not significant to the core workings of the device were neglected. The final device is a substantial base for the continuation of the project in the future.

The core of the project is finding the eye and calculating where it is looking, as a result the team focused efforts on that rather than superficial parts. The moving camera was difficult to implement due to the slow movement of the stepper motors. The motors were slower than expected and relaying information to a second camera to control it would be difficult to implement. Additionally, the purpose of the second camera was to add a lens which made the idea too difficult.

B. Work Breakdown (AON and Critical Path)

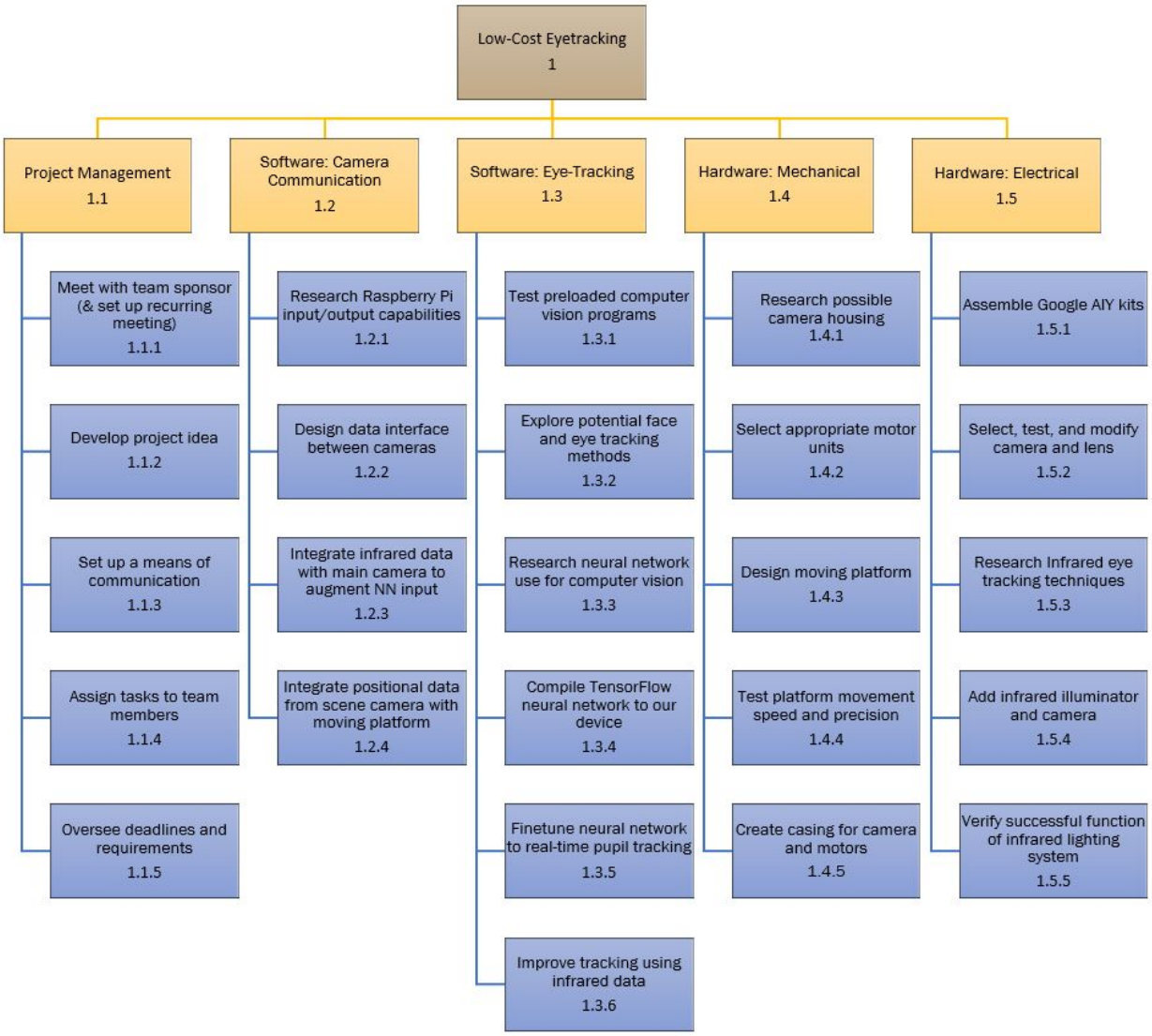


Figure 1: Work Breakdown Structure as of December 2019

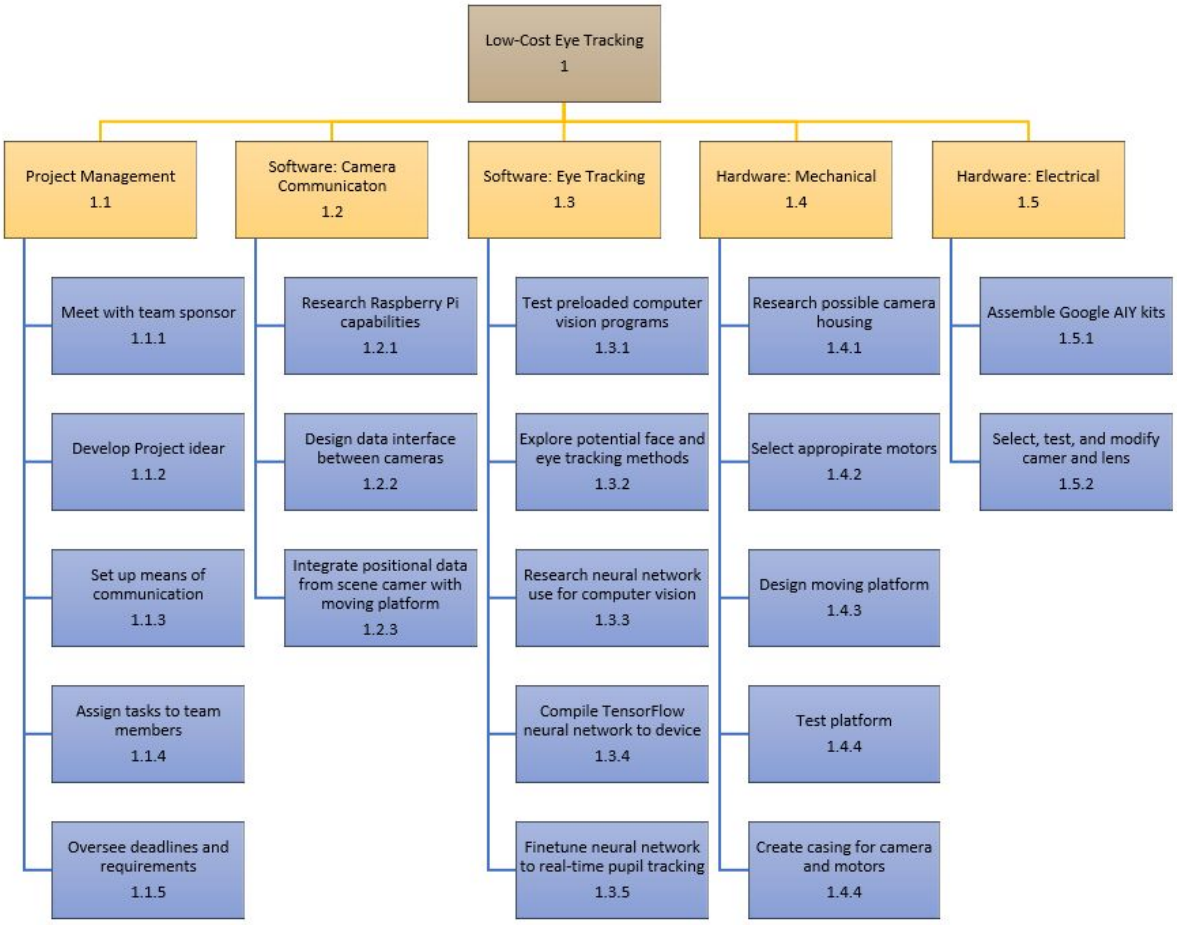


Figure 2: Final Work Breakdown Structure

The two Work Breakdown Structures show the original and updated overviews for all the work packages in our project. Most of the packages in our Work Breakdown Structure have remained the same from the CDR we submitted last semester. The five major work categories have not changed, but we have modified several subpackages related to the infrared camera and infrared data components as we have not integrated them into the final system. Specifically, the infrared work packages from the main work packages “Software: Camera Communication”, “Software: Eye-Tracking”, and “Hardware: Electrical” (main sections 1.2, 1.3, and 1.5) were removed in order to ensure that our team could focus on creating a working integrated gaze tracking system before Design Day.

AON:

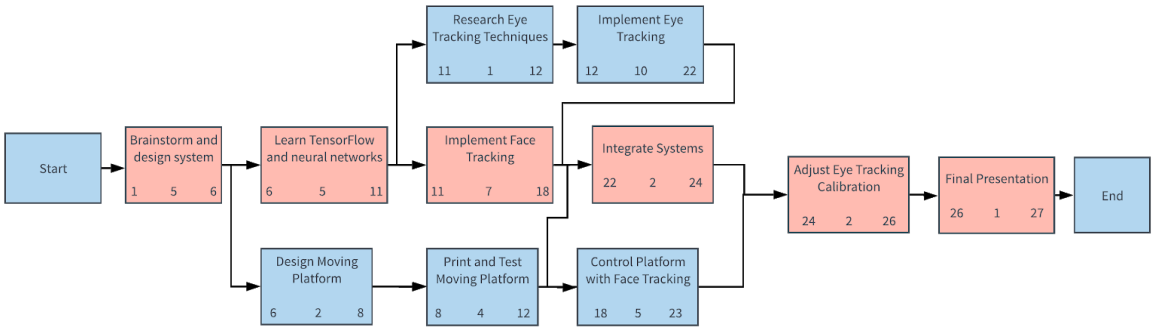


Figure 3: Activity on Node Diagram

The original Activity on Node (AON) chart can be found in the Revised Master Plan (Appendix A), and the revised AON diagram appears above. The two diagrams are color coded to demonstrate the critical path. The critical path of 27 weeks is colored by red boxes. Because the length of the semesters and school year are constant, the length of the critical path did not change from the original AON diagram. However, some of the components of the diagram as well as the lengths of tasks have been modified. The team found that the implementation of both face and eye tracking were much more complicated than originally anticipated, so the lengths given to both of those tasks increased significantly: implementing face tracking was given four extra weeks, while implementing eye tracking was given two extra weeks. The integration of the face and eye tracking, however, was slightly less time consuming than expected, so its duration was reduced by one week.

Additionally, the infrared systems intended in the original AON diagram have been removed. Given the amount of time it took to reasonably successfully code the face and eye tracking, the team decided that to construct and integrate an infrared illumination and infrared sensing system would be unfeasible. Furthermore, it was unclear how much the hypothetical infrared system would improve the tracking. Thus the team focused on improving the face/eye tracking systems given the current hardware.

Overall, the team adhered to the original AON diagram as best as possible, adjusting where necessary due to time constraints. The major issues stemmed from the anticipated lengthiest taking even longer than predicted. Still, the team followed the critical path to finish the eye tracking system.

C. Project Schedule

The Gantt chart appears in the figures underneath. The changes that were made through the course of the project are reflected in this final Gantt chart, which is the same as the Gantt chart in the previous Master Plan (Appendix A). One of significant decisions the team made was to remove the integration of infrared illuminator and systems into the system as aforementioned in the AON section.

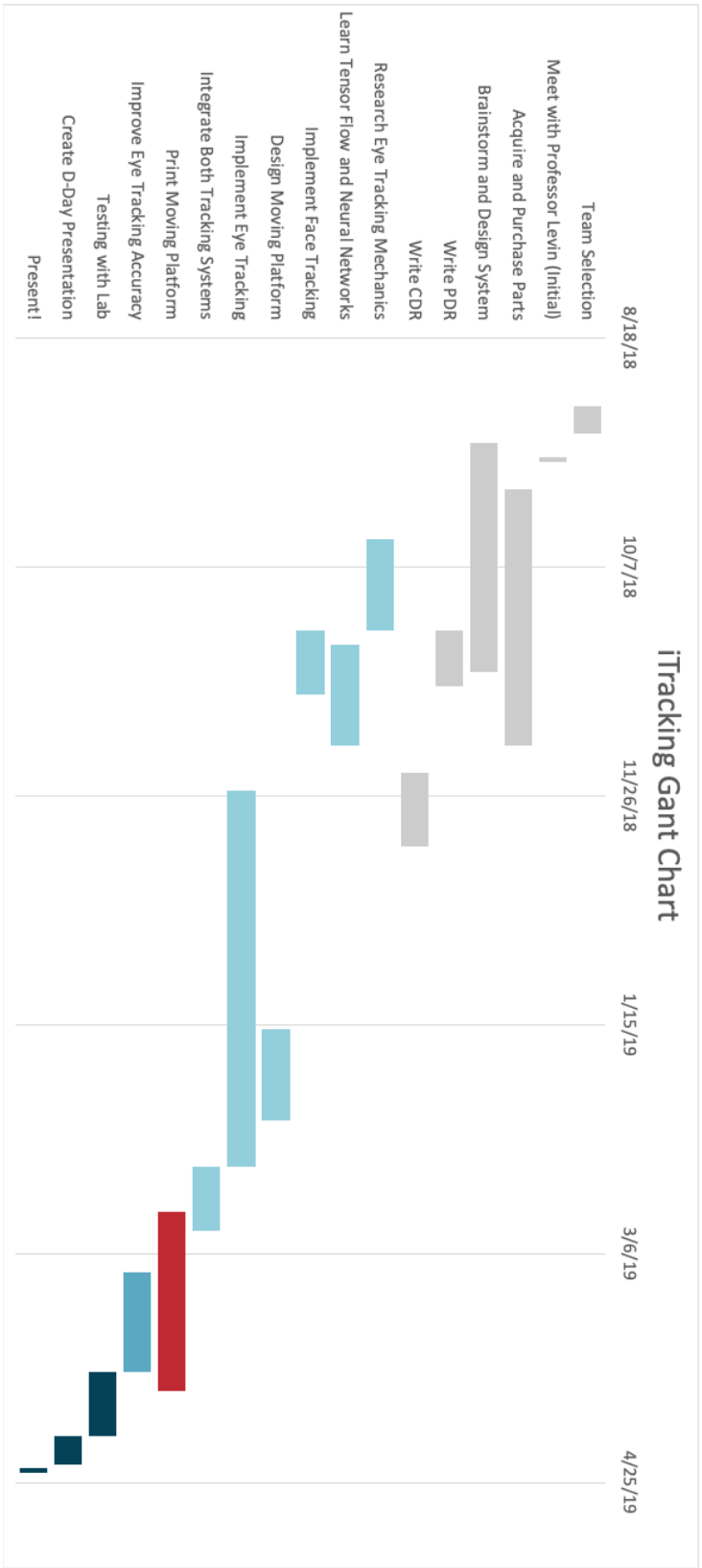


Figure 4: Gantt Chart

D. Summary Budget

Parts	Cost Per Unit	Number	Total Cost
Dell Wired Keyboard KB216 (580-ADMT)	$12.68	1	$12.68
AmazonBasics 3-Button USB Wired Mouse (Black)	$6.99	1	$6.99
LoveRPi MicroUSB to USB 4 Port Black OTG Hub for Raspberry Pi Zero	$6.99	1	$6.99
AmazonBasics High-Speed Mini-HDMI to HDMI Cable	$5.00	1	$5.00
Anbear Mini HDMI to VGA Adapter	$8.59	1	$8.59
kuman Arduino 5V Stepper Motor (x5) + ULN2003 Driver Board + Better Dupont Wire 40pin Breadboard Jumper Wires Ribbon Cables K67	$11.90	1	11.99
Google Vision Kit AIY	$89.99	2	$179.98
3D-printed System Enclosure	$0.00	1	$0.00
Pupil Labs Eye-Tracker	$0.00	1	$0.00
Materials Total			**$232.22**
Budget Summary		Total Costs (To Date)	
Project Management			$19,437.70
Software: Eye Tracking			$25,383.94
Software: Camera Connection			$2,267.02
Hardware: Mechanical			$5,282.62
Hardware: Electrical (incl. materials)			$3,630.02
Total Cost (incl. overhead and profit)			**$52,371.28**

Table 1: Budget Summary

The main cost of the project is the cost of labor paying the engineers. The actual hardware cost was only around $200 which is a fraction of the total $52,000. The main part of the project that took the most time and money was developing the software for the eye tracking. The next biggest was the project management. Beyond that, there was minimal work for hardware which was expected from our project which was mostly based on the eye tracking algorithm. For future plans, the hardware could be upgraded to more expensive and higher specification parts.

E. Sponsor Interaction

Throughout the fall and spring semester the team met Dr. Levin once a week to summarize the project progress. The team also met with Dr. Bruce during both fall and spring semester and met with teaching assistant Charlie Gerrity during spring semester.

The team kept in contact with their sponsor throughout the project via email or in person. Dr. Levin's office was located on Peabody campus so it was convenient for both parties to convene for a meeting. Dr. Levin gave freedom to choose the general direction of the project. He often provided suggestions and guidance whenever progress was stymied.

Sponsor was responsive to emails and was available whenever he was needed. Overall, the team's interactions with the sponsor were enjoyable and efficient.

Part 2: Technical

A. Summary of Proposed and Achieved

The original purpose of the project was to create a low-cost yet robust eye tracking system with matching software to be implemented for research purposes. The goal was to deploy these devices in middle school classrooms in order to collect data on student interaction with instructional videos and learning programs. Current systems are functional, but carry heavy costs as associated software is tied to the user adding to the overall cost. The design aimed to lower hardware costs by using Google AIY Vision Kits as well as remove software costs to the researchers. In order to successfully replace existing technology, it was deemed necessary that the the ability to track gaze must be fully functional. In addition, the product was designed be portable from class to class, durable enough to withstand usage with children, and ideally be able to track multiple students performing the same task. In addition, software should be customizable for various research purposes including test duration and data output. Furthermore, the program should be entertaining to maintain the attention of young users and come with a precise calibration method. A foreseeable problem was maintaining data collection while the subject is moving -- a common problem with younger children. In order to capture significant data, the proposed solution was to use a two camera system with a larger field of view camera that relays the position of the head and a second moving camera to zoom in on the eyes.

Throughout the project it was found that some requirements were unrealistic and that there were components of the project that the team had not thought of in the design. It was difficult to create software capable of successfully tracking the eyes due to constraints on our knowledge of the problem as well as constraints in the ability to train a neural net to a good accuracy. However, the team did have time to develop an ease-to-use user interface in addition to focusing on the strength of the gaze tracking. Lastly, the team was unable to achieve motor function that was effective enough to include in the final design.

The team found that coding the Google AIY Vision Kit was more difficult than anticipated. The device was not compatible with certain libraries and the additional task of translating code to work with the Raspberry Pi took considerable effort, especially given the small memory limitation. Furthermore, with all of the variables in the success of the device, it was decided that a neural network and cascading classifier provide the best outcome. However since getting the network to function proved to be a larger effort than anticipated; even with extensive databases of eye gaze data online, the output of the network was unable to meet the team’s desired standard. As a result, the cascading classifier approach was preferred. The team also spent time working with a wearable eye tracker in order to calibrate data from the known device with the created device.

The final system is capable of finding the eye in real time and maintaining focus on the eye even with movement from the user. Additionally, the device is capable of collecting data on the user's gaze as they follow dots on the screen. With the cascading classifier approach, the team was able to arrive at a 58.98% accuracy in determining the quadrant of the gaze.

B. Design Reflection

The initial project was centered around a cheap eye tracking system with supportive software capable of tracking eye movements in a middle school classroom. The initial need for the project stems from the high cost of current eye tracking devices manufactured by Tobii which require the purchase of an additional software license. In order to build a better system, keeping costs around $200 was the initial plan. Additionally, software was to be included capable of mapping the subject's gaze as well as being able to be calibrated. In regards to calibration, it was proposed to match the cursor with the gaze mapping in order to solidify measurements. Another parameter was being able to track two separate people at the same time. Oftentimes students are set to watch the same video in conjunction, so one system being able to monitor both participants would aid research. Finally, the camera system also needed to be able to work in different lighting scenarios with varying color, intensity, and type of light. At the very least, the system should be proficient in a middle school classroom environment. These initial design requirements underwent some rework at the very beginning of our project as the needs and possibilities were reviewed by the team and sponsors in conjunction.

In discussion with the sponsor, our team finalized a detailed list of specifications and hopeful capabilities of the system. One design aspect we saw of importance was making the system portable. This would allow it to function in multiple classrooms with minimal work and open research capabilities to offsite locations. In addition to being portable, the device must be durable. Then we decided the software needs to be simple to run as many possible researchers, teachers, and students would be operating the system. The student aspect of the project introduced an abundance of design parameters in our initial discussion. The software should be engaging enough to keep the attention of the young students. Also, the tracking system should be robust enough to accommodate the often rapid and erratic movements of younger children. This poses many problems in regards to having the system be accurate, not losing the subject, and working with the eyes at varying angles. In addition, there was an expressed need to accommodate various height differences between subjects as well as the addition of glasses and contacts in the equation. The design aspect proposed to fix all of these issues was a two camera system. One scene camera would capture the entire scene with one moving camera magnified to focus on the eyes of the subjects. The purpose of this setup was to prevent losing focus of moving subjects and be an easy fix for taller or shorter subjects. The system also should not be distracting or intimidating to the subject to prevent any artificial behavior. With all of these requirements in mind, the device should be able to accurately track gaze while providing a meaningful data output. Ideally

the researchers would be able to view gaze overlayed with the video programs for comparison. These design parameters made up the focus of the project direction.

The initial attempt at building the system focused mainly on being able to follow faces, including a moving camera, and tracking eye movements. To do this, two Google AIY vision kits were bought which are capable of recognizing and tracking faces. The hope was to build a convolutional neural network to track eye movements and train it to specify the location of the gaze. Then, motors were enrolled to control the secondary camera which would have a lens affixed to provide further magnification. The motors would be controlled by the output of the scene camera which would relay the location of the face to the moving camera. Stepper motors were chosen as they were well-suited for the slower moving requirement of the project. For the convolutional neural network, we enlisted TensorFlow as a base. Our design was to use 3D printing to construct a case and all moving parts for the project. Unfortunately, problems were encountered during the project which required changes in the design parameters.

Our findings were that the parts we were working with did not match up to the capabilities we required. The Raspberry Pi had too little memory which was a limiter in terms of the software available. The kit was also limited in its power capabilities, so the motor had to be externally powered while controlled by the Pi board. We were able to obtain a 32GB nano SD card to replace the original 8GB card. Also, many of the packages online were not available to the specific Pi model that we were using. To combat this, we modified programs we are able to implement on a laptop to be compatible for the Google AIY kit. Additionally, we found that the gaze tracking methods were difficult to enable with good accuracy. Our abilities were limited to the general direction of the eye, left, right, up, down; but unable to implement accuracy to reflect a spot on the screen. To fix this issue, we sampled the use of a head mounted eye tracker to extract useful data. This device, however, only works well when the wearer has their head in a constant position for the entire duration. Our approach was to take the data from this eye tracker to supplement the natural data from the eye tracking system we were developing. This way we were be able to add data that we know is accurate to our methodology.

Another method we attempted in parallel was using OpenCV in Python, which has extensive libraries on feature detection and eye detection capabilities. This method proved to be our main solution instead of the deep learning algorithm which faced problems with accuracy. We also found difficulty in figuring out how to implement a case design that would fit the motors for one camera, but also allow the cameras to be a fixed distance apart for accuracy. We kept our approach of using 3D printing for a case for one camera, and as a result our code currently tracks using a single camera.

C. Discussion of Parts Design, Testing, and, Integration

i. Stepper Motors

The original plan was to use a system with *two* AIY Vision Kits with an integrated stepper motor that would rotate the two-camera structure in responses to user movement. On the software side, this proposal worked flawlessly. Using the OpenCV cascading classifiers, the AIY kits were able to identify the subject's face and place a highlighting bounding box around the detected face edge. Programmatically, this allowed the software to generate motion commands (in the range of "turn left" and "turn right") for a potential motor connection. This program was internally tested with multiple team members by simulating the motor movement via manual rotation. The robustness of identification was also tested with multiple subjects and with various head positions and orientations. The system was able to successfully identify the face in roughly 80% of these cases. See Figure 1 in Appendix D.

From a hardware perspective, a plastic case was 3D-printed to wrap the AIY kits, providing increased durability to improve resiliency during research studies and a connection point. The electrical connections for the motors came from the AIY kit's GPIO expansion pins, but it was later discovered that the voltage provided was insufficient to maximize the rotation speed. Subsequently, the hardware design was revised to include a second power source for the motor alone. From testing, the maximum rotational speed was discovered to be five revolutions per minute, which, although low, was sufficient for all but the most animated users of the system.

Thus, the software and hardware components were verified to be functional. However, due to time constraint at the end of the project, the team was unable to fully integrate both parts. Progress towards this goal can be further expanded as future work.

ii. Computer Webcam

Most of the program development was done on a 2015 15-inch Apple Macbook Pro for convenience, but the interoperability of both Python and Google's native camera functions in the AIY Vision Kit should make the eventual switch easy. The software product was structured to be easy-to-use and extensible. Command-line arguments allow the user or researcher to specify the length in seconds of each segment of the calibration process as well as the inclusion of the accuracy tests after calibration. The program employs two threads -- one to manage the general operation of the webcam and one to handle the calibration process and accuracy test.

For every frame of the live video capture, the image is processed as black and white after modifying certain properties, such as increasing the contrast and removing small noise. A red bounding box surrounds the closest subject's face when correctly identified by the OpenCV face cascading classifier. If successful, another cascading classifier places a yellow bounding box around the subject's eyes. Subsequently, an involved

algorithm uses contours and the distance of potential matches to the very center of eye bounding box to identify the pupil. The algorithm to estimate the subject's gaze on the screen is as follows:

1. Instructions are detailed in bold letters at the bottom of the screen.
2. During the calibration, the percentages of the pupil's horizontal and vertical location with respect to the webcam viewer's left and top side taken while the user stares at four sequential dots on the screen.
3. After ¼ of the data is discarded to allow the subject time to find the points, a trim mean, removing the bottom and top 25% by value, is taken. These percentage now divine the edges of the viewing window. In other words, any percentage that falls outside of this range from bottom-to-top and left-to-right means that the subject is no longer looking at the screen.
4. The gaze is extrapolated by using the current pupil's percentage from the left and top bounds of the eye bounding box and expanding it to the size of the webcam viewing window. A red dot is assigned to this calculated coordinate.
5. A translucent green overlay is added to the quadrant where the gaze is detected. This will provide another measure of accuracy.
6. An optional accuracy test places a moving and pulsing blue dot on the screen. The program records the distance between the calculated gaze and the blue dot as well as the quadrants they are in.
7. A final processing determines the accuracy of the program by counting the percentage when the quadrants match and the percentage of the points where the distance between the blue dot and calculated gaze falls below 1/10 of the viewing window's width.

The accuracy test makes the reasonable assumption that the location of the blue dot is in fact the location on the screen that the subject is looking at. After a total of 6 full run-throughs, the average quadrant accuracy 58.98% and the average point accuracy within 128 px is 11.37%. The full code with appropriate comments can be found as Document 1 in Appendix D. Integration with the program code has not currently extended past a laptop's webcam, but again should be trivial to implement.

iii. Google AIY Vision Kits

The Google AIY Vision Kits are the final desired output due to their relatively cheap cost and ease-of-use. Early on, a program was designed using the included Raspberry Pi and Raspberry Pi Camera to detect the closest face and ignore all others. This program was successfully manually tested again in multiple lighting configurations.

Part 3: Conclusions

A. Summary

The original requirements of the project were expansive and formed from the ideal components that would make a device fit for research. The aim was to fuel cheap research with children working with computers and instructional videos. To fulfill that need, the team has worked exceptionally hard and come up with a device that functions. The final product is a more realistic version of the original plan with appended goals due to both the difficulty of the project and the constraints -- time and cost.

The device was created with two cascading classifiers using OpenCV for Python. The final product is capable of detecting faces in real time and locating the eye. The camera is also capable of following the eye as it moves and finding the center of the eye - the pupil. The system does a single initial calibration of pupil position using fixed points on the screen, which confers moderate accuracy. Our hardware components, the Google AIY kit connected to the stepper motors, were not fully integrated with our eye tracking software due to memory and version limitations of the kit combined with time limitations for constructing a new hardware system.

Despite having to cut back on some of the initial design and expectations, the team is pleased with the result of the project. The initial goals were set very high, and the team did not adequately anticipate the amount of difficulty from both hardware and software. However, with continual help from the sponsor and modifications to the design, the team was able to create a working eye tracking system. Screenshots of the working program can be seen in Appendix D, Figure 2.

B. Future Work

i. Infrared Illumination and Infrared Cameras

This project only involved the use of standard webcams and Raspberry Pi cameras and did not use any infrared illumination or infrared cameras. Many of the successful modern eye tracking systems that have been developed and are currently on the market utilize infrared illumination and infrared cameras to enhance gaze tracking capabilities. A commonly used technique in eye tracking is pupil center corneal reflection (PCCR) which requires the ability to determine the pupil location and corneal reflection. Normal light sources and cameras do not provide a sufficient level of contrast to distinguish these points of interest, so it is difficult to achieve a meaningful level of accuracy without an infrared light source and infrared detecting camera.

ii. Raspberry Pi Based System

The Google AIY Vision kits used in the project included Raspberry Pi Zeros which were capable of performing face detection and some level of eye detection, but were limited

in computing power. This, in addition to a low amount of memory included with the system, made it difficult to utilize the Raspberry Pi Zeros in a meaningful way for the eye tracking algorithm the team developed. The use of two Raspberry Pi 3s with larger system memory could be a good way to have enough increased computing power and storage to run the eye tracking algorithms.

iii. Head Mounted System for Enhanced Calibration

The use of a head mounted system was explored by the team near the end of the semester in an effort to look at a possible improved method of calibration. There are multiple examples of successful head mounted systems that use integrated systems with many cameras that track the subject's eyes as well as their field of view. Using this kind of setup could enhance the calibration of the system as the cameras are located right near the patient's eyes and can move along with the patient's head. By fixing cameras to the head mounted system, the patients can move their heads in a wider range of motion while reducing the difficulty of detecting where they are looking.

iv. Integrated Face Following Software and Hardware

The team completed working versions of the desired stepper motors controls and face detection software but was unable to fully integrate both portions. Combining the face detection software and the stepper motors to create a fully functional face following system would be the next step in completing the two camera system that was originally envisioned.

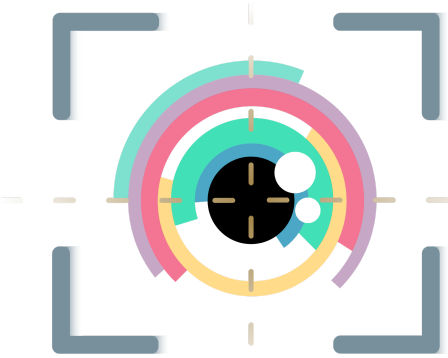
Part 4: Appendices

A. Master Plan – Revised

iTracking (Eye Tracking System)

Final Report/Master Plan

Prepared By:
Alan Wilms
Jefferson Kim
Will Lee
Pam Karwowski
Minh Chung
Matthew Kernen



VANDERBILT®
Peabody College

Presented in Partial Fulfillment of the Requirements for

EECE 4951 Spring 2019

March 22, 2019

Table of Contents

Section 1: Identification and Significance of the Innovation – Scope/SOW

The purpose of the project is to create a low-cost yet robust eye tracking system with matching software to be implemented for research purposes. The goal is to deploy these devices in middle school classrooms in order to collect data on student interaction with instructional videos and learning programs. Current systems are functional, but carry heavy costs as associated software is tied to the user adding to the overall cost. The design aims to lower hardware costs by using Google AIY Vision kits as well as remove software costs to the researchers. In order to successfully replace existing technology, the ability to track gaze must be fully functional. In addition, the product must be portable from class to class, durable enough to withstand usage with children, and ideally be able to track multiple students performing the same task. In addition, software should be customizable for various research purposes including test duration and data output. Furthermore, the program should be entertaining to maintain the attention of young users and come with a precise calibration method. A foreseeable problem is maintaining data collection while the subject is moving - a common problem with younger children. In order to capture significant data, a solution is to use a two camera system with a larger field of view camera that relays the position of the head and a second moving camera to zoom in on the eyes.

Section 2: Management and Organization Section

A. Project Management and Organization: Key Personnel and Authority Relationships

The team is composed of two computer engineers - Alan Wilms and Matthew Kernen, two electrical engineers - Will Lee and Pam Karwowski, and two biomedical engineers - Minh Chung and Jefferson Kim. The diverse areas of concentration give the project a wide perspective of the design including hardware, software, and subject focuses. The aggregate team also contains many valuable skills in software development, image processing, circuit design, and general technical aptitude and is divided by role based on experience and major. Further work breakdown will be discussed in Section 3.3.

Overseeing the team are Dr. Daniel Levin and Dr. Gautam Biswas. Dr. Levin is the co-director of the Cognition & Cognitive Neuroscience Program in the Vanderbilt Peabody School and Professor of Psychology and Human Development. His lab examines the interface between knowledge and visual perception, particularly exploring how visual attention affects learning from agent-based tutoring systems. Having collaborated with Dr. Levin on interdisciplinary research in the past, Dr. Biswas is a Cornelius Vanderbilt Professor of Engineering with a research focus in intelligent learning environments. He has developed innovative techniques with educational data mining for analyzing students' learning behaviors and their link to known metacognitive strategies. They will provide the overall direction for the project and define the requirements.

B. Manpower: Workforce requirements estimates, etc

The bulk of the project consists of the development of algorithms for tracking the eye as well as designing and constructing of hardware and electrical systems to assemble the physical device. Skills in software development (specifically in Python) and in implementing convolutional neural networks are essential to the team's efforts in developing a more powerful eye-tracking algorithm. The ability to design 3-D printable blueprints and a deep understanding of microcontrollers are also extremely important for this project. The engineers in the team do not have any extensive background in computer-aided design (CAD) or neural networks, but this is something that can be definitely self-taught in the process of building a case and moveable parts for the device and the software. Members of the team have detailed knowledge of image processing techniques and microcontrollers which will be helpful in filtering a quality signal in data processing as well as circuit design. Dr. Levin is instrumental with his current knowledge of image tracking systems as well as specific insight as to what features and data are most important for research purposes. Dr. Biswas will be helpful during the prototyping stage and for obtaining working feedback.

C. Training and Development

The team is learning how to use CAD software as well as learning how to utilize convolutional neural networks with the specific TensorFlow package. This may involve individual members of the team seeking outside assistance from Vanderbilt faculty and other students. This includes the instructional class at the Wond'ry on how to use the 3D printers. The Google AIY kits are well-document and, as expected, are greatly compatible with their open-source TensorFlow development library. As such, online resources will be the most valuable for general training and development. In the future, the team is planning to oversee the research process in order to refine design needs and take note of any specific environmental factors.

Section 3: Technical Section

A. User Requirements and System Requirements

User Requirements

- Inexpensive, portable, durable design
- All moving and electrical components kept fully away from child contact
- Video and eye gaze data easily transferable to a computer
- Simple user interface

System Requirements

- Flexibility in various settings and lighting conditions
- Integrated two-camera system to maintain focused zoom on subject's eyes
- Instantaneous gaze detection
- Neural networking to maintain software reliability without manual editing of code

- Integration of infrared light and sensing

B. Work Breakdown Structure

The following figure details the Work Breakdown Structure for this project. The project is broken into sub-projects, and necessary work packages for each sub-project are defined below.

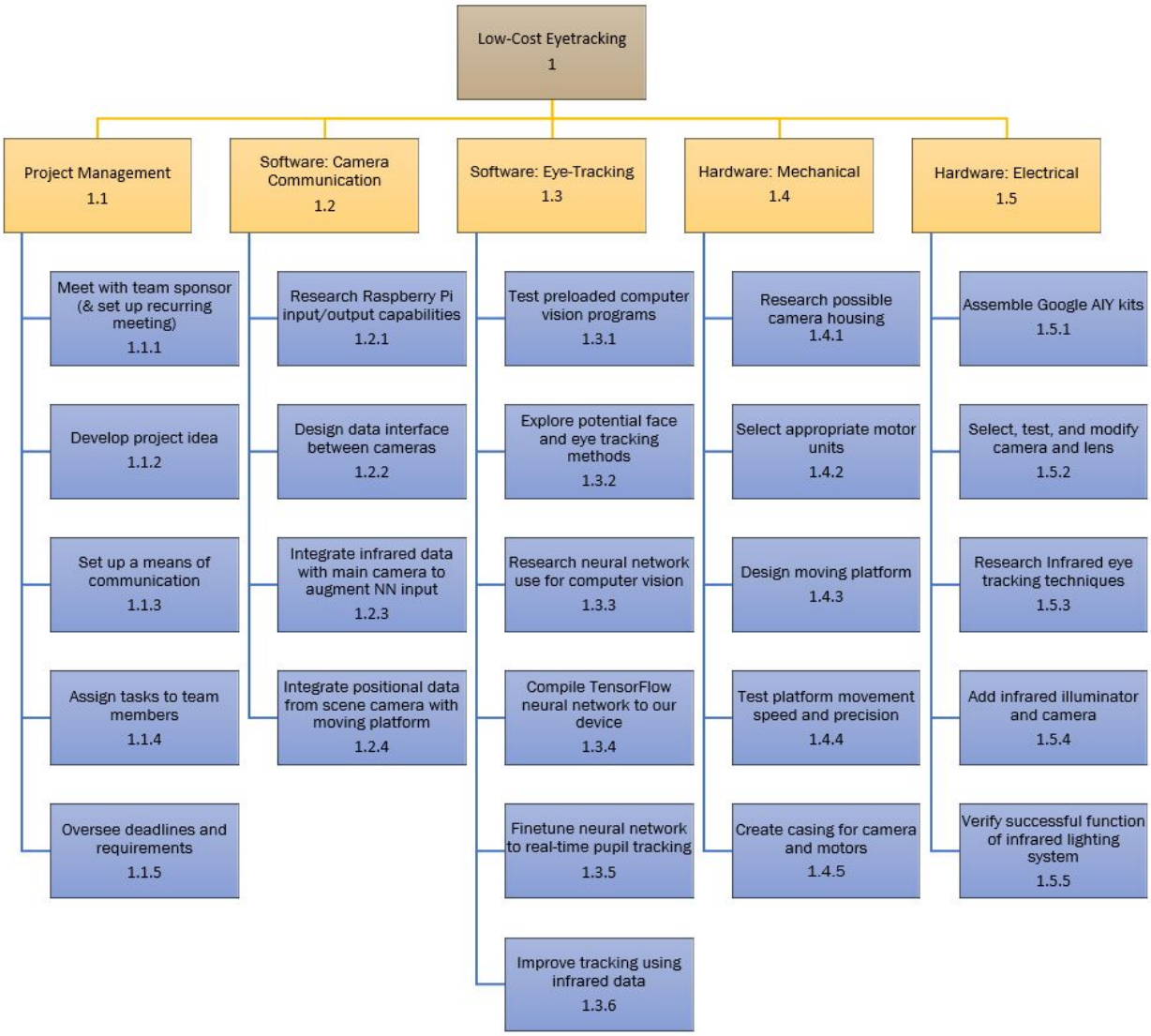


Figure 1: Work Breakdown Structure.

C. Responsibility Assignments

At this stage, we have determined the general roles and major responsibilities for this project and have properly distributed them to our team members. Current key responsibilities include motor control, neural networking, and camera modification and

communication. Each team member has experience with all aspects of design for this project and will contribute to all parts, but the main roles have been assigned based on expertise. In addition, team members all took the innovator designator “A” vs “R” test, as shown in Table 1, to further identify strengths.

Will Lee has been assigned the role of team leader due to the “A” vs “R” results. With the exception of Will, all our team members were either neutral or leaned towards “A.” Will serving as the group leader ensures that any discussions and decisions involve the opinions of an “R”-type individual.

Neural networking is a responsibility requiring a strong coding background and very little hardware experience. Therefore, this task is best suited for Alan and Matthew, as their studies focus on coding more so than hardware. Both Alan and Matthew are “A” types. Their methodical approach to problems is well-suited for work in neural networking, which will require constant code modification and compute training. Although they are both “A” types, neither Alan nor Matthew are very strongly “A” types and thus still have the capability to think radically when necessary. This is important because neural networks is still a developing field which will require new ideas.

Motor control will involve both hardware and software knowledge, making it a perfect area for all members to contribute to. This means the input of the “A” types, the “R” type, and the mid “R”/”A” type will be taken into consideration. Will and Minh have been assigned to this task. Will’s “R” type and Minh’s “A” type balance each other out, and they both have sufficient coding experience to be able to connect stepper/servo motors to the Raspberry Pi units. Additionally, both Will and Minh have independently researched how eye tracking is typically conducted and thus best understand how the cameras need to be positioned.

Camera modification and communication will primarily be the responsibility of Pam and Jeff. Pam is a moderate “A” type and Jeff is a mid “A”/”R” type, which means there will be a good balance of the “A” type and “R” type characteristics in the design process. Jeff and Pam both have experience with both signal processing and coding which will prove useful in the implementation of cross camera communication and acquisition of meaningful data with a high signal-to-noise ratio. This makes them good candidates to apply filters to the eye tracking system to eliminate background noise.

Table 1. Team “A” vs “R” Results

Name	**A**	**R**	**Difference**
Alan Wilms	7	3	4
Jefferson Kim	5	5	0
Will Lee	4	6	2
Pam Karwowski	6	4	2
Minh Chung	6	4	2
Matthew Kernen	6	4	2

D. Project Schedules

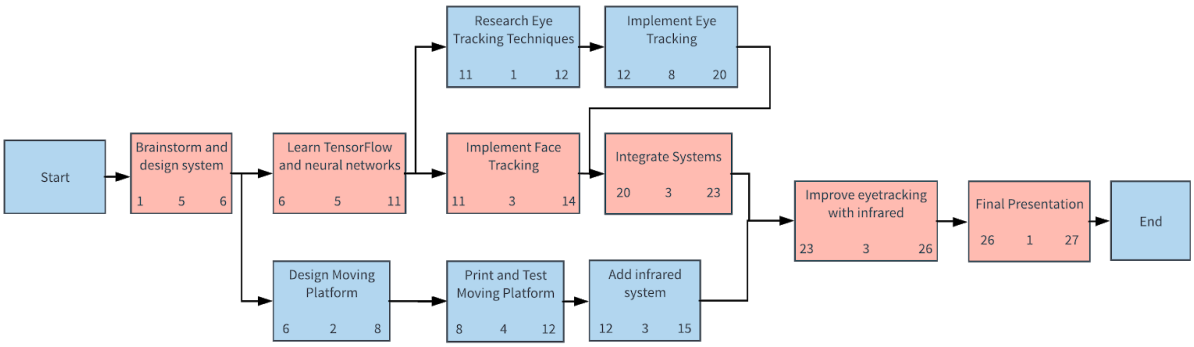


Figure 2: Activity on Node Diagram

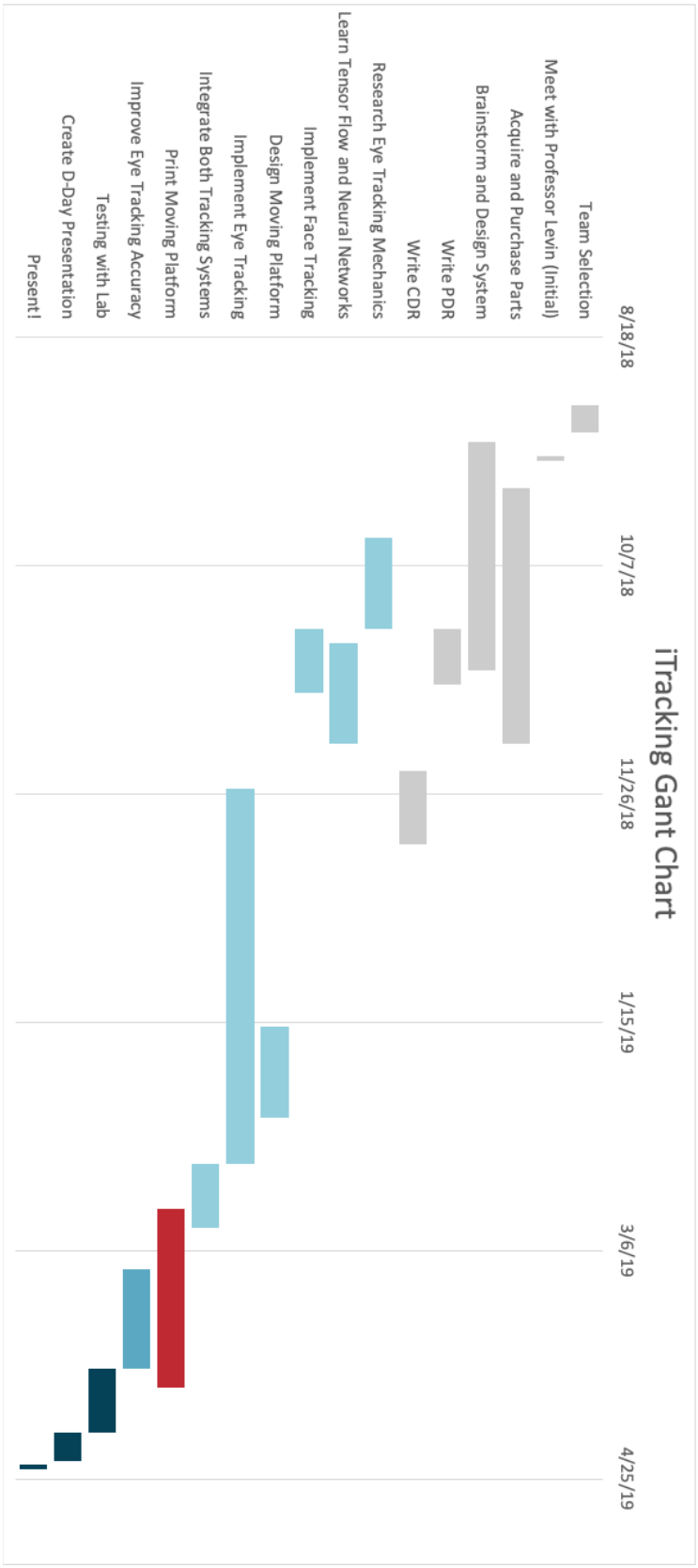


Figure 3: Gantt Chart

E. Budget

Table 2: Budget Highlights

Labor Summary	Direct Costs		
Project Management	$11,494.40		
Software: Eye Tracking	$7,778.38		
Software: Camera Connection	$8,084.50		
Hardware: Mechanical	$1,716.53		
Hardware: Electrical	$6,660.01		
Labor Total	**$35,733.82**		
Parts	Cost Per Unit	Number	Total Cost
Dell Wired Keyboard KB216 (580-ADMT)	$12.68	1	$12.68
AmazonBasics 3-Button USB Wired Mouse (Black)	$6.99	1	$6.99
LoveRPi MicroUSB to USB 4 Port Black OTG Hub for Raspberry Pi Zero	$6.99	1	$6.99
AmazonBasics High-Speed Mini-HDMI to HDMI Cable	$5.00	1	$5.00
Anbear Mini HDMI to VGA Adapter	$8.59	1	$8.59
kuman Arduino 5V Stepper Motor (x5) + ULN2003 Driver Board + Better Dupont Wire 40pin Breadboard Jumper Wires Ribbon Cables K67	$11.90	1	11.99
Google Vision Kit AIY	$89.99	2	$179.98
3D-printed System Enclosure	$0.00	1	$0.00
Materials Total			**$232.22**
Total Cost (incl. overhead and profit)	**$66,036.10**		

This table only gives budgetary highlights. A detailed budget, including the calculations along the way, is included in Appendix B. Because of the low cost need for our system, the parts costs are minimal. The Google AIY Vision kit is integrated with a Raspberry Pi Zero, Pi Camera v2, and Intel's Movidius chip that has integrated TensorFlow support as well as useful additions such as a piezoelectric speaker speaker and LED button. This integration further reduces materials costs. Software costs are free due to their open-source nature.

Due to the complexity of the software and hardware system requirements, we project a large labor total, with a predicted seven hours worked by each team member every week during the second phase of the project, to reach over $35,000. To calculate the labor costs, we assume that the team leader, Will, would be paid as a senior engineer ($73.81 per hour). As computer engineers, Alan and Matthew will be paid $38.33 per hour. Pam is an electrical engineer, so will be paid $35.94 per hour, as will the two biomedical engineers, Minh and Jeff. The hours are estimated based on the expected time to complete each subsystem and the division of responsibilities according to each team member's strength. Together with general and administrative expenses, parts, and profit overhead, the total cost will reach $66,000.

Section 4: Project Schedules

The Peabody eye tracking system project team, under the project name “iTracking,” has been working on the two approaches to implementing gaze detection with the Raspberry Pi on Google’s AIY Kits. One approach uses TensorFlow Neural Networks with ~4300 images and 3000 training steps but has only achieved a tested accuracy of 59%. Another approach utilizes the Python OpenCV library using feature detection. Unfortunately, this technique suffers from spotty recognition. The goal of the remaining weeks is to improve the accuracy of both approaches to reach a desired goal of 80% accuracy, which is defined by the difference in the calculated gaze direction and the actual gaze direction.

The detailed budget by task appears in the Appendix (with a summary in Section 2) and the updated Gantt chart appears in Section 6.

Section 5: Budget

The detailed budget appears in Appendix A. While the budget has primarily remained the same from the Critical Design Review and the previous Spring Update Report, the last six weeks of work (including spring break) have been changed from an estimation to actual numbers. These changes are conveniently highlighted in red and cover January 7th, 2019 through March 18th, 2019. The budget now consists of accurate recorded labor hours. The estimations for the rest of the semester have stayed the same.

We have since pivoted to use the Pupil Labs Eye Tracker. Thankfully, our sponsor already had a unit on hand, so the materials cost is $0. Normally, this hardware would cost upwards of $1000, and would involve some manual labor to install the infrared-LED, which would undoubtedly increase the hours reported under “Hardware: Mechanical.” For now, we are still using the slow stepper motors and have held off from procuring an additional set of servo motors. As a result, the $20 cost from the last Spring Update Report has been removed.

Our budget overview has been updated to reflect the slight change in the labor summary. Again, from the CDR’s rough estimate, it is clear that more labor hours have been allocated toward the camera connection, in which we include the creation of software to control the motors. We are looking at a total estimated cost, including all overhead and profit, of approximately $50,000. While this is a large sum, when taken in context, it equates to approximately $8,333 per team member for work done over 8 months.

Parts	Cost Per Unit	Number	Total Cost
Dell Wired Keyboard KB216 (580-ADMT)	$12.68	1	$12.68
AmazonBasics 3-Button USB Wired Mouse (Black)	$6.99	1	$6.99
LoveRPi MicroUSB to USB 4 Port Black OTG Hub for Raspberry Pi Zero	$6.99	1	$6.99
AmazonBasics High-Speed Mini-HDMI to HDMI Cable	$5.00	1	$5.00
Anbear Mini HDMI to VGA Adapter	$8.59	1	$8.59
kuman Arduino 5V Stepper Motor (x5) + ULN2003 Driver Board + Better Dupont Wire 40pin Breadboard Jumper Wires Ribbon Cables K67	$11.90	1	11.99
Google Vision Kit AIY	$89.99	2	$179.98
3D-printed System Enclosure	$0.00	1	$0.00
Pupil Labs Eye-Tracker	$0.00	1	$0.00
Materials Total			**$232.22**
Budget Summary	Total Costs (To Date)	Total Costs (Estimated)	
Project Management	$13,539.32	$18,040.05	
Software: Eye Tracking	$16,919.23	$20,319.26	
Software: Camera Connection	$2,267.02	$3,316.27	
Hardware: Mechanical	$5,282.62	$8,513.44	
Hardware: Electrical (incl. materials)	$3,630.02	$8,056.35	
Total Cost (incl. overhead and profit)	**$38,008.19**	**$50,189.02**	

Table 3: Actual and Estimated Budget Overview To Date

Section 6: Documentation

External Documentation

The project implements Google AIY Vision Kits which have online instructions of how to build, set up and use pre-installed programs on the kits. The instructions also contain information about pin configurations on the Raspberry Pi that is in the kit; the 40-pins are especially documented because some of those pins are already used in the connection with the vision bonnet. The instructions also contain information on the use of a Python API library. These APIs are also commented in several sample scripts, including one for face detection. Further instructions describe how to tweak the neural network model TensorFlow compiler, which is relevant to the project.

Our design also now incorporates the Pupil Head Mounted Eye Tracker, which comes with detailed documents for the eye tracker itself as well as all the associated programs and functions. These include the construction of the eye tracker, the installation and execution of the tracking programs, and the extraction of the data. These documents are highly valuable to our project since we can modify the tracker and calibrate the programs to best suit our needs. Furthermore, the Pupil program code is open source, so we can examine it to help us improve our own software.

Additionally, the datasheet of the stepper motors is one of project's vital pieces of external documentation. The datasheet provides the team with a lot of important specifications regarding the motor and its functionality. These include information on how to operate the motor, its functioning range, torque per square millimeter and the voltage/current needed to power the motor.

Internal Documentation

The algorithms and software are quite robust for this project, which in turn has an entire repository of new Python code. All code is managed through GitHub for not only an ease of access but also as an efficient way to handle code history and merge conflicts. Every member of the team is added to the private GitHub repository as a collaborator so that code changes can be worked on simultaneously and to prevent bottlenecking. Furthermore, the “README.md” file that is automatically loaded is detailed with the terminal commands necessary to run various software components, such as the face detection script, the stepper motor script, and the work-in-progress eye tracking script.

Additionally, all code is commented for improved readability. This serves to improve the ability of each team member to understand and interface with code written by another team member for use in their own code. At the end of our project, in addition to making our private repository open to our sponsor in the Peabody School, we are considering making the entire repository public so that other universities can benefit from our work for their own research.

In addition, the team maintains internal documentation on a shared Google Drive folder. The folder contains such resources as research done on eye tracking algorithms and explanation of the neural network logic of the project. The folder also contains test results, CAD designs, parts that need to be purchased, and weekly reports that detail each team member’s progress.

Lastly, the team employs Asana, an application designed to help organize, to distribute tasks and keep all members updated. All the tasks that are completed and that need to be completed are laid out in a fashion which is easy to comprehend.

Section 7: Standards

The main standards for the project relate to the hardware and software of the eye tracking system. The hardware, including the motors, housings, Raspberry Pi, and wires, must adhere to the IEEE design and safety standards. The software will also follow standard software development practices and standards, with the proper documentation, source control, and language standards specific to Python.

Also, our project must follow the standards set forth by our project sponsor, the Peabody School of Education. The eye tracking system is being designed for the use in a classroom setting and must be in well enough to withstand normal movement and adjustment. The housing of the system should enclose all of the exposed wires and circuit boards in order to prevent any injuries to the students.

Section 8: Risk Analysis

Sources of impact on our project have remained the same since our last evaluation. Cost holds the lowest weight because the project is currently well within budget. Scheduling has a somewhat large impact, as we have a deadline that cannot be extended past April. Still, impacts to our schedule are highly unlikely to cause a delay longer than a few weeks. Technical factors are given the highest weight and impact; anything that could cause a downgrade in performance will result in our device being inoperative; the eye detection, camera movement, and gaze calculation all need to function properly to create meaningful data. Together, our composite impact factor (CIF) is 0.42, which again reflects a moderate risk.

Impact Source	Weight	Impact Result	Impact Value
Technical	0.4	Significant Reduction in Performance	0.7
Cost	0.2	Within Budget	0.1
Schedule	0.4	Minor Slip (<1 month)	0.3

Table 4: Impact Values in the Case of Risk

$$CIF = (0.4)(0.7) + (0.2)(0.1) + (0.4)(0.3) = 0.42$$

Our updated risk analysis still includes both hardware and software, considering the importance of both to our project. The risk from immaturity of hardware has been decreased because our design runs off fairly dated and established components, such as the Raspberry Pi Zero. However, the hardware complexity risk remains high due to the many parts included in our system: the AIY kits (Vision bonnets, Raspberry Pi Zeros, and Raspberry Pi Cameras), the Pupil eye tracker, the motors, and various cables. The hardware only requires USB connections for communication and control. Having all these parts working in conjunction proves challenging, but the camera positioning does not require high precision since the eyes and face are large, prominent features.

The immaturity of our software is not a huge concern because we are mostly using well-established packages and libraries, such as Python's OpenCV library. However, the creation of our own neural network and our custom face/eye detection programs introduces a great degree of software complexity into our project. Not only are they extremely complicated, but they are also convoluted and difficult to develop flawlessly. Since these algorithms are the basis of the eye tracking in our project, they are required to work extremely well.

External factors remain with the lowest weight since the finished product is ideally self sustaining. The use of neural networks in software design reduces the need to provide

software updates, and our physical components are nothing state-of-the-art. Therefore, we expect that our device will perform consistently without support. Considering all our sources of failure together, our composite likelihood factor (CLF) is 0.50, which represents a moderate overall risk.

Sources of Failure	Relative Weight	Status	Risk Likelihood
Immaturity of Hardware (Raspberry Pi Board/Camera, Pupil Labs Tracker)	0.1	Minor Redesign	0.3
Immaturity of Software (OpenCV library, custom Neural Network)	0.2	Existing	0.2
Complexity of Hardware (System of two moving cameras)	0.2	Moderate Complexity	0.5
Complexity of Software (Neural Networks, OpenCV program)	0.4	High Complexity	0.8
Dependency on External Factors (Need for updating)	0.1	Independent	0.1

Table 5: Sources and Likelihood of Failure

$$CLF = (0.1)(0.3) + (0.2)(0.2) + (0.2)(0.5) + (0.4)(0.8) + (0.1)(0.1) = 0.50$$

$$RCR = CLF + CIF - (CLF)(CIF) = 0.50 + 0.42 - (0.50)(0.42) = 0.71$$

Risks	Value
CLF	0.50
CIF	0.42
RCR	0.71

Table 6: Risk Values

Section 9: Updated Gantt Chart

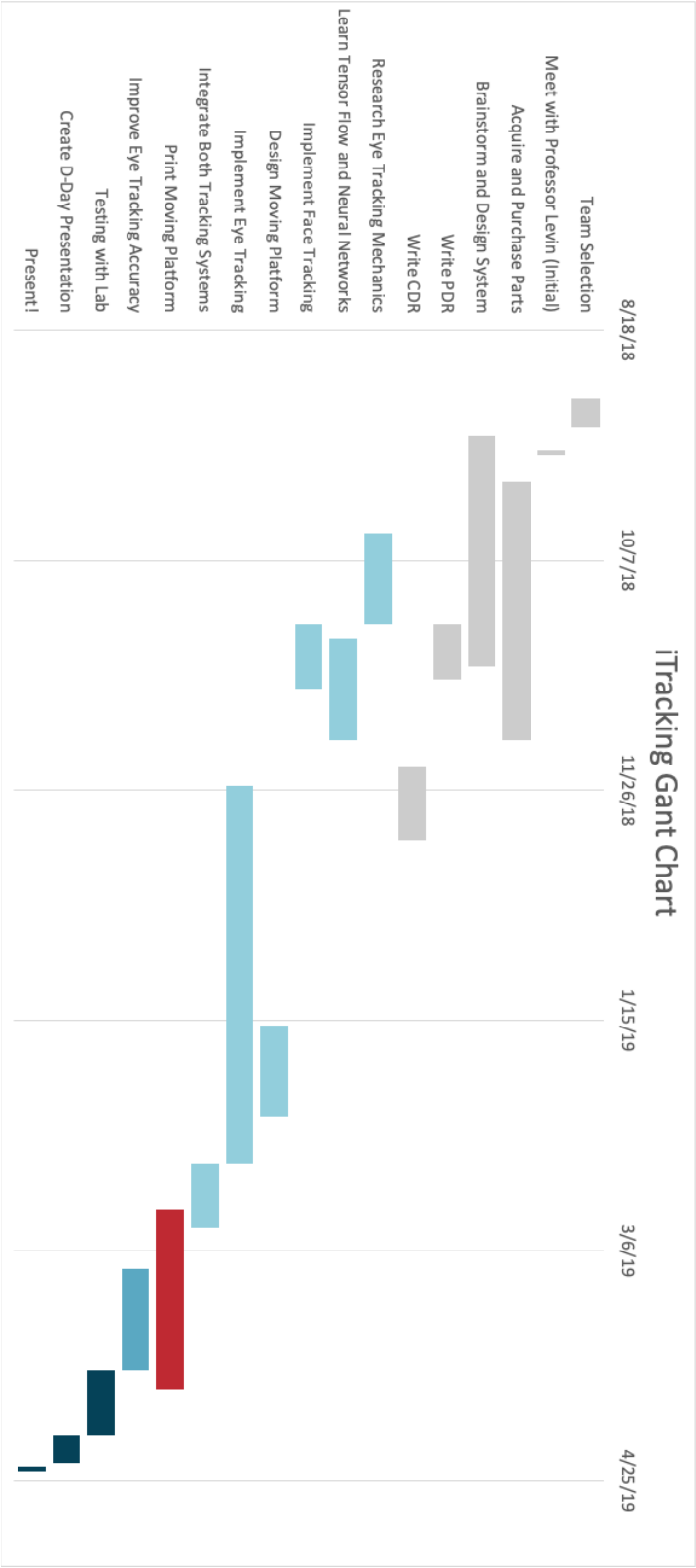


Figure 4: Project Gantt Chart with Updates

The main change is scrapping the inclusion of infrared-LED and replacing it with additional time for tweaking the accuracy of the eye tracking algorithm. This change is highlighted in red.

Section 10: Design Process Reflection

The initial project was centered around a cheap eye tracking system with supportive software capable of tracking eye movements in a middle school classroom. The initial need for the project stems from the high cost of current eye tracking devices manufactured by Tobii which require the purchase of an additional software license. In order to build a better system, keeping costs around $200 was the initial plan. Additionally, software was to be included capable of mapping the subject's gaze as well as being able to be calibrated. In regards to calibration, it was proposed to match the cursor with the gaze mapping in order to solidify measurements. Another parameter was being able to track 2 separate people at the same time. Oftentimes students are set to watch the same video in conjunction, so one system being able to monitor both participants would aid research. Finally, the camera system also needed to be able to work in different lighting scenarios with varying color, intensity, and type of light. At the very least, the system should be proficient in a middle school classroom environment. These initial design requirements underwent some rework at the very beginning of our project as the needs and possibilities were reviewed by the team and sponsors in conjunction.

In discussion with the sponsor, our team finalized a detailed list of specifications and hopeful capabilities of the system. One design aspect we saw of importance was making the system portable. This would allow it to function in multiple classrooms with minimal work and open research capabilities to offsite locations. In addition to being portable, the device must be durable. Then we decided the software needs to be simple to run as many possible researchers, teachers, and students would be operating the system. The student aspect of the project introduced an abundance of design parameters in our initial discussion. The software should be engaging enough to keep the attention of the young students. Also, the tracking system should be robust enough to accommodate the often rapid and erratic movements of younger children. This poses many problems in regards to having the system be accurate, not losing the subject, and working with the eyes at varying angles. In addition, there was an expressed need to accommodate various height differences between subjects as well as the addition of glasses and contacts in the equation. The design aspect proposed to fix all of these issues was a two camera system. One scene camera would capture the entire scene with one moving camera magnified to focus on the eyes of the subjects. The purpose of this setup was to prevent losing focus of moving subjects and be an easy fix for taller or shorter subjects. The system also should not be distracting or intimidating to the subject to prevent any artificial behavior. With all of these requirements in mind, the device should be able to accurately track gaze while providing a meaningful data output. Ideally the researchers would be able to view gaze overlayed with the video programs for comparison. These design parameters made up the focus of the project direction.

The initial attempt at building the system focused mainly on being able to follow faces, including a moving camera, and tracking eye movements. To do this, two Google AIY vision kits were bought which are capable of recognizing and tracking faces. The hope was to build a convolutional neural network to track eye movements and train it to specify the location of the gaze. Then, motors were enrolled to control the secondary camera which would have a lens affixed to provide further magnification. The motors would be controlled by the output of the scene camera which would relay the location of the face to the moving camera. Stepper motors were chosen as they were well-suited for the slower moving requirement of the project. For the convolutional neural network, we enlisted TensorFlow as a base. Our design was to use 3D printing to construct a case and all moving parts for the project. Unfortunately, problems were encountered during the project which required changes in the design parameters.

Our findings were that the parts we were working with did not match up to the capabilities we required. The stepper motors were much slower than expected prompting a switch to servo motors which are more expensive and require more set up but are ultimately faster. Also, the raspberry pi had too little memory which was a limiter in terms of the software available. We were able to obtain a 32GB nano SD card to replace the original 8GB card. Also, many of the packages online were not available to the specific model that we were using. To combat this, we modified programs we are able to implement on a laptop to be compatible for the Google AIY kit. Additionally, we found that the gaze tracking methods were difficult to enable with good accuracy. Our abilities were limited to the general direction of the eye, left, right, up, down; but unable to implement accuracy to reflect a spot on the screen. To fix this issue, we sampled the use of a head mounted eye tracker to extract useful data. This device, however, only works well when the wearer has their head in a constant position for the entire duration. Our approach is to take the data from this eye tracker to supplement the natural data from the eye tracking system we are developing. This way we will be able to add data that we know is accurate to our methodology. The headset software also has the ability to post-process the tracking data based on points that the user is intentionally staring at. Using this, we should be able to calibrate our own device and input the data into our neural network to develop a working algorithm.

Another method we attempted in parallel was using OpenCV in Python which has extensive libraries on feature detection and eye detection capabilities. This method could be an alternate solution to the deep learning algorithm which faced problems with accuracy. We also found difficulty in figuring out how to implement a case design that would fit the motors for one camera, but also allow the cameras to be a fixed distance apart for accuracy. We kept our approach of using 3D printing, but implemented various designs for the overall system until we arrived at a two-platform approach. This enables us to have a moving camera, but also ensure that whenever the system runs the cameras are a set distance apart.

Section 11: Change Control Plan

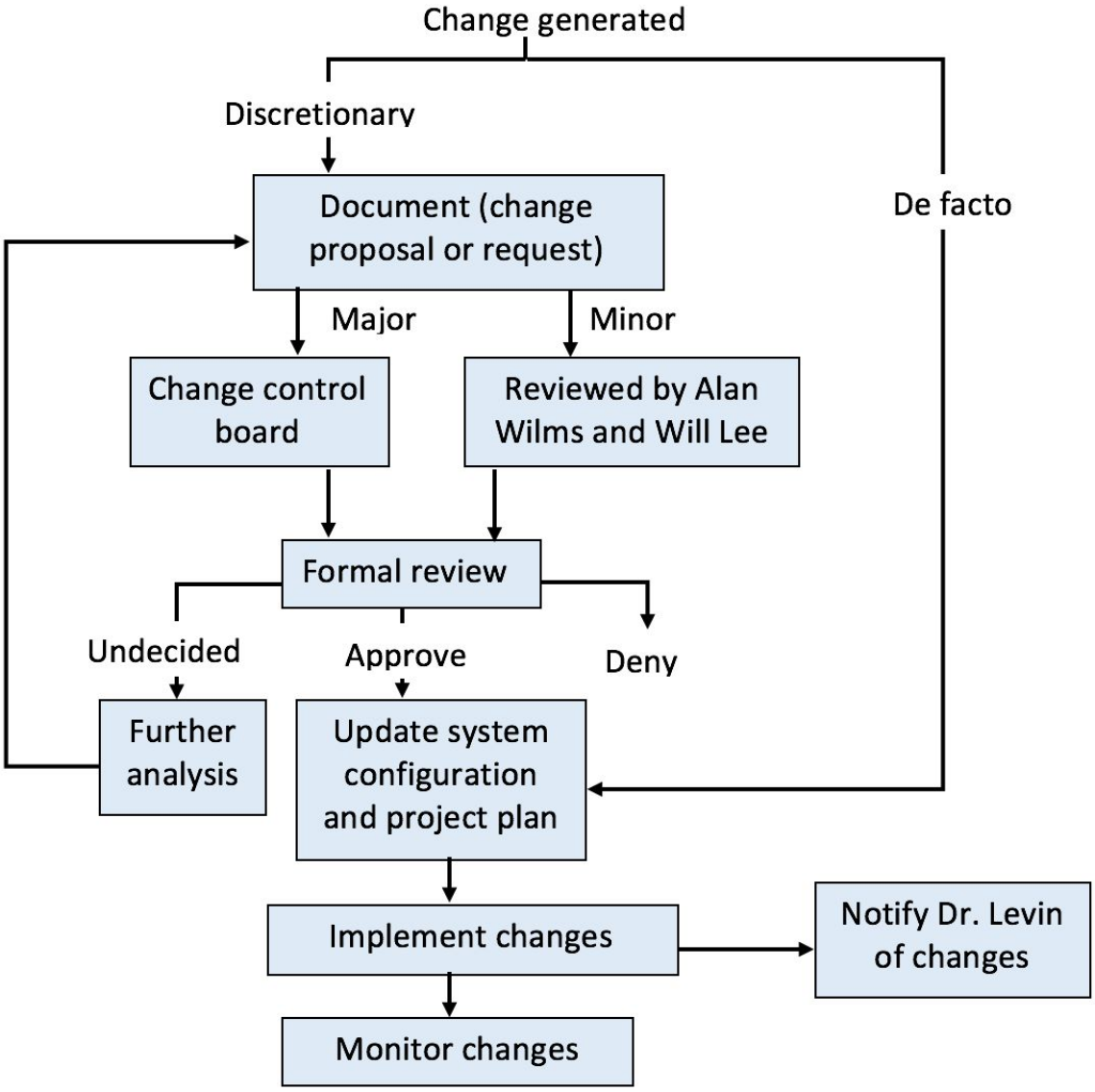


Figure 5: Change Control Plan Diagram

Due to the complexity of the goal that the group was trying achieve, there have been many instances that needed changes to occur on the preexisting plans. Some changes are brought up during meetings or during the process of planning the following steps, and were reviewed and implemented. On the other hand, some changes were implemented without review or discussion when there were minor changes.

Changes that result from modifications to the project requirement/goal are reviewed by the group members and evaluated. When the group members agree that the change is pertinent and efficient, we update the system configuration and project plan and implement changes to our project. This type of change occurred in high frequencies to

our stretch goals - first our stretch goal was to implement an infrared sensor system to our eye tracker for robustness, but this goal was altered multiple times which led the goal to change to implementing a two camera system. The changes were mostly by limitations of time and budget for our project.

The changes that do not go through the review process, or de facto changes, are implemented as soon as the team members solidify the changes. These changes also occurred very frequently, as the project is extremely software intensive. Since the team members are using the limited hardware on the Google AIY kits which have limited processing power, the hardware limitations are mostly what causes these changes. Our change in eye tracking algorithm with Tensorflow delineates one of the examples of such changes. The Raspberry Pi mounted on the Google AIY kits did not have enough processing power to handle all the libraries for the eye tracking algorithm. Changes to the algorithm was implemented to make it lighter to need less processing power.

All these changes mentioned above are reported to our sponsor at the weekly meeting and are given feedback. If there are any problems or complications with the design requirements, changes are proposed and discussed with the team members.

Our sponsor, Dr. Levin, has proposed numerous changes along the progress of the project. At the weekly meetings, our sponsor provided insight and feedback to our progress and proposes some changes if needed. The changes are documented and discussed with the group, and if the change is feasible, the team implements those changes. Additional needs for changes arose during this process quite frequently, and those were taken care of as outlined before.

All changes are documented and kept on record (Github, Google Drive).

Section 12: Testing Plan

The Peabody college does not have an extensive testing process in place for the type of device we are fabricating. However, in order to ensure that our device works to the required standards we will pursue a large degree of our own testing. Our unit testing focuses mainly on the accuracy of the eye-tracking system for various environments and situations. However, we also have requirements for the durability of the device and capabilities in terms of run-time and processing ability. In order to test the accuracy of the device in various situations, we stressed the limits in terms of lighting, user motion, head position, inclusion of glasses, different facial structures, and height differences.

To test for lighting effects, we used the device in a variety of locations. A classroom was the first and most important location to test the device as the classroom setting is our intended market. We made sure to test multiple classrooms with different lights such as fluorescent lights with gratings and without, upward facing lamps, and individual bulb lights. In each scenario we also experimented with open windows and closed windows and supplemented the open windows with a phone flashlight to ensure proper function

in an especially bright location. We also attempted lights off in the rare case that the device would be used without lights.

User motion and head position are two components of the project that make the objective unique as children are commonly seen to move frequently and erratically. While using the device we had our team members move back and forth, left and right, up and down, and tilting their heads. This is a critical part of our testing to make sure that the device does not stop transmitting information. These proved to not be a problem for basic identification of the eye which is the first part of the tracking algorithm. However, further testing needs to be done to ensure that gaze is not disrupted with motion.

Person to person differences such as height, glasses, facial structure were all tested using each member of our six person team. We used various chair positions to simulate tall people and short people and tested in those conditions. We also had them wear glasses and conducted tests, however, all members wore the same glasses which could require further testing. Due to the limited members of our team, we are looking into methods to test a larger population and have more data to feed into the neural network. To do this we have discussed setting up a station with our device and detailed instructions for anyone to participate. This would enable us to collect large amounts of data samples while also having a diverse population with organic movement. Due to the timeline constraints, as evident in Section 3, this will likely be an future improvement for the project that would happen after Demo Day.

Ideally, the project would undergo an intensive test-run in the actual research setting and with the same children who usually participate in the research study. Since the research study analyses child engagement in video tutorial, we cannot simply replicate the experiment twice since the subsequent time, the child will be certainly less engaged and attentive. A-B testing, where one half of the participants uses the existing Tobii Eye Tracker and the other half employs our eye tracker, can compare multiple components, such as the usability of the output data, the robustness of tracking under more difficult conditions, the set-up and break-down time, and unobtrusiveness to the experimenters and experimentees. Possibly during this test-run, the experimenters can replace the existing video tutorials with a custom testing program (to be developed) that instructs the participants to simultaneously follow a visible moving dot with their eyes while simultaneously moving the computer cursor to match their gaze direction. This would provide a source of truth by which the accuracy of both hardware units.

Section 13: Work Review Plan

The Work Breakdown Structure (Figure 3) shows the updated overview for all the work packages in our project. Most of the packages in our Work Breakdown Structure have remained the same from the CDR we submitted last semester. The five major work categories have not changed, but we have modified several subpackages related to the

infrared camera and infrared data components as we have not made as much progress in this area as we had anticipated and are not confident that it can be integrated into the final gaze tracking system. Specifically, the infrared work packages from the main work packages “Software: Camera Communication”, “Software: Eye-Tracking”, and “Hardware: Electrical” (main sections 1.2, 1.3, and 1.5) have been removed in order to ensure that our team can focus on creating a working integrated gaze tracking system before Design Day.

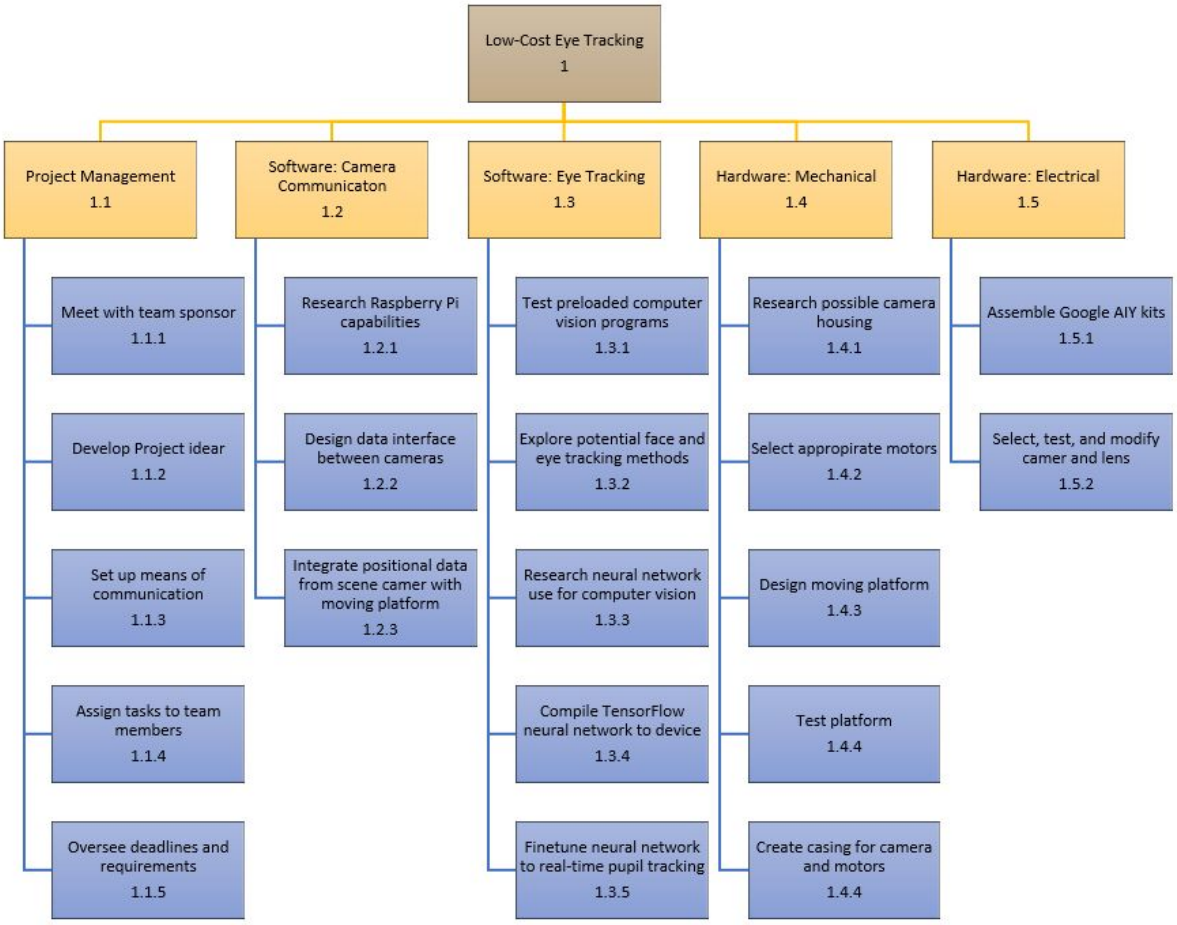


Figure 6: The Updated Work Breakdown Structure after Work Review

The team’s Work Review Plan ensures that each team member is up to date with and knows the progress on different parts of the project. We use a shared Google Drive folder to share our information, plans, and research and frequently update and create Google documents with any new or relevant information that we find. The Google Drive folder also contains documentation, research, and course materials relevant to our project. When a team member has any new findings or research papers they have discovered that could be pertinent to our project, they will add links to or will include the entire paper into our shared Google Drive folder to make sure that everyone on the

team can see and benefit from the new information. We also use the Google Drive folder as an efficient way to collaborate on and complete course requirements, including our weekly update reports, major progress reports, and presentations.

The code for our project is stored in a GitHub repository that each team member has access to so all team members can view, comment on, and contribute to the changes and updates that are pushed throughout the development process. With this source control we can see what changes were made and when they were made and can efficiently identify and fix errors that were introduced in different versions. Using Git for our version control makes our progress trackable and helps us in identifying the rate at which progress has been made.

In addition to our work review through Google Docs and source control through GitHub, our team meets twice a week during the class time when there is no lecture scheduled. During those meetings our teams continues to work on the project together in addition to updating each other on the work we have done throughout the week to keep everyone informed on the progress of the hardware and software components of the project. We also meet with our advisor, Dr. Levin, and our TA, Charlie, once a week to ensure we are staying on track and keeping all the relevant stakeholders informed on the progress and direction of our project.

Section 14: Procurement Policies Plan

Our design project is sponsored by the Peabody College. Our advisor and his team at Peabody have spent some time researching this topic and thus have money and other resources set aside for further research. As a result, a large majority of the financial aspect of our project is handled directly by our advisor. Whenever we require a new part, we first discuss it with our advisor. This is usually done in our weekly meeting, and further discussion is done via email. We send over an email with the name of the part, a description, and a direct link. He then responds and tells us if there are any issues when it comes to acquiring the specific part we selected. Next, he orders it and notifies us when it arrives, and we subsequently schedule a time to go pick it up from his Peabody office.

In the case that our advisor cannot assist us in the procurement of certain parts, we follow the procedure set by the EECE department. This means we create an excel spreadsheet detailing the parts we need with information such as the part number and supplier. We then send the document over to Ralph Bruce. He reviews it and sends it over to Jamie Harris, who is in charge of ordering all parts for the department.

Section 15: Quality Plan

There are certain steps we should take to ensure the parts we receive are correct and match what we order. First, we inspect the incoming parts. This means we look at the label and compare the name and serial numbers on the box with those that we sent over to our advisor to be ordered. Our project requires larger hardware components, such as Google AIY kits, Raspberry Pis, and the Pupil tracker, as opposed to smaller components like resistors and wires. As a result, testing the received parts does not require constructing circuits. Instead, we try out the demo programs for the larger devices. For example, we have tried the facial recognition demo program for the Google AIY kit and tried the Pupil Capture program for the Pupil head mounted tracker. Additionally, our design project is heavily focused on software. Consequently, a lot of the software is comprised of open source algorithms, which we need to test. Therefore, whenever we want to incorporate an open source algorithm into our software, we make sure it works by running it in isolation.

Section 16: Detailed Budget Plan

Detailed Budget Plan

Task No. 1		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Project Management												Break			Break	Break	Break	Break	
Labor																			
Sr. Engineer W	$73.81	0	2	3	2	1	2	3	4	1	1	0	3	4	0	0	0	0	2
Engineer - CmpE A	$38.33	0	2	2	2	1	1	1	4	1	1	0	3	4	0	0	0	0	1
Engineer - CmpE M	$38.33	0	2	2	2	1	1	1	4	1	0	0	1	4	0	0	0	0	1
Engineer - EE P	$35.94	0	2	1	1.5	1	1	1	4	1	0	0	2	4	0	0	0	0	1
Engineer - BME M	$35.94	0	2	2	2	1	1	1	4	0	1	0	2	4	0	0	0	0	1
Engineer - BME J	$35.94	0	2	2	2	0	1	0	6	3	0	0	3	2	0	0	0	0	0
Direct Labor Cost		0.00	516.58	554.45	498.61	222.35	332.10	369.97	1105.04	294.23	148.08	0.00	626.33	961.28	0.00	0.00	0.00	0.00	296.16
Labor Overhead	40.00%	0.00	206.63	221.78	199.44	88.94	132.84	147.99	442.02	117.69	59.23	0.00	250.53	384.51	0.00	0.00	0.00	0.00	118.46
Other Direct Cost (ODC)																			
Nonlabor - consulting		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Materials		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ODC Overhead	33.33%	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Direct Cost		0.00	723.21	776.23	698.05	311.29	464.94	517.96	1547.06	411.92	207.31	0.00	876.86	1345.79	0.00	0.00	0.00	0.00	414.62
G & A	20.00%	0.00	144.64	155.25	139.61	62.26	92.99	103.59	309.41	82.38	41.46	0.00	175.37	269.16	0.00	0.00	0.00	0.00	82.92
Total Costs		0.00	867.85	931.48	837.66	373.55	557.93	621.55	1856.47	494.31	248.77	0.00	1052.23	1614.95	0.00	0.00	0.00	0.00	497.55
Profit	10.00%	0.00	86.79	93.15	83.77	37.35	55.79	62.15	185.65	49.43	24.88	0.00	105.22	161.50	0.00	0.00	0.00	0.00	49.75
Costs + Profit		0.00	954.64	1024.62	921.43	410.90	613.72	683.70	2042.11	543.74	273.65	0.00	1157.46	1776.45	0.00	0.00	0.00	0.00	547.30

19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Total	
							Break								Hrs	Cost
1	1	2	2	1	1	1	0	2	3	3	3	3	3	3	57	4207.17
0	0	1	1	0	0	0	0	1	1	1	1	1	1	1	32	1226.56
0	0	1	1	0	0	0	0	1	1	1	1	1	1	1	29	1111.57
0	0	1	1	0	0	0	0	1	1	1	1	1	1	1	28.5	1024.29
1	0	1	1	0	0	0	0	1	1	1	1	1	1	1	31	1114.14
0	0	1	1	0	0	0	0	1	1	1	1	1	1	1	30	1078.20
109.75	73.81	332.10	332.10	73.81	73.81	73.81	0.00	332.10	405.91	405.91	405.91	405.91	405.91	405.91		9761.93
43.90	29.52	132.84	132.84	29.52	29.52	29.52	0.00	132.84	162.36	162.36	162.36	162.36	162.36	162.36		3904.77
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
153.65	103.33	464.94	464.94	103.33	103.33	103.33	0.00	464.94	568.27	568.27	568.27	568.27	568.27	568.27		13666.70
30.73	20.67	92.99	92.99	20.67	20.67	20.67	0.00	92.99	113.65	113.65	113.65	113.65	113.65	113.65		2733.34
184.38	124.00	557.93	557.93	124.00	124.00	124.00	0.00	557.93	681.93	681.93	681.93	681.93	681.93	681.93		16400.04
18.44	12.40	55.79	55.79	12.40	12.40	12.40	0.00	55.79	68.19	68.19	68.19	68.19	68.19	68.19		1640.00
202.82	136.40	613.72	613.72	136.40	136.40	136.40	0.00	613.72	750.12	750.12	750.12	750.12	750.12	750.12		**18040.05**

Task No. 2		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Hardware: Electrical												Break			Break	Break	Break	Break	
Labor																			
Sr. Engineer	$73.81	0	0	0	2	0	0	0	1	1	1	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - EE	$35.94	0	0	1	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0
Engineer - BME	$35.94	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1
Engineer - BME	$35.94	0	0	0	1	2	0	1	0	0	1	0	0	0	0	0	0	0	0
Direct Labor Cost		0.00	0.00	35.94	372.82	110.21	35.94	71.88	109.75	73.81	181.63	0.00	0.00	0.00	0.00	0.00	0.00	0.00	35.94
Labor Overhead	40.00%	0.00	0.00	14.38	149.13	44.08	14.38	28.75	43.90	29.52	72.65	0.00	0.00	0.00	0.00	0.00	0.00	0.00	14.38
Other Direct Cost (ODC)																			
Nonlabor - consulting		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Materials		0.00	0.00	179.98	26.66	5.00	12.99	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ODC Overhead	33.33%	0.00	0.00	59.99	8.89	1.67	4.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Direct Cost		0.00	0.00	290.29	557.49	160.96	67.64	100.63	153.65	103.33	254.28	0.00	0.00	0.00	0.00	0.00	0.00	0.00	50.32
G & A	20.00%	0.00	0.00	58.06	111.50	32.19	13.53	20.13	30.73	20.67	50.86	0.00	0.00	0.00	0.00	0.00	0.00	0.00	10.06
Total Costs		0.00	0.00	348.35	668.99	193.15	81.16	120.76	184.38	124.00	305.14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	60.38
Profit	10.00%	0.00	0.00	34.83	66.90	19.32	8.12	12.08	18.44	12.40	30.51	0.00	0.00	0.00	0.00	0.00	0.00	0.00	6.04
Costs + Profit		0.00	0.00	383.18	735.89	212.47	89.28	132.83	202.82	136.40	335.65	0.00	0.00	0.00	0.00	0.00	0.00	0.00	66.42

19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Total	
							Break								Hrs	Cost
2	0	0	0	0	0	0	0	0	2	2	2	2	2	2	19	1402.39
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	3	114.99
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	114.99
0	2	0	0	0	0	2	0	2.5	3	3	3	3	3	3	29.5	1060.23
2	1	0	0	0	0	2	0	2.5	2	2	2	2	2	2	22.5	808.65
0	0	0	0	0	0	0	0	0	2	2	2	2	2	2	17	610.98
219.50	146.15	0.00	0.00	0.00	0.00	143.76	0.00	179.70	399.20	399.20	399.20	399.20	399.20	399.20		4112.23
87.80	58.46	0.00	0.00	0.00	0.00	57.50	0.00	71.88	159.68	159.68	159.68	159.68	159.68	159.68		1644.89
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	35.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		259.63
0.00	0.00	0.00	0.00	0.00	0.00	11.67	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		86.54
307.30	204.61	0.00	0.00	0.00	0.00	247.93	0.00	251.58	558.88	558.88	558.88	558.88	558.88	558.88		6103.30
61.46	40.92	0.00	0.00	0.00	0.00	49.59	0.00	50.32	111.78	111.78	111.78	111.78	111.78	111.78		1220.66
368.76	245.53	0.00	0.00	0.00	0.00	297.52	0.00	301.90	670.66	670.66	670.66	670.66	670.66	670.66		7323.95
36.88	24.55	0.00	0.00	0.00	0.00	29.75	0.00	30.19	67.07	67.07	67.07	67.07	67.07	67.07		732.40
405.64	270.09	0.00	0.00	0.00	0.00	327.27	0.00	332.09	737.72	737.72	737.72	737.72	737.72	737.72		**8056.35**

Task No. 3		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Software: Eye Tracking												Break			Break	Break	Break	Break	
Labor																			
Sr. Engineer	$73.81	0	0	0	0	2	0	2	2	0	0	0	0	0	0	0	0	0	2
Engineer - CmpE	$38.33	0	0	0	0	2	5	6	4	3	10	0	0	4	4	8	0	0	6
Engineer - CmpE	$38.33	0	0	0	0	1	4	6	4	3	3	0	2	0	0	0	0	0	4
Engineer - EE	$35.94	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - BME	$35.94	0	0	0	0	1	3	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - BME	$35.94	0	0	0	0	0	2	0	0	0	0	0	1	0	0	0	0	0	0
Direct Labor Cost		0.00	0.00	0.00	0.00	334.49	560.61	607.58	454.26	229.98	498.29	0.00	112.60	153.32	153.32	306.64	0.00	0.00	530.92
Labor Overhead	40.00%	0.00	0.00	0.00	0.00	133.80	224.24	243.03	181.70	91.99	199.32	0.00	45.04	61.33	61.33	122.66	0.00	0.00	212.37
Other Direct Cost (ODC)																			
Nonlabor - consulting		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Materials		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ODC Overhead	33.33%	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Direct Cost		0.00	0.00	0.00	0.00	468.29	784.85	850.61	635.96	321.97	697.61	0.00	157.64	214.65	214.65	429.30	0.00	0.00	743.29
G & A	20.00%	0.00	0.00	0.00	0.00	93.66	156.97	170.12	127.19	64.39	139.52	0.00	31.53	42.93	42.93	85.86	0.00	0.00	148.66
Total Costs		0.00	0.00	0.00	0.00	561.94	941.82	1020.73	763.16	386.37	837.13	0.00	189.17	257.58	257.58	515.16	0.00	0.00	891.95
Profit	10.00%	0.00	0.00	0.00	0.00	56.19	94.18	102.07	76.32	38.64	83.71	0.00	18.92	25.76	25.76	51.52	0.00	0.00	89.19
Costs + Profit		0.00	0.00	0.00	0.00	618.14	1036.01	1122.81	839.47	425.00	920.84	0.00	208.08	283.34	283.34	566.67	0.00	0.00	981.14

19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Total	
							Break								Hrs	Cost
2	3	1	2	3	3	3	0	3	0	0	0	0	0	0	28	2066.68
6	4	5	7	6	5	7	0	6	4	4	4	4	4	4	122	4676.26
10	5	5	5	5	5	4	0	5	4	4	4	4	4	4	95	3641.35
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	71.88
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	143.76
0	0	0	0	3	3	2	0	0	0	0	0	0	0	0	11	395.34
760.90	566.40	457.11	607.58	750.88	712.55	714.94	0.00	643.06	306.64	306.64	306.64	306.64	306.64	306.64		10995.27
304.36	226.56	182.84	243.03	300.35	285.02	285.98	0.00	257.22	122.66	122.66	122.66	122.66	122.66	122.66		4398.11
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
1065.26	792.96	639.95	850.61	1051.23	997.57	1000.92	0.00	900.28	429.30	429.30	429.30	429.30	429.30	429.30	0.00	15393.38
213.05	158.59	127.99	170.12	210.25	199.51	200.18	0.00	180.06	85.86	85.86	85.86	85.86	85.86	85.86		3078.68
1278.31	951.55	767.94	1020.73	1261.48	1197.08	1201.10	0.00	1080.34	515.16	515.16	515.16	515.16	515.16	515.16		18472.05
127.83	95.16	76.79	102.07	126.15	119.71	120.11	0.00	108.03	51.52	51.52	51.52	51.52	51.52	51.52		1847.21
1406.14	1046.71	844.74	1122.81	1387.63	1316.79	1321.21	0.00	1188.37	566.67	566.67	566.67	566.67	566.67	566.67		**20319.26**

Task No. 4		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Software: Camera Communications												Break			Break	Break	Break	Break	
Labor																			
Sr. Engineer	$73.81	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - EE	$35.94	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
Engineer - BME	$35.94	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	0	2
Engineer - BME	$35.94	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Direct Labor Cost		0.00	0.00	0.00	0.00	0.00	0.00	35.94	71.88	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	251.58
Labor Overhead	40.00%	0.00	0.00	0.00	0.00	0.00	0.00	14.38	28.75	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	100.63
Other Direct Cost (ODC)																			
Nonlabor - consulting		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Materials		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ODC Overhead	33.33%	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Direct Cost		0.00	0.00	0.00	0.00	0.00	0.00	50.32	100.63	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	352.21
G & A	20.00%	0.00	0.00	0.00	0.00	0.00	0.00	10.06	20.13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	70.44
Total Costs		0.00	0.00	0.00	0.00	0.00	0.00	60.38	120.76	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	422.65
Profit	10.00%	0.00	0.00	0.00	0.00	0.00	0.00	6.04	12.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	42.27
Costs + Profit		0.00	0.00	0.00	0.00	0.00	0.00	66.42	132.83	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	464.92

19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Total	
							Break								Hrs	Cost
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00
0	2	0	0	0	0	0	0	0	1	1	1	1	1	1	8	306.64
0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	6	229.98
1	4	0	0	2	2	1.5	0	2	0.5	0.5	0.5	0.5	0.5	0.5	20.5	736.77
0	2	0	0	1	2	1.5	0	2	0	0	0	0	0	0	13.5	485.19
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	35.94
35.94	292.30	0.00	0.00	107.82	143.76	143.76	0.00	143.76	94.63	94.63	94.63	94.63	94.63	94.63		1794.52
14.38	116.92	0.00	0.00	43.13	57.50	57.50	0.00	57.50	37.85	37.85	37.85	37.85	37.85	37.85		717.81
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
50.32	409.22	0.00	0.00	150.95	201.26	201.26	0.00	201.26	132.48	132.48	132.48	132.48	132.48	132.48		2512.33
10.06	81.84	0.00	0.00	30.19	40.25	40.25	0.00	40.25	26.50	26.50	26.50	26.50	26.50	26.50		502.47
60.38	491.06	0.00	0.00	181.14	241.52	241.52	0.00	241.52	158.98	158.98	158.98	158.98	158.98	158.98		3014.79
6.04	49.11	0.00	0.00	18.11	24.15	24.15	0.00	24.15	15.90	15.90	15.90	15.90	15.90	15.90		301.48
66.42	540.17	0.00	0.00	199.25	265.67	265.67	0.00	265.67	174.88	174.88	174.88	174.88	174.88	174.88		**3316.27**

Task No. 5		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Hardware: Mechanical												Break			Break	Break	Break	Break	
Labor																			
Sr. Engineer	$73.81	0	0	0	0	0	0	2	2	2	3	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - EE	$35.94	0	0	0	0	0	2	3	3	2	2	0	0	0	0	0	0	0	0
Engineer - BME	$35.94	0	0	0	0	0	0	2	2	2	3	0	1	0	0	0	0	0	2
Engineer - BME	$35.94	0	0	0	0	0	1	2	0	0	3	0	0	0	0	0	0	0	1.5
Direct Labor Cost		0.00	0.00	0.00	0.00	0.00	107.82	399.20	327.32	291.38	508.95	0.00	35.94	0.00	0.00	0.00	0.00	0.00	125.79
Labor Overhead	40.00%	0.00	0.00	0.00	0.00	0.00	43.13	159.68	130.93	116.55	203.58	0.00	14.38	0.00	0.00	0.00	0.00	0.00	50.32
Other Direct Cost (ODC)																			
Nonlabor - consulting		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Materials		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ODC Overhead	33.33%	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Direct Cost		0.00	0.00	0.00	0.00	0.00	150.95	558.88	458.25	407.93	712.53	0.00	50.32	0.00	0.00	0.00	0.00	0.00	176.11
G & A	20.00%	0.00	0.00	0.00	0.00	0.00	30.19	111.78	91.65	81.59	142.51	0.00	10.06	0.00	0.00	0.00	0.00	0.00	35.22
Total Costs		0.00	0.00	0.00	0.00	0.00	181.14	670.66	549.90	489.52	855.04	0.00	60.38	0.00	0.00	0.00	0.00	0.00	211.33
Profit	10.00%	0.00	0.00	0.00	0.00	0.00	18.11	67.07	54.99	48.95	85.50	0.00	6.04	0.00	0.00	0.00	0.00	0.00	21.13
Costs + Profit		0.00	0.00	0.00	0.00	0.00	199.25	737.72	604.89	538.47	940.54	0.00	66.42	0.00	0.00	0.00	0.00	0.00	232.46

19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Total	
							Break								Hrs	Cost
0	0	1	0	0	0	0	0	0	2	2	2	2	2	2	22	1623.82
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00
0	0	3	2	1	0	0	0	0	0	0	0	0	0	0	18	646.92
0	3	3	3	2	0	0	0	0	2	2	2	2	2	2	35	1257.90
1	1.5	3	3	0	0	0	0	2	2	2	2	2	2	2	30	1078.20
35.94	161.73	397.27	287.52	107.82	0.00	0.00	0.00	71.88	291.38	291.38	291.38	291.38	291.38	291.38		4606.84
14.38	64.69	158.91	115.01	43.13	0.00	0.00	0.00	28.75	116.55	116.55	116.55	116.55	116.55	116.55		1842.74
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
50.32	226.42	556.18	402.53	150.95	0.00	0.00	0.00	100.63	407.93	407.93	407.93	407.93	407.93	407.93		6449.58
10.06	45.28	111.24	80.51	30.19	0.00	0.00	0.00	20.13	81.59	81.59	81.59	81.59	81.59	81.59		1289.92
60.38	271.71	667.41	483.03	181.14	0.00	0.00	0.00	120.76	489.52	489.52	489.52	489.52	489.52	489.52		7739.49
6.04	27.17	66.74	48.30	18.11	0.00	0.00	0.00	12.08	48.95	48.95	48.95	48.95	48.95	48.95		773.95
66.42	298.88	734.15	531.34	199.25	0.00	0.00	0.00	132.83	538.47	538.47	538.47	538.47	538.47	538.47		**8513.44**

B. Detailed Budget

Task No. 1		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Project Management												Break			Break	Break	Break	Break	
Labor																			
Sr. Engineer W	$73.81	0	2	3	2	1	2	3	4	1	1	0	3	4	0	0	0	0	2
Engineer - CmpE A	$38.33	0	2	2	2	1	1	1	4	1	1	0	3	4	0	0	0	0	1
Engineer - CmpE M	$38.33	0	2	2	2	1	1	1	4	1	0	0	1	4	0	0	0	0	1
Engineer - EE P	$35.94	0	2	1	1.5	1	1	1	4	1	0	0	2	4	0	0	0	0	1
Engineer - BME M	$35.94	0	2	2	2	1	1	1	4	0	1	0	2	4	0	0	0	0	1
Engineer - BME J	$35.94	0	2	2	2	0	1	0	6	3	0	0	3	2	0	0	0	0	0
Direct Labor Cost		0.00	516.58	554.45	498.61	222.35	332.10	369.97	1105.04	294.23	148.08	0.00	626.33	961.28	0.00	0.00	0.00	0.00	296.16
Labor Overhead	40.00%	0.00	206.63	221.78	199.44	88.94	132.84	147.99	442.02	117.69	59.23	0.00	250.53	384.51	0.00	0.00	0.00	0.00	118.46
Other Direct Cost (ODC)																			
Nonlabor - consulting		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Materials		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ODC Overhead	33.33%	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Direct Cost		0.00	723.21	776.23	698.05	311.29	464.94	517.96	1547.06	411.92	207.31	0.00	876.86	1345.79	0.00	0.00	0.00	0.00	414.62
G & A	20.00%	0.00	144.64	155.25	139.61	62.26	92.99	103.59	309.41	82.38	41.46	0.00	175.37	269.16	0.00	0.00	0.00	0.00	82.92
Total Costs		0.00	867.85	931.48	837.66	373.55	557.93	621.55	1856.47	494.31	248.77	0.00	1052.23	1614.95	0.00	0.00	0.00	0.00	497.55
Profit	10.00%	0.00	86.79	93.15	83.77	37.35	55.79	62.15	185.65	49.43	24.88	0.00	105.22	161.50	0.00	0.00	0.00	0.00	49.75
Costs + Profit		0.00	954.64	1024.62	921.43	410.90	613.72	683.70	2042.11	543.74	273.65	0.00	1157.46	1776.45	0.00	0.00	0.00	0.00	547.30

19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Total	
							Break								Hrs	Cost
1	1	2	2	1	1	1	0	2	1	2	1	1	5.5	4	53.5	3948.84
0	0	1	1	0	0	0	0	1	0	2	0	0	5.5	4	37.5	1437.38
0	0	1	1	0	0	0	0	1	0	2	0	0	5.5	4	34.5	1322.39
0	0	1	1	0	0	0	0	1	0	2	0	0	5.5	4	34	1221.96
1	0	1	1	0	0	0	0	1	0	2	0	0	5.5	4	36.5	1311.81
0	0	1	1	0	0	0	0	1	0	2	0	0	5.5	4	35.5	1275.87
109.75	73.81	332.10	332.10	73.81	73.81	73.81	0.00	332.10	73.81	516.58	73.81	73.81	1420.60	1033.16		10518.24
43.90	29.52	132.84	132.84	29.52	29.52	29.52	0.00	132.84	29.52	206.63	29.52	29.52	568.24	413.26		4207.29
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
153.65	103.33	464.94	464.94	103.33	103.33	103.33	0.00	464.94	103.33	723.21	103.33	103.33	1988.83	1446.42		14725.53
30.73	20.67	92.99	92.99	20.67	20.67	20.67	0.00	92.99	20.67	144.64	20.67	20.67	397.77	289.28		2945.11
184.38	124.00	557.93	557.93	124.00	124.00	124.00	0.00	557.93	124.00	867.85	124.00	124.00	2386.60	1735.71		17670.63
18.44	12.40	55.79	55.79	12.40	12.40	12.40	0.00	55.79	12.40	86.79	12.40	12.40	238.66	173.57		1767.06
202.82	136.40	613.72	613.72	136.40	136.40	136.40	0.00	613.72	136.40	954.64	136.40	136.40	2625.26	1909.28		**19437.70**

Task No. 2		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Hardware: Electrical												Break			Break	Break	Break	Break	
Labor																			
Sr. Engineer	$73.81	0	0	0	2	0	0	0	1	1	1	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - EE	$35.94	0	0	1	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0
Engineer - BME	$35.94	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1
Engineer - BME	$35.94	0	0	0	1	2	0	1	0	0	1	0	0	0	0	0	0	0	0
Direct Labor Cost		0.00	0.00	35.94	372.82	110.21	35.94	71.88	109.75	73.81	181.63	0.00	0.00	0.00	0.00	0.00	0.00	0.00	35.94
Labor Overhead	40.00%	0.00	0.00	14.38	149.13	44.08	14.38	28.75	43.90	29.52	72.65	0.00	0.00	0.00	0.00	0.00	0.00	0.00	14.38
Other Direct Cost (ODC)																			
Nonlabor - consulting		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Materials		0.00	0.00	179.98	26.66	5.00	12.99	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ODC Overhead	33.33%	0.00	0.00	59.99	8.89	1.67	4.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Direct Cost		0.00	0.00	290.29	557.49	160.96	67.64	100.63	153.65	103.33	254.28	0.00	0.00	0.00	0.00	0.00	0.00	0.00	50.32
G & A	20.00%	0.00	0.00	58.06	111.50	32.19	13.53	20.13	30.73	20.67	50.86	0.00	0.00	0.00	0.00	0.00	0.00	0.00	10.06
Total Costs		0.00	0.00	348.35	668.99	193.15	81.16	120.76	184.38	124.00	305.14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	60.38
Profit	10.00%	0.00	0.00	34.83	66.90	19.32	8.12	12.08	18.44	12.40	30.51	0.00	0.00	0.00	0.00	0.00	0.00	0.00	6.04
Costs + Profit		0.00	0.00	383.18	735.89	212.47	89.28	132.83	202.82	136.40	335.65	0.00	0.00	0.00	0.00	0.00	0.00	0.00	66.42

19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Total	
							Break								Hrs	Cost
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	516.67
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	3	114.99
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	114.99
0	2	0	0	0	0	2	0	2.5	0	0	0	0	0	0	11.5	413.31
2	1	0	0	0	0	2	0	2.5	0	0	0	0	0	0	10.5	377.37
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	179.70
219.50	146.15	0.00	0.00	0.00	0.00	143.76	0.00	179.70	0.00	0.00	0.00	0.00	0.00	0.00		1717.03
87.80	58.46	0.00	0.00	0.00	0.00	57.50	0.00	71.88	0.00	0.00	0.00	0.00	0.00	0.00		686.81
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	35.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		259.63
0.00	0.00	0.00	0.00	0.00	0.00	11.67	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		86.54
307.30	204.61	0.00	0.00	0.00	0.00	247.93	0.00	251.58	0.00	0.00	0.00	0.00	0.00	0.00		2750.02
61.46	40.92	0.00	0.00	0.00	0.00	49.59	0.00	50.32	0.00	0.00	0.00	0.00	0.00	0.00		550.00
368.76	245.53	0.00	0.00	0.00	0.00	297.52	0.00	301.90	0.00	0.00	0.00	0.00	0.00	0.00		3300.02
36.88	24.55	0.00	0.00	0.00	0.00	29.75	0.00	30.19	0.00	0.00	0.00	0.00	0.00	0.00		330.00
405.64	270.09	0.00	0.00	0.00	0.00	327.27	0.00	332.09	0.00	0.00	0.00	0.00	0.00	0.00		**3630.02**

Task No. 3		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Software: Eye Tracking												Break			Break	Break	Break	Break	
Labor																			
Sr. Engineer	$73.81	0	0	0	0	2	0	2	2	0	0	0	0	0	0	0	0	0	2
Engineer - CmpE	$38.33	0	0	0	0	2	5	6	4	3	10	0	0	4	4	8	0	0	6
Engineer - CmpE	$38.33	0	0	0	0	1	4	6	4	3	3	0	2	0	0	0	0	0	4
Engineer - EE	$35.94	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - BME	$35.94	0	0	0	0	1	3	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - BME	$35.94	0	0	0	0	0	2	0	0	0	0	0	1	0	0	0	0	0	0
Direct Labor Cost		0.00	0.00	0.00	0.00	334.49	560.61	607.58	454.26	229.98	498.29	0.00	112.60	153.32	153.32	306.64	0.00	0.00	530.92
Labor Overhead	40.00%	0.00	0.00	0.00	0.00	133.80	224.24	243.03	181.70	91.99	199.32	0.00	45.04	61.33	61.33	122.66	0.00	0.00	212.37
Other Direct Cost (ODC)																			
Nonlabor - consulting		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Materials		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ODC Overhead	33.33%	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Direct Cost		0.00	0.00	0.00	0.00	468.29	784.85	850.61	635.96	321.97	697.61	0.00	157.64	214.65	214.65	429.30	0.00	0.00	743.29
G & A	20.00%	0.00	0.00	0.00	0.00	93.66	156.97	170.12	127.19	64.39	139.52	0.00	31.53	42.93	42.93	85.86	0.00	0.00	148.66
Total Costs		0.00	0.00	0.00	0.00	561.94	941.82	1020.73	763.16	386.37	837.13	0.00	189.17	257.58	257.58	515.16	0.00	0.00	891.95
Profit	10.00%	0.00	0.00	0.00	0.00	56.19	94.18	102.07	76.32	38.64	83.71	0.00	18.92	25.76	25.76	51.52	0.00	0.00	89.19
Costs + Profit		0.00	0.00	0.00	0.00	618.14	1036.01	1122.81	839.47	425.00	920.84	0.00	208.08	283.34	283.34	566.67	0.00	0.00	981.14

19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Total	
							Break								Hrs	Cost
2	3	**1**	**2**	**3**	**3**	**3**	**0**	**3**	**2**	**1**	**1.5**	**4**	**0**	**0**	36.5	2694.07
6	4	**5**	**7**	**6**	**5**	**7**	**0**	**6**	**8**	**3**	**7**	**8.5**	**5.5**	**0**	130	4982.90
10	5	**5**	**5**	**5**	**5**	**4**	**0**	**5**	**6**	**2.5**	**6**	**5**	**1**	**0**	91.5	3507.20
0	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**5.5**	**3.5**	**6**	**5.5**	**0**	**0**	22.5	808.65
0	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**5.5**	**3.5**	**6.5**	**6**	**0.5**	**0**	26	934.44
0	0	**0**	**0**	**3**	**3**	**2**	**0**	**0**	**0**	**1**	**6**	**3**	**1.5**	**0**	22.5	808.65
760.90	566.40	457.11	607.58	750.88	712.55	714.94	0.00	643.06	1079.58	572.15	1273.90	1333.83	321.03	0.00		13735.90
304.36	226.56	182.84	243.03	300.35	285.02	285.98	0.00	257.22	431.83	228.86	509.56	533.53	128.41	0.00		5494.36
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
1065.26	792.96	639.95	850.61	1051.23	997.57	1000.92	0.00	900.28	1511.41	801.00	1783.45	1867.36	449.44	0.00	0.00	19230.26
213.05	158.59	127.99	170.12	210.25	199.51	200.18	0.00	180.06	302.28	160.20	356.69	373.47	89.89	0.00		3846.05
1278.31	951.55	767.94	1020.73	1261.48	1197.08	1201.10	0.00	1080.34	1813.69	961.20	2140.14	2240.83	539.32	0.00		23076.31
127.83	95.16	76.79	102.07	126.15	119.71	120.11	0.00	108.03	181.37	96.12	214.01	224.08	53.93	0.00		2307.63
1406.14	1046.71	844.74	1122.81	1387.63	1316.79	1321.21	0.00	1188.37	1995.06	1057.32	2354.16	2464.91	593.25	0.00		**25383.94**

Task No. 4		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Software: Camera Communications												Break			Break	Break	Break	Break	
Labor																			
Sr. Engineer	$73.81	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - EE	$35.94	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
Engineer - BME	$35.94	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	0	2
Engineer - BME	$35.94	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Direct Labor Cost		0.00	0.00	0.00	0.00	0.00	0.00	35.94	71.88	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	251.58
Labor Overhead	40.00%	0.00	0.00	0.00	0.00	0.00	0.00	14.38	28.75	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	100.63
Other Direct Cost (ODC)																			
Nonlabor - consulting		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Materials		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ODC Overhead	33.33%	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Direct Cost		0.00	0.00	0.00	0.00	0.00	0.00	50.32	100.63	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	352.21
G & A	20.00%	0.00	0.00	0.00	0.00	0.00	0.00	10.06	20.13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	70.44
Total Costs		0.00	0.00	0.00	0.00	0.00	0.00	60.38	120.76	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	422.65
Profit	10.00%	0.00	0.00	0.00	0.00	0.00	0.00	6.04	12.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	42.27
Costs + Profit		0.00	0.00	0.00	0.00	0.00	0.00	66.42	132.83	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	464.92

19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Total	
							Break								Hrs	Cost
0	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	0	0.00
0	2	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	2	76.66
0	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	0	0.00
1	4	**0**	**0**	**2**	**2**	**1.5**	**0**	**2**	**0**	**0**	**0**	**0**	**0**	**0**	17.5	628.95
0	2	**0**	**0**	**1**	**2**	**1.5**	**0**	**2**	**0**	**0**	**0**	**0**	**0**	**0**	13.5	485.19
0	0	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	1	35.94
35.94	292.30	0.00	0.00	107.82	143.76	143.76	0.00	143.76	0.00	0.00	0.00	0.00	0.00	0.00		1226.74
14.38	116.92	0.00	0.00	43.13	57.50	57.50	0.00	57.50	0.00	0.00	0.00	0.00	0.00	0.00		490.70
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
50.32	409.22	0.00	0.00	150.95	201.26	201.26	0.00	201.26	0.00	0.00	0.00	0.00	0.00	0.00		1717.44
10.06	81.84	0.00	0.00	30.19	40.25	40.25	0.00	40.25	0.00	0.00	0.00	0.00	0.00	0.00		343.49
60.38	491.06	0.00	0.00	181.14	241.52	241.52	0.00	241.52	0.00	0.00	0.00	0.00	0.00	0.00		2060.92
6.04	49.11	0.00	0.00	18.11	24.15	24.15	0.00	24.15	0.00	0.00	0.00	0.00	0.00	0.00		206.09
66.42	540.17	0.00	0.00	199.25	265.67	265.67	0.00	265.67	0.00	0.00	0.00	0.00	0.00	0.00		**2267.02**

19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	Total	
							Break								Hrs	Cost
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	10	738.10
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.00
0	0	3	2	1	0	0	0	0	0	0	0	0	0	0	18	646.92
0	3	3	3	2	0	0	0	0	0	0	0	0	0	0	23	826.62
1	1.5	3	3	0	0	0	0	2	0	0	0	0	0	0	18	646.92
35.94	161.73	397.27	287.52	107.82	0.00	0.00	0.00	71.88	0.00	0.00	0.00	0.00	0.00	0.00		2858.56
14.38	64.69	158.91	115.01	43.13	0.00	0.00	0.00	28.75	0.00	0.00	0.00	0.00	0.00	0.00		1143.42
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
50.32	226.42	556.18	402.53	150.95	0.00	0.00	0.00	100.63	0.00	0.00	0.00	0.00	0.00	0.00		4001.98
10.06	45.28	111.24	80.51	30.19	0.00	0.00	0.00	20.13	0.00	0.00	0.00	0.00	0.00	0.00		800.40
60.38	271.71	667.41	483.03	181.14	0.00	0.00	0.00	120.76	0.00	0.00	0.00	0.00	0.00	0.00		4802.38
6.04	27.17	66.74	48.30	18.11	0.00	0.00	0.00	12.08	0.00	0.00	0.00	0.00	0.00	0.00		480.24
66.42	298.88	734.15	531.34	199.25	0.00	0.00	0.00	132.83	0.00	0.00	0.00	0.00	0.00	0.00		**5282.62**

Task No. 5		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Hardware: Mechanical												Break			Break	Break	Break	Break	
Labor																			
Sr. Engineer	$73.81	0	0	0	0	0	0	2	2	2	3	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - CmpE	$38.33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Engineer - EE	$35.94	0	0	0	0	0	2	3	3	2	2	0	0	0	0	0	0	0	0
Engineer - BME	$35.94	0	0	0	0	0	0	2	2	2	3	0	1	0	0	0	0	0	2
Engineer - BME	$35.94	0	0	0	0	0	1	2	0	0	3	0	0	0	0	0	0	0	1.5
Direct Labor Cost		0.00	0.00	0.00	0.00	0.00	107.82	399.20	327.32	291.38	508.95	0.00	35.94	0.00	0.00	0.00	0.00	0.00	125.79
Labor Overhead	40.00%	0.00	0.00	0.00	0.00	0.00	43.13	159.68	130.93	116.55	203.58	0.00	14.38	0.00	0.00	0.00	0.00	0.00	50.32
Other Direct Cost (ODC)																			
Nonlabor - consulting		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Materials		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ODC Overhead	33.33%	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total Direct Cost		0.00	0.00	0.00	0.00	0.00	150.95	558.88	458.25	407.93	712.53	0.00	50.32	0.00	0.00	0.00	0.00	0.00	176.11
G & A	20.00%	0.00	0.00	0.00	0.00	0.00	30.19	111.78	91.65	81.59	142.51	0.00	10.06	0.00	0.00	0.00	0.00	0.00	35.22
Total Costs		0.00	0.00	0.00	0.00	0.00	181.14	670.66	549.90	489.52	855.04	0.00	60.38	0.00	0.00	0.00	0.00	0.00	211.33
Profit	10.00%	0.00	0.00	0.00	0.00	0.00	18.11	67.07	54.99	48.95	85.50	0.00	6.04	0.00	0.00	0.00	0.00	0.00	21.13
Costs + Profit		0.00	0.00	0.00	0.00	0.00	199.25	737.72	604.89	538.47	940.54	0.00	66.42	0.00	0.00	0.00	0.00	0.00	232.46

C. Short Bios

Alan Wilms is a senior Computer Engineering major at Vanderbilt University with a minor in engineering management. Alan has experience with modern web development and distributed systems. He has built on his general programming knowledge from upper level courses with multiple internships, including one at Google working on security for Google Cloud Platform and one at Facebook working on internal recruiting tools. Alan is active with VandyHacks, the 36 hour-long annual hackathon for more than 600 hackers hosted in Vanderbilt's Engineering and Science building, as the Director of

Design and was the former Engineering Council President of Student Affairs. Alan will work at Facebook next fall.

Will Lee is a senior Electrical Engineering major at Vanderbilt University with a minor in Computer Science. Will has experience with microcontrollers and circuit design/testing. He attained the relevant knowledge through coursework and personal projects such as constructing a switch controlled LED system. In addition, he has enough knowledge of computer science to understand the integration of Tensorflow in to the project. He also has experience managing and working in a group environment thanks to his extracurricular involvements. He will design hardware for the project while serving as a bridge between computer engineers and electrical engineers in the group.

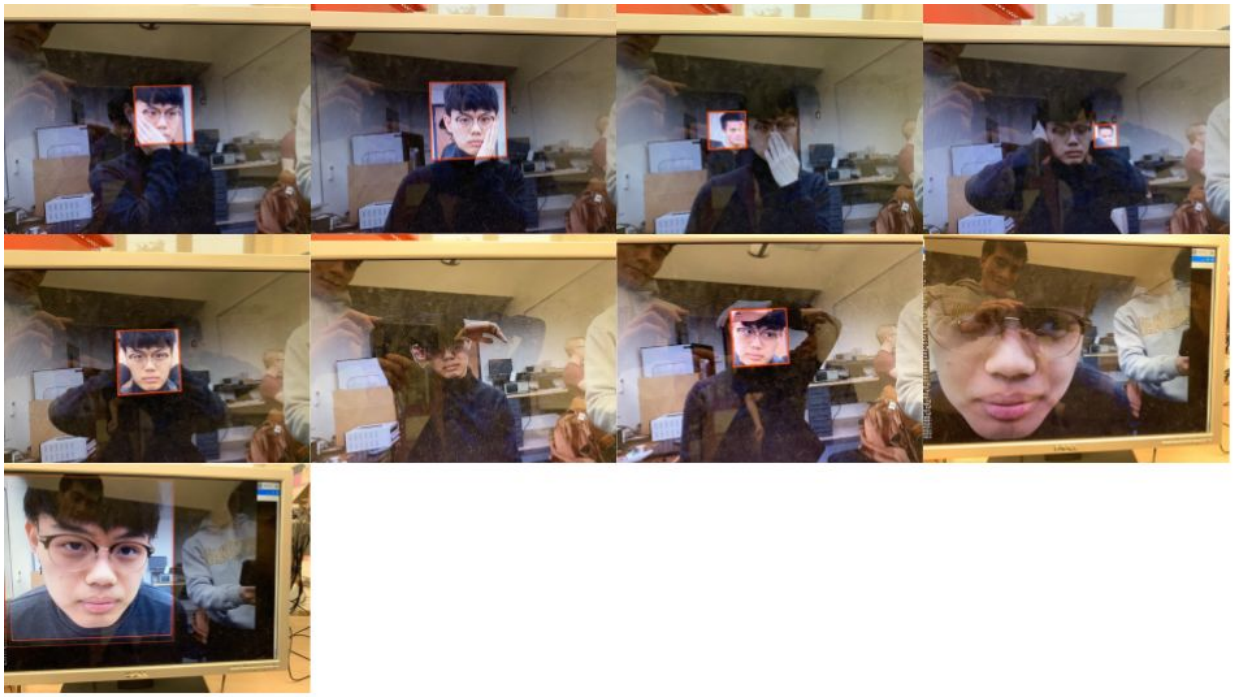
Matthew Kernen is a senior majoring in Computer Engineering and Applied Mathematics at Vanderbilt University. Matthew has experience with web development as well as object oriented and low-level programming languages. He has learned about software development through his computer science depth coursework which he has been able to apply at several internships, most recently at Capital One where he worked on a cloud resource management project. He also has sufficient experience with microcontrollers and electronics through his coursework to understand the electrical part of the hardware for this project. Matthew will be joining Facebook after graduating.

Pam Karwowski is a senior Electrical Engineering major at Vanderbilt University. Pam has experience with signal processing, specifically image processing, as well as the process of designing and testing circuits. Her concentration in computer engineering has provided her with basic knowledge of programming languages that she has built upon in her own time. Additionally, she has sufficient experience with mechanical design for electronic systems, gained from coursework and personal projects. Pam is a leader in several of her organizations, providing her with skills that will be beneficial in a group environment. For this project, Pam will work on the mechanical and electrical aspects of the hardware.

Jefferson Kim is a senior Biomedical and Electrical Engineering major at Vanderbilt University. He has research experience in image processing using Matlab, particularly with peripheral nerve images. He has worked on various projects throughout his education which contribute to a working knowledge of the design process and how to work well in a team. Relevant to this project, his studies have prepared him with a good deal of image and signal processing which will be useful in terms of the data output of the project.

Minh Chung is a senior Biomedical Engineering major at Vanderbilt University. Minh has experience working with eyes in his retinal development research lab. He has also completed image processing projects in MATLAB and Python as well as infrared light projects such as a blood pressure sensor. Through his coursework Minh has a strong background in interfacing light with the human body, which was important in setting up the camera system of the project.

D. Supporting Documents



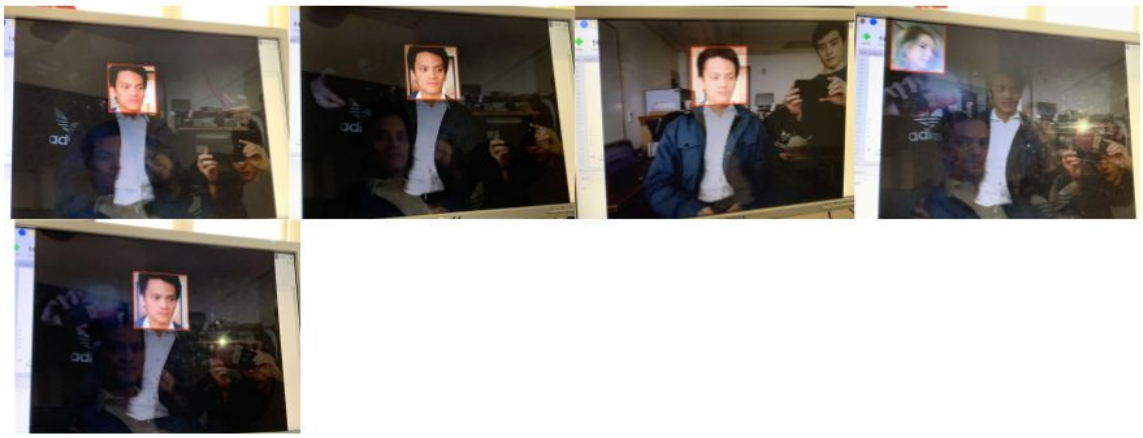


Figure 1. Face Detection Limit Testing

```
#!/usr/bin/env python
# Author: Alan Wilms

import sys
import cv2
import numpy as np
import math
import threading
import time
import random
from scipy import stats


class EyeTracker:
    # define several class variables
    def __init__(self, webcam_index, calibration_sleep_time, measure_accuracy):
        try:
            self.face_cascade = cv2.CascadeClassifier(
                cv2.data.haarcascades + "haarcascade_frontalface_alt.xml")
        except Exception as e:
            print("Could not load face detector:", e)

        try:
            self.eye_cascade = cv2.CascadeClassifier(
                cv2.data.haarcascades + "haarcascade_eye_tree_eyeglasses.xml")
        except Exception as e:
            print("Could not load eye detector:", e)

        cap = cv2.VideoCapture(webcam_index)
        if not cap.isOpened():
            print("Webcam not detected.")

        self.frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
        self.frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
        self.frame_rate = cap.get(cv2.CAP_PROP_FPS)
        print("Video stats: width = {0:.0f} px | height = {1:.0f} px | FPS = {2:.0f}
    frames per second.".format(
            self.frame_width, self.frame_height, self.frame_rate))

        self.sleep_time = calibration_sleep_time
        self.message = ""
```

```
41.         self.calibration_stage = 0
42.         self.calibration_stage1_data = []
43.         self.calibration_stage2_data = []
44.         self.calibration_stage3_data = []
45.         self.calibration_stage4_data = []
46.         self.top = 0
47.         self.bottom = 0
48.         self.left = 0
49.         self.right = 0
50.         self.measure_accuracy = True if measure_accuracy.lower() == "true" or
    measure_accuracy.lower() == "1" else False
51.         self.accuracy_x = self.frame_width // 2
52.         self.accuracy_y = self.frame_height // 2
53.         self.accuracy_distance = []
54.         self.accuracy_quadrant = []
55.
56.         # start calicbration thread
57.         calibration = threading.Thread(target=self.start_calibration, args=())
58.         calibration.daemon = True
59.         calibration.start()
60.
61.         while(True):
62.             # Capture frame-by-frame
63.             ret, frame = cap.read()
64.
65.             if not frame.data:
66.                 break
67.
68.             frame = cv2.flip(frame, 1)
69.             frame = self.process_frames(frame)
70.
71.             # Display the resulting frame
72.             cv2.imshow('Webcam', frame)
73.             # if cv2.waitKey(1) & 0xFF == ord('q'):
74.             #     break
75.
76.             # takes 30 frames per second. if the user presses any button, it stops from
    showing the webcam
77.             if cv2.waitKey(30) >= 0:
78.                 break
79.
80.         # When everything done, release the capture
81.         cap.release()
82.         cv2.destroyAllWindows()
83.
84.     def get_left_eye(self, eyes):
85.         # safety check
86.         assert (len(eyes) >= 1), "no eyes detected for tracking!"
87.         leftmost_x = eyes[0][0]  # x
88.         leftmost_index = 0
89.         # print("eye", leftmost_x)
90.         for i in range(1, len(eyes)):
91.             # print("eye", eyes[i][0])
92.             if (eyes[i][0] < leftmost_x):
93.                 leftmost_x = eyes[i][0]
94.                 leftmost_index = i
95.         # print(leftmost_x)
96.         return eyes[leftmost_index]
97.
98.     def process_frames(self, frame):
99.         grayscale = cv2.equalizeHist(cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY))
```

```
100.
101.            # convert image to grayscale and increase contrast
102.            faces = self.face_cascade.detectMultiScale(
103.                grayscale, scaleFactor=1.1, minNeighbors=2, flags=0 |
    cv2.CASCADE_SCALE_IMAGE, minSize=(150, 150))
104.
105.            # check cannot be "not faces" because using NumPy arrays
106.            if len(faces) == 0:
107.                print("no faces detected")
108.                return self.add_text(frame)
109.
110.            # print(faces)
111.            # face = grayscale[faces[0]]
112.
113.            # detect closest face
114.            closest_face = None
115.            closest_size = None
116.            for face in faces:
117.                size = face[2] * face[3]  # w * h
118.                if closest_face is None or size > closest_size:
119.                    closest_face = face
120.                    closest_size = size
121.
122.            x, y, w, h = closest_face
123.            font = cv2.FONT_HERSHEY_SIMPLEX
124.            cv2.putText(frame, 'Face', (x + w, y + h), font,
125.                        0.5, (0, 0, 139), 2, cv2.LINE_AA)
126.            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 139), 2)
127.            # color_face = frame[y:y + h, x:x + w]
128.            grayscale_face = grayscale[y:y + h, x:x + w]
129.            eyes = self.eye_cascade.detectMultiScale(
130.                grayscale_face, scaleFactor=1.1, minNeighbors=2, flags=0 |
    cv2.CASCADE_SCALE_IMAGE, minSize=(30, 30))
131.            if len(eyes) != 2:
132.                print("no eyes detected")
133.                return self.add_text(frame)
134.
135.            left_eye = self.get_left_eye(eyes)
136.            for eye in eyes:
137.                ex, ey, ew, eh = eye
138.                cv2.rectangle(frame, (x + ex, y + ey), (x + ex + ew, y + ey + eh),
139.                              (0, 140, 255) if eye[0] == left_eye[0] else (0, 0, 139),
    2)
140.
141.            lex, ley, lew, leh = left_eye
142.            left_eye = grayscale_face[ley:ley + leh, lex:lex + lew]
143.            # increase contrast
144.            left_eye = cv2.equalizeHist(left_eye)
145.            # left_eye = cv2.equalizeHist(cv2.cvtColor(left_eye, cv2.COLOR_BGR2GRAY))
146.
147.            thres = cv2.inRange(left_eye, 0, 20)
148.            kernel = np.ones((3, 3), np.uint8)
149.
150.            # processing to remove small noise
151.            # originally set to 2 and 3, respectively
152.            dilation = cv2.dilate(thres, kernel, iterations=2)
153.            erosion = cv2.erode(dilation, kernel, iterations=3)
154.
155.            # find contours
156.            contours, hierarchy = cv2.findContours(
157.                erosion, cv2.RETR_TREE, cv2.CHAIN_APPROX_NONE)
```

```
158.
159.            # algorithm to find pupil and reduce noise of other contours by selecting
    the closest to center
160.            closest_cx = None
161.            closest_cy = None
162.            closest_distance = None
163.            center = (ley + leh//2, lex + lew//2)
164.            if len(contours) >= 1:
165.                M = cv2.moments(contours[0])
166.                if M['m00'] != 0:
167.                    cx = int(M['m10']/M['m00'])
168.                    cy = int(M['m01']/M['m00'])
169.                    for contour in contours:
170.                        # distance between center and potential pupil
171.                        distance = math.sqrt(
172.                            (cy - center[0])**2 + (cx - center[1])**2)
173.                        if closest_distance is None or distance < closest_distance:
174.                            closest_cx = cx
175.                            closest_cy = cy
176.                            closest_distance = distance
177.
178.            if closest_cx is not None and closest_cy is not None:
179.                # base size of pupil to size of eye
180.                cv2.circle(frame, (x + lex + closest_cx, y + ley + closest_cy),
181.                           lew//12, (0, 140, 255), -1)
182.                # process calibration
183.                percentage_x = closest_cx / lew
184.                percentage_y = closest_cy / leh
185.                if self.calibration_stage > 0 and self.calibration_stage < 5:
186.                    # print(self.calibration_stage)
187.                    # print(percentage_x)
188.                    # self.add_calibration_data(
189.                    #     x + lex + closest_cx, y + ley + closest_cy)
190.                    # self.add_calibration_data(closest_cx, closest_cy)
191.                    self.add_calibration_data(percentage_x, percentage_y)
192.
193.                if self.calibration_stage >= 5:
194.                    radius = 10
195.
196.                    gaze_x = ((percentage_x - self.left) /
197.                              (self.right - self.left)) * self.frame_width
198.                    gaze_y = ((percentage_y - self.top) /
199.                              (self.bottom - self.top)) * self.frame_height
200.
201.                    if percentage_x < self.left:
202.                        gaze_x = 0
203.                    elif percentage_x > self.right:
204.                        gaze_x = self.frame_width
205.
206.                    if percentage_y < self.top:
207.                        gaze_y = 0
208.                    elif percentage_y > self.bottom:
209.                        gaze_y = self.frame_height
210.
211.                    # adjusting from the edges
212.                    if gaze_x <= radius:
213.                        gaze_x += radius
214.                    elif gaze_x + radius >= self.frame_width:
215.                        gaze_x -= radius
216.
217.                    if gaze_y <= radius:
```

```
218.                     gaze_y += radius
219.                 elif gaze_y + radius >= self.frame_height:
220.                     gaze_y -= radius
221.
222.                 cv2.circle(frame, (int(round(gaze_x)), int(
223.                     round(gaze_y))), radius, (0, 0, 139), -1)
224.
225.                 # add overlay
226.                 frame = self.add_overlay(frame, gaze_x, gaze_y)
227.
228.             if self.calibration_stage == 6:
229.                 distance = math.sqrt((gaze_x - self.accuracy_x)**2 + (gaze_y -
    self.accuracy_y)**2)
230.                 self.accuracy_distance.append(distance);
231.                 gaze_quadrant = self.find_quadrant(gaze_x, gaze_y)
232.                 accuracy_quadrant = self.find_quadrant(self.accuracy_x,
    self.accuracy_y)
233.                 self.accuracy_quadrant.append((gaze_quadrant, accuracy_quadrant))
234.
235.         return self.add_text(frame)
236.
237.     def find_quadrant(self, point_x, point_y):
238.         mid_x = self.frame_width // 2
239.         mid_y = self.frame_height // 2
240.         if point_x <= mid_x and point_y <= mid_y:
241.             return 1
242.         elif point_x > mid_x and point_y <= mid_y:
243.             return 2
244.         elif point_x > mid_x and point_y > mid_y:
245.             return 3
246.         else:
247.             # point_x <= mid_x and point_y > mid_y
248.             return 4
249.
250.     def add_overlay(self, frame, gaze_x, gaze_y):
251.         overlay = frame.copy()
252.         mid_x = self.frame_width // 2
253.         mid_y = self.frame_height // 2
254.         alpha = 0.3  # Transparency factor.
255.
256.         cv2.rectangle(overlay, (0 if gaze_x <= mid_x else mid_x, 0 if gaze_y <=
    mid_y else mid_y), (mid_x if gaze_x <= mid_x else self.frame_width, mid_y if gaze_y <=
    mid_y else self.frame_height), (0, 200, 0), -1)
257.         # Overlays transparent rectangle over the image
258.         return cv2.addWeighted(overlay, alpha, frame, 1 - alpha, 0)
259.
260.     def add_text(self, frame):
261.         if (self.message != ""):
262.             cv2.putText(frame, self.message,
263.                         (50, self.frame_height - 15),
264.                         cv2.FONT_HERSHEY_SIMPLEX,
265.                         1,
266.                         (255, 255, 255),
267.                         2)
268.
269.         if (self.calibration_stage > 0 and self.calibration_stage <= 6):
270.             radius = random.randint(12, 13)
271.             color = (0, 0, 139)
272.             offset = 20
273.
274.             if self.calibration_stage == 1:
```

```
275.                     cv2.circle(frame, (offset, offset), radius, color, -1)
276.                 elif self.calibration_stage == 2:
277.                     cv2.circle(frame, (self.frame_width -
278.                                        offset, offset), radius, color, -1)
279.                 elif self.calibration_stage == 3:
280.                     cv2.circle(frame, (self.frame_width - offset,
281.                                        self.frame_height - offset), radius, color, -1)
282.                 elif self.calibration_stage == 4:
283.                     cv2.circle(frame, (offset, self.frame_height -
284.                                        offset), radius, color, -1)
285.                 elif self.calibration_stage == 6:
286.                     cv2.circle(frame, (self.accuracy_x, self.accuracy_y), radius, (139,
    0, 0), -1)
287.                     cv2.circle(frame, (self.accuracy_x, self.accuracy_y), radius + 7,
    (139, 0, 0), 3)
288.             return frame
289.
290.         def add_calibration_data(self, x, y):
291.             if (self.calibration_stage > 0):
292.                 if self.calibration_stage == 1:
293.                     self.calibration_stage1_data.append([x, y])
294.                 elif self.calibration_stage == 2:
295.                     self.calibration_stage2_data.append([x, y])
296.                 elif self.calibration_stage == 3:
297.                     self.calibration_stage3_data.append([x, y])
298.                 else:
299.                     # calibration = 4
300.                     self.calibration_stage4_data.append([x, y])
301.
302.         def start_calibration(self):
303.             self.message = "Welcome to the iTracker!"
304.             time.sleep(self.sleep_time * 1.5)
305.             self.message = "Starting calibration..."
306.             time.sleep(self.sleep_time)
307.             self.message = "Please look at the sequences of four red dots as they
    appear."
308.             time.sleep(self.sleep_time)
309.             for _ in range(4):
310.                 self.calibration_stage = self.calibration_stage + 1
311.                 time.sleep(self.sleep_time)
312.
313.             # end of calibration
314.             self.calibration_stage = 5
315.             self.message = ""
316.             self.process_calibration_data(True)
317.
318.             if self.measure_accuracy:
319.                 time.sleep(self.sleep_time)
320.                 self.message = "Starting accuracy test..."
321.                 time.sleep(self.sleep_time)
322.                 self.message = "Please follow the moving blue dot on screen."
323.                 self.calibration_stage = 6
324.                 movement_distance = 10 # px
325.                 time_end = time.time() + 3.5
326.                 while time.time() < time_end:
327.                     self.move_accuracy_point_y(self.accuracy_y - movement_distance)
328.                     self.move_accuracy_point_x(self.accuracy_x - movement_distance)
329.                     time.sleep(0.1)
330.                 time_end = time.time() + 6
331.                 while time.time() < time_end:
332.                     self.move_accuracy_point_x(self.accuracy_x + movement_distance)
```

```
333.                     time.sleep(0.1)
334.                 time_end = time.time() + 5
335.                 while time.time() < time_end:
336.                     self.move_accuracy_point_y(self.accuracy_y + movement_distance)
337.                     time.sleep(0.1)
338.                 time_end = time.time() + 7
339.                 while time.time() < time_end:
340.                     self.move_accuracy_point_x(self.accuracy_x - movement_distance)
341.                     time.sleep(0.1)
342.                 time_end = time.time() + 3
343.                 while time.time() < time_end:
344.                     self.move_accuracy_point_y(self.accuracy_y - movement_distance)
345.                     self.move_accuracy_point_x(self.accuracy_x + movement_distance)
346.                     time.sleep(0.1)
347.
348.             self.calibration_stage = 7 # done with everything
349.             self.process_accuracy_data()
350.
351.         def move_accuracy_point_x(self, new_value):
352.             offset = 20
353.             self.accuracy_x = new_value if new_value > offset and new_value <
    self.frame_width - offset else self.accuracy_x
354.
355.         def move_accuracy_point_y(self, new_value):
356.             offset = 20
357.             self.accuracy_y = new_value if new_value > offset and new_value <
    self.frame_height - offset else self.accuracy_y
358.
359.         def process_calibration_data(self, remove_outliers=True):
360.             top_left_x = 0
361.             top_left_y = 0
362.             top_right_x = 0
363.             top_right_y = 0
364.             bottom_left_x = 0
365.             bottom_left_y = 0
366.             bottom_right_x = 0
367.             bottom_right_y = 0
368.
369.             # removing outliers will account for times when the iTracking fails and
    thinks the pupil is on the eye bounding box border
370.             if not remove_outliers:
371.                 top_left_x = sum(i[0] for i in
    self.calibration_stage1_data[len(self.calibration_stage1_data) // 3: len(
372.                     self.calibration_stage1_data) - 1]) /
    (len(self.calibration_stage1_data) - (len(self.calibration_stage1_data) // 3) - 1)
373.                 top_left_y = sum(i[1] for i in
    self.calibration_stage1_data[len(self.calibration_stage1_data) // 3: len(
374.                     self.calibration_stage1_data) - 1]) /
    (len(self.calibration_stage1_data) - (len(self.calibration_stage1_data) // 3) - 1)
375.                 top_right_x = sum(i[0] for i in
    self.calibration_stage2_data[len(self.calibration_stage2_data) // 3: len(
376.                     self.calibration_stage2_data) - 1]) /
    (len(self.calibration_stage2_data) - (len(self.calibration_stage2_data) // 3) - 1)
377.                 top_right_y = sum(i[1] for i in
    self.calibration_stage2_data[len(self.calibration_stage2_data) // 3: len(
378.                     self.calibration_stage2_data) - 1]) /
    (len(self.calibration_stage2_data) - (len(self.calibration_stage2_data) // 3) - 1)
379.                 bottom_right_x = sum(i[0] for i in
    self.calibration_stage3_data[len(self.calibration_stage3_data) // 3: len(
380.                     self.calibration_stage3_data) - 1]) /
    (len(self.calibration_stage3_data) - (len(self.calibration_stage3_data) // 3) - 1)
```

```
381.                bottom_right_y = sum(i[1] for i in
    self.calibration_stage3_data[len(self.calibration_stage3_data) // 3: len(
382.                    self.calibration_stage3_data) - 1]) /
    (len(self.calibration_stage3_data) - (len(self.calibration_stage3_data) // 3) - 1)
383.                bottom_left_x = sum(i[0] for i in
    self.calibration_stage4_data[len(self.calibration_stage4_data) // 3: len(
384.                    self.calibration_stage4_data) - 1]) /
    (len(self.calibration_stage4_data) - (len(self.calibration_stage4_data) // 3) - 1)
385.                bottom_left_y = sum(i[1] for i in
    self.calibration_stage4_data[len(self.calibration_stage4_data) // 3: len(
386.                    self.calibration_stage4_data) - 1]) /
    (len(self.calibration_stage4_data) - (len(self.calibration_stage4_data) // 3) - 1)
387.            else:
388.                proportion_to_cut = 0.10
389.                top_left_x, top_left_y =
    stats.trim_mean(np.array(self.calibration_stage1_data[len(
390.                    self.calibration_stage1_data) // 4:
    len(self.calibration_stage1_data) - 1]), proportion_to_cut, axis=0)
391.                top_right_x, top_right_y =
    stats.trim_mean(np.array(self.calibration_stage2_data[len(
392.                    self.calibration_stage2_data) // 4:
    len(self.calibration_stage2_data) - 1]), proportion_to_cut, axis=0)
393.                bottom_right_x, bottom_right_y =
    stats.trim_mean(np.array(self.calibration_stage3_data[len(
394.                    self.calibration_stage3_data) // 4:
    len(self.calibration_stage3_data) - 1]), proportion_to_cut, axis=0)
395.                bottom_left_x, bottom_left_y =
    stats.trim_mean(np.array(self.calibration_stage4_data[len(
396.                    self.calibration_stage4_data) // 4:
    len(self.calibration_stage4_data) - 1]), proportion_to_cut, axis=0)
397.
398.            self.left = (bottom_left_x + top_left_x) / 2
399.            self.right = (bottom_right_x + top_right_x) / 2
400.            self.top = (top_left_y + top_right_y) / 2
401.            self.bottom = (bottom_left_y + bottom_right_y) / 2
402.            if self.right <= self.left or self.bottom <= self.top:
403.                raise Exception('Calibration process failed. Please try again')
404.            print('Left: {}'.format(self.left))
405.            print('Right: {}'.format(self.right))
406.            print('Top: {}'.format(self.top))
407.            print('Bottom: {}'.format(self.bottom))
408.
409.        def process_accuracy_data(self):
410.            threshold = self.frame_width // 10 # px
411.            numerator = 0
412.            denominator = 0
413.
414.            for quadrant in self.accuracy_quadrant:
415.                if quadrant[0] == quadrant[1]:
416.                    numerator += 1
417.                denominator += 1
418.            print('Quadrant Accuracy: {}'.format(numerator / denominator))
419.
420.            numerator = 0
421.            denominator = 0
422.            for distance in self.accuracy_distance:
423.                if distance <= threshold:
424.                    numerator += 1
425.                denominator += 1
426.            print('Accuracy within {} px: {}'.format(threshold, numerator /
    denominator))
```

```
427.
428.   if __name__ == "__main__":
429.       # print('\n'.join(sys.path))
430.       # looking for one argument
431.       if len(sys.argv) == 4:
432.           EyeTracker(int(sys.argv[1]), int(sys.argv[2]), sys.argv[3])
433.       else:
434.           exit("Missing Webcam index. Run 'python3 eye_detector.py 0 5 true'.")
435.
436.   # RUN via "python3 eye_detector.py 0 5 True"
```

Document 1: Eye Tracking Program

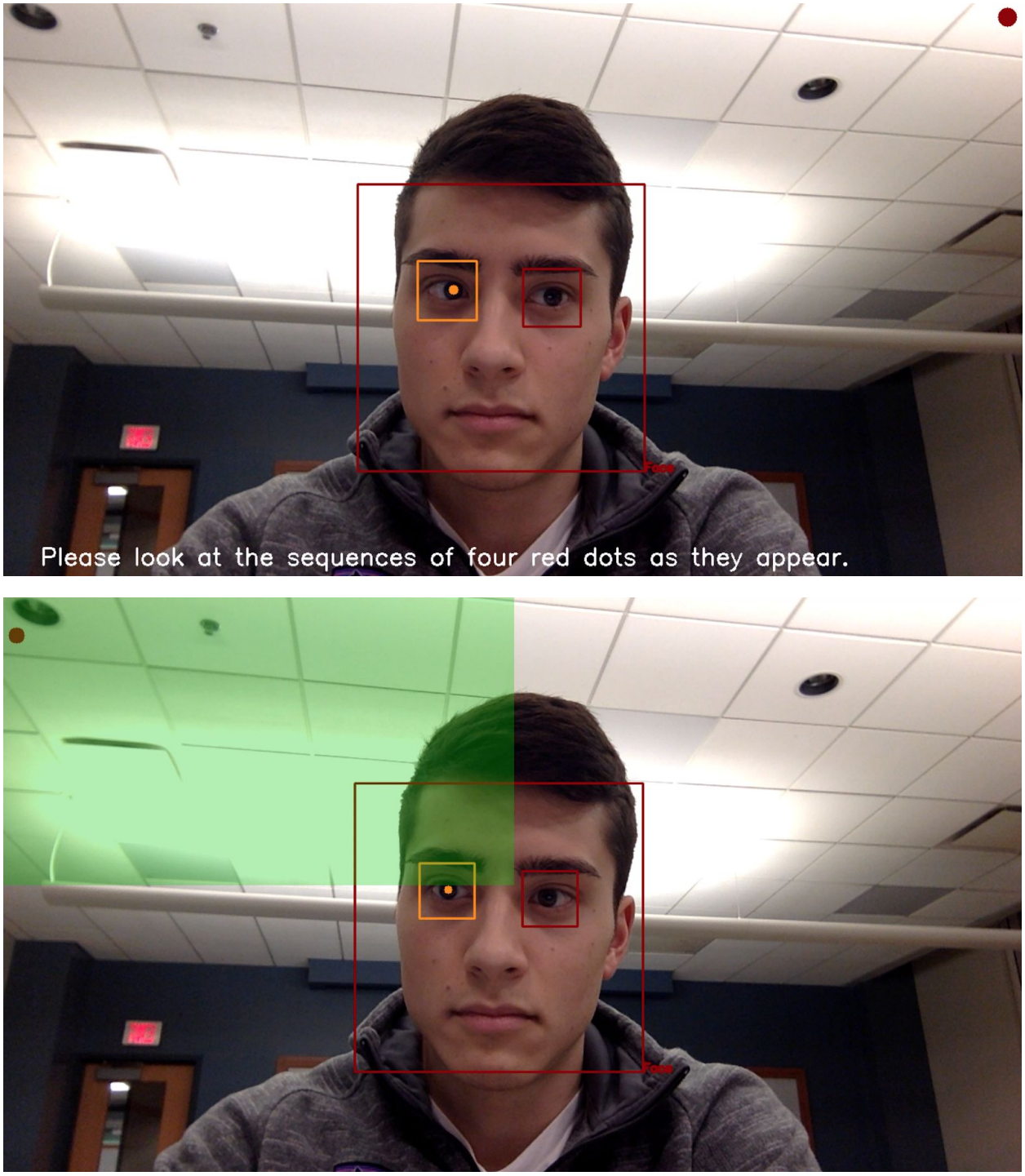


Figure 2: Screenshots from Running Program Showing Calibration and Gaze Detection

E. Works Cited

[1] https://imotions.com/blog/top-eye-tracking-hardware-companies/
[2] https://imotions.com/blog/eye-tracker-prices/