# Eye Tracking in a Virtual Reality Flight Simulator

*Team Members: James Burgess, York Delloyd, Richard Deng, Travis Whitten*

*Faculty Advisor: Dr. Kyle Johnsen*

*Client: Gulfstream Aerospace Corporation*

*April 23, 2018*

# Table of Contents

# Problem Statement/Executive Summary

Historically, engineers have been unable to see and experience what their end user does directly and have had to utilize feedback forms and other user-reported data in which the user, purposefully or not, determines what information is relevant. In the data driven world that we live in today, this should not be the only way we collect user data. Our team, in conjunction with Gulfstream, is working on creating a solution to perform more holistic data collection to see what their pilots are focused on while operating or preparing an airplane. In addition, this technology could be used for other applications such as the collection of customer data. More specifically, the technology could be used to determine which parts of the plane customers are focused on while their sales team does a walk through. It could even allow Gulfstream to collect data on areas needing further emphasis during sales or to upgrade particular aspects or features of their planes to be more desirable over their competition.

The objective has been to develop a solution within a head mounted display (also referred to as HMD) to track and record where users are looking not only at any given moment but also how often and how long they are focused on a point for a period of time. We also developed a way to extend the technology's use for potential training/testing purposes such as a pilot's reaction speed while in flight. While we can easily track and see the general area where users are looking, we wanted to go further and get more accurate locations on where users are looking. We accomplished this with the implementation of eye tracking inside of a head mounted display. This allows for users to be in a realistic simulated environment where everything they subconsciously look at is recorded. The users do not need to remember specific details about their experience, which potentially leads to less bias in what the user remembers.

Over the past two semesters, our group has successfully developed a rudimentary flight simulator with Pupil Labs eye tracking cameras. Using the simulator we developed, we ran a study to collect data for our client. The simulator tracks a participant's gaze and records where and when the participant is looking throughout a session. Additionally, a session's data can be rendered into a heat map to better visualize where participants were focusing for the majority of the time. Throughout the project's evolution, Gulfstream granted us with total control and creativity. Gulfstream was very hands off with the project and allowed is to tackle the problem in any way we saw fit.

## Customers/Stakeholders and Their Requirements

Our client for this project is the Gulfstream Aerospace Corporation. As aforementioned in their name, Gulfstream develops, designs, and manufactures aircraft. In terms of our project, Gulfstream expressed interest in eye-tracking as a growing technology. Given this interest and their busy schedule, they had asked us to explore current eye-tracking technologies in conjunction with virtual reality. The requirements that they had laid out for us are as such; judge the usage of eye-tracking in a virtual reality and return feedback on our conclusions.

Feedback from Gulfstream was lacking due to their schedule, yet the requirements that they laid out from the beginning where followed through with this project. Most of the decisions on what our requirements were for the second half of the project timeline were expressed by our faculty advisor Professor Johnsen and ourselves. Stemming from the original requirements, the requirements that we created follows the need to verify the usage of eye-tracking in a virtual space and develop that into a study in which data obtained from the eye-tracking modules will be recorded and analyzed. The basis for our study came about at the end of the first semester after seeing the potential of the eye-tracking modules with the first implementation of the trackers in the form of a reaction-based game.

To meet the initial requirements, our team has developed a flying simulator in Unity. Using the Pupil Labs head-mounted display eye-tracking modules and their API, we can track where in the cockpit participants look as well as identify what they are looking at and for how long.

## Background/Introduction

The need for the project stems from Gulfstream's desire to explore uses for the relatively new eye tracking technology. Typically, eye tracking is thought of for use in foveated rendering, but Gulfstream wants to explore the technology in a more commercial sense, with a focus on utility rather than performance. Since eye tracking in head mounted displays are relatively new; there is a general lack of data and use information published on the subject. A few of the studies we have looked at include a psychology study on eye tracking and its impact for HCI design, a very old Clemson study on developing one of the first eye trackers for HMDs and its challenges, and a 2002 study that validates that the use of HMDs and eye trackers can accurately depict the real world (Duchowski).

These studies and research show that eye tracking has possible and credible uses that other than foveated rendering. We know that the data from this can and should be used in design of interfaces. We also know that virtual environments mimic the real world enough to show and give usable data. Of all the studies we've read, none of them perform a study on gaze reaction times, which is what Gulfstream wants to see. Our technical requirements for this project consists of implementing some Pupil Labs programs with a previous project all in VR. The software must be in VR and track eye data. Since this is just creating a module we can run studies on, we do not need domain knowledge as we are not analyzing data.

## Engineering Specifications

The engineering specifications are as follows:

1. Virtual Reality is a requirement
2. Usage of an Eye-tracker and Head Mounted Display is required
3. The experience must utilize an eye tracker
4. A study is required to analyze the usage of eye-trackers
5. The simulation requires a set format to create a baseline for the study
    a. Requires a start and finish condition
    b. Requires a component to test eye-tracking

c. Requires a calibration set for each person

## Design Concepts

Since our project layout is straight forward, we did not brainstorm and come up with different designs for the project. The challenge we had to undergo was implementing the solution of eye tracking within a flight simulator made for Virtual Reality use in a head mounted display. For our design we broke our solution into three more manageable parts: A virtual reality flight simulator in Unity, an eye tracker implementation in Virtual Reality along with recording data such as reaction time, and combining the both of them.

We chose to approach the problem with a combination of three designs because of time constraints and the group format. Because we have about a semester and a half to work on this, we have to find a way to create individual parts so the four man group can break into smaller teams and work in parallel with different parts of the project. In the end, we can combine the two “finished” products (hopefully) easily for a finished result.

This culminated into the creation of an independent project solely based on the eye-trackers and our final design implementation with the integration of the eye-trackers in our flying simulator. The approach here allowed us to flesh out the capabilities of the eye-trackers as well as determine a focus for our study to give a comprehensive conclusion that Gulfstream would like. The process of the first project developed very differently than what we had imagined due to the various changes in the eye-tracking API from Pupil labs. There's a vast difference in the performance and usage since we've obtained the eye-trackers, which has come as both a good and bad experiences.

## New Knowledge Development

Coming into the project, two of the group members had never programmed in Unity VR. None of us have used eye trackers before, and none of us knew the implications of eye tracking. We had to do research on different eye tracking solutions which included looking into a Chinese manufacturer’s product called AGlass, Pupil Labs, and Tobii eye trackers

(Tobii). With this pool of manufacturers, we had to choose the Pupil Labs based on price and predicted functionality. After obtaining the eye trackers, we had to figure out what they could do and what are the implications of the data. We worked with Gulfstream's Advance Technology Team to determine a use case and develop project goals. Afterwards, it was all working in Unity to implement the eye tracker which was either all learned knowledge, Google, Dr. Johnsen, or Pupil Labs Documentation/Discord. We plan to study more about data collection and analysis once we hit that phase of the project.

In terms of working with the eye-trackers, there was a slew of difficulties that we were faced with due to the development stages the eye-trackers that we had were in. While they had working versions of their API, Pupil Labs constantly made updates and changes to their API to fix bugs and sometimes introduce them unintentionally. Some of our team members found this frustrating at times as it would impede progress, but this also allowed us to discuss with the Pupil Lab developers directly to troubleshoot any issues as the Pupil Labs API is both open source and collaborative.

Going into the second semester of our project, most of our concerns focused on bettering the original flight simulator. Due to some feedback from Gulfstream, the original flight simulator was not allowed to be used and another one needed to be created. This version of the flight simulator had an implementation of the eye-trackers and was heavily based on the previous project with the use of different assets. For this part of the project, our team needed to heavily focus on creating a study. Aspects of this subject includes data collection and creating an objective for the study. The ideas that we had down were to create a heatmap out of the eye-tracking data and to focus our study on where study participants look based on their familiarity of virtual reality and flight experience. We decided on this idea in the context of identifying pilot habits' and the like. Seeing where most people looked in general compared to where experienced pilots looked can assist in instructing up and coming pilots.

## Design Constraints

To design for excellence, we looked at a few specific subsets and we tried to make sure we hit them. Since we are doing a software project that's end goal involves data collection, we want to design it for reliability, speed, reuse, and test.

We chose to focus on design for reliability, because we will be collecting data to run analytics on and it should not break. To make the design as reliable as possible, we've chosen the best Virtual Reality Headset and Eye Tracker available to us. This will keep us from getting false reads. We can also flush this out by writing bug free code to make sure it runs well every time giving accurate data. For design for speed, we want to make sure the program runs as smooth as possible with the highest frame rates. This is done by optimizing our assets and being smart with not flooding the program with useless data and functions. To design for reuse, we made sure that programs were easy to start up and reset - literally by a press of a button. Lastly, since we are running tests with this program, we must Design for Test. We did so having a making our program mainly do one thing so it eliminates outside variables.

In terms of the actual design of our virtual reality set-up, there was nothing to really build off. The previous assets that we had were reused from another project that was a flight simulator. Seeing that half of the team is also working on this project, we were able to reuse most of the digital assets and the remote yoke as well as the HTC Vive set-up that maintained a set location for the yoke and the pilot's seat.

## Creativity and Innovation

Development for virtual reality in head-mounted displays is still in its infancy. Head-mounted displays are just now widely available to the masses, and development for virtual reality will only continue to become more popular. Simply developing games and educational content in Unity for head-mounted displays is innovative, and our project takes this a step further with the implementation of eye-trackers.

Eye-tracking is not new, but eye-tracking accurately within a head-mounted display is new. We have little content to pull inspiration from and few products to develop with. Our team feels that we are exploring a new frontier of intractability within virtual reality. From our time with eye-trackers, our team feels that it is inevitable that they will be used extensively in the coming years. We do not doubt that head-mounted displays will ship with eye-tracking capabilities in the box in the near future. Eye-tracking doesn't necessarily change the amount of potential that developers have, but it does make the entire experience for the user more organic. With the advancement of computing technology, virtual reality comes closer and closer to its tipping point of affordable performance -- a more organic experience. Eye-tracking aims to take this experience a step further.

## Final Design

Over the past two semesters, our group has successfully developed a rudimentary flight simulator with a model of a Cessna Mustang within Unity3D. The plane can be controlled with flight simulation controls consisting of a yoke, throttle, and pedals. Our virtual reality application uses an HTC Vive head mounted display, as well as Pupil Labs eye tracking cameras. A user's gaze is tracked while in session and data, such as where a user is looking and at what time, is recorded and exported along with a video of the session. Afterward, the data recorded from a session can be read and processed to render a heatmap within the virtual environment. This heatmap consists of cubes located at the user's gaze positions from the session, and they are colored from green to red based on how concentrated the data is at any given point (*appendix B)*.

The set of the physical area for the virtual reality space is more involved than a typical virtual reality setup, as we have some additional hardware that most people do not make use of regularly. The head-set that we used for this project is the HTC Vive with a Saitek flying yoke as a physical controller. The space set up is approximately 10 ft by 10 ft for the Vive space. The Pupil Labs HTC Vive eye-tracking modules are locked to the lens of the head-mounted display. The physical placement of the controller is set inside of the Vive space. To maintain immersion, the controller and user are required to start in a certain position inside of the space.

The plane model is based off a Cessna Mustang, and its cockpit was stripped of unnecessary buttons and other controls, as non-pilots do not need them. Altimeter, compass, and airspeed is displayed in front of the pilot. A co-pilot seat is situated in the cockpit as well. Each part of the plane was labeled appropriately as to be easily identifiable when analyzing what participants are looking at (refer to *appendix B)*. The models rendered in the cockpit have mesh colliders so that the invisible ray cast from the eye tracker can collide and identify where and at what the participant is looking at.

The Pupil Labs eye tracker has open-source software that has contributions from many programmers using python for pupil detection. This library is heavily utilized for our project, as the image recognition software used to track the pupil (*appendix C)* is necessary for our design. Using the Pupil Labs eye tracker, data is collected on where the participant is looking at any given time, recording information at 60Hz. This is done by grabbing information from the Pupil Capture software and translating that to the viewport point on the Headset. From this information we can create a segmented ray cast that extends infinitely and independently of where the head is facing. When this generated ray cast intersects with an object in the simulated world, it will create data, and since the user is almost certainly always looking at something in the simulation space we will generate sixty data points a second.

The gaze location data recorded from the flight session are then exported to a CSV file. Which are then reviewed upon user request at the end of the flight, and calculations are made to determine the colors of each individual cube to create the heatmap of where the user was gazing during the flight. Our equation simply counts how many other data points are within .035 x .035 units of each individual point. This count is then normalized using a function called *percent rank.* Each data point is compared with the other data using an equation that compares how many data points have a lower count, divided by the total number of data points minus one.

Within Unity, we can render this heatmap by spawning cube prefabs at each data point location from the CSV file. After the instantiation, each object can have its material color set, which is based off the percent rank explained previously. The most remote cubes are pure green – rgb(0, 255, 0). The cubes at the highest concentration are red – rgb(255, 0, 0). Cubes in between are yellow – rgb(255, 255, 0). Since cubes have a value assigned from zero to one hundred, we decided to step up/down the values in increments of five. If the percent rank was less than 50%, the cube would be colored rgb(*percent rank* * 5, 255, 0). If the percent rank is over 50%, the cube would be colored rgb(255, 255 – (*percent rank* * 5), 0).

For the flying simulation, we used Unity's physics engine to handle most of the calculations and such, only editing the ridged body of the object that we needed to manipulate. For our project, most of the physics implementation is strongly based on a past project, as well as Unity's demo asset that allowed for an arcade-like flying simulation with gamepad controls. With all the code being in C# for the use of Unity scripts, most of the methods are written as method calls with the use of a setup and loop method to constantly update any of the variables used to calculate any inputs taken in from the user. This would include any of the inputs coming in from the yoke, which are also made as custom set inputs for the yoke. This usually translates into any adjustments that would cause changes in the plane's thrust, drag, pitch, yaw, and the like.

## Conclusions and Recommendations

The final product – the culmination of the past two semesters – accomplishes a great deal of what was expected. However, with great accomplishment, comes small failures. Throughout the eight-month development of this class and project, design requirements were met and even surpassed in some cases, but as the design changed over time, some initial design goals had to be reworked or sacrificed to meet more important and pertinent requirements. Examples of secondary goals that had to be sacrificed or reworked are: realistic flight simulation, real pilot data, more accurate eye-tracking hardware/software, and a larger test population. In addition, our reaction timing application did not make it into our study, and therefore, we do not have data to present on reaction time of participants during flight. Although it was not finished prior to the study, it is now complete and functional. Despite these features and goals being set aside, and most importantly, everything put forth in the problem statement has been implemented in a way that we find satisfactory.

Results of the study that was conducted show some interesting conclusions. Throughout the study, participants were asked to answer some simple questions, of which included two self-assessments of how familiar with VR the participant was and how familiar with piloting of an aircraft the participant was. We found a trend in the data suggesting that participants with more VR experience ended up visually exploring the environment more thoroughly. Participants with little VR experience (familiarity of one or two out of five) only diverted their gaze from the main left center window approximately 2% of the time. Users that had some experience with VR (familiarity of three out of five) diverted their gaze from the same window for 5% of the time. Interestingly, more experienced VR users (familiarity of four or five out of five) looked out other windows roughly 20% of the time during their session. It is clear that being familiar with VR makes an individual more likely to visually explore their environment while other tasks are at hand. *(Appendix A)*

While we are happy with our final product, there is still much that could be done if this project were to continue. For starters, if this simulator were ever needed for use with real pilots, it would need an overhaul in order to properly immerse the participant. The physics would need to be reworked as well. Another feature that could be implemented is multiplayer with a co-pilot. This co-pilot could be an instructor or simply a spectator. They could be given control over a visual line or “ray cast” to direct the pilot’s attention where they want. Lastly, while the heatmap and data collection is functional, we would have liked to have more data for collection and analyzation. This would allow us to draw better conclusions to the usefulness of the software.

## Contributions of Each Group Member

**Travis Whitten –** I researched the various products on the market for tracking pupils in a head-mounted display, and I assisted in the acquisition of the product of choice. I've helped integrate the eye-tracker into the environments and quality assurance pertaining to it. I contributed to the creation of ray casts from the eye-trackers as well. Most importantly, I designed the implementation of the heat map, and with James's assistance, polished the experience and toggle to display the heat map. I learned almost everything I currently know about Unity and vide game programming from this project, and it has brought me a long way. I now know the process for making a game or other 3D application. I also value the experience I gained from working on a major long-term project with multiple colleagues, as a long-term collaboration has been new for me.

**James Burgess –** Documentation and Testing Design. Created a test box area to test the eye tracking hardware before integrating it into the flight simulation. I have developed some of the unity testing code to create a bubble map of where the user looks during a simulation, and I have created a reaction timer test area in the same virtual reality space. I have also successfully implemented the Pupil Labs Eye Tracking software with the VR Flight Simulation. I worked with Travis and Richard to test and make sure that the eye tracking portion of the flight simulation could collect accurate and relatable data. These contributions have proven useful towards the end of our project, as we have generated data from each test subject individually, and created a better understanding for how people react to being in a cockpit for the first time.

**York Delloyd –** As team leader, my main responsibilities are to set goals and get the group to move towards them. In the beginning of the project, I was the primary liaison between our team, Gulfstream, and our mentor – Dr. Johnsen. I set objectives and a timeline for deliverables so that we can meet our client's needs and also fulfill requirements of the capstone class. Additionally, I was a secondary programmer for the project, providing support to my teammates as we participated in peer programming. One of my most major roles was to generate and run a study for our simulator to collect validation data for Gulfstream as to determine the usefulness of our software and technology. I learned about the process of leading a group to a arbitrarily set objective. As other parties are only awaiting results and are not worried about the process, we had to set our own goals and milestones that would lead to a good deliverable. Regular classes prepare you for work and group work by setting milestones to submit, but this opportunity allowed me to learn to create my own milestones for our project.

**Richard Deng –** Programmer. Maintained the previous version of the project and currently handling the integration of the eye tracking module into the current version of the project. I've looked into the module made by James and made some adjustments to better track the eyes and create a reliable ray cast from the camera to the point presented on the screen. I worked with James and Travis on implementing the eye tracking up until we decided to work on in separate veins to accomplish more work over a shorter period. Most of the coding that I was involved with was creating a flying simulator based on another project, with a few restrictions. I collaborated mostly with my previous team member who I worked with to help me rebuild the flying simulator that we have now. I also worked on assisting James implement the eye tracking modules into the current simulator, as there were some quirks required to mesh the two together.

# References


*aglass.com*. [Online]. Available: http://www.aglass.com/product/. [Accessed: 2017].

"Binocular eye tracking in virtual reality for inspection training," *ACM Digital Library*. [Online]. Available: https://dl.acm.org/citation.cfm?doid=355017.355031. [Accessed: 2017].

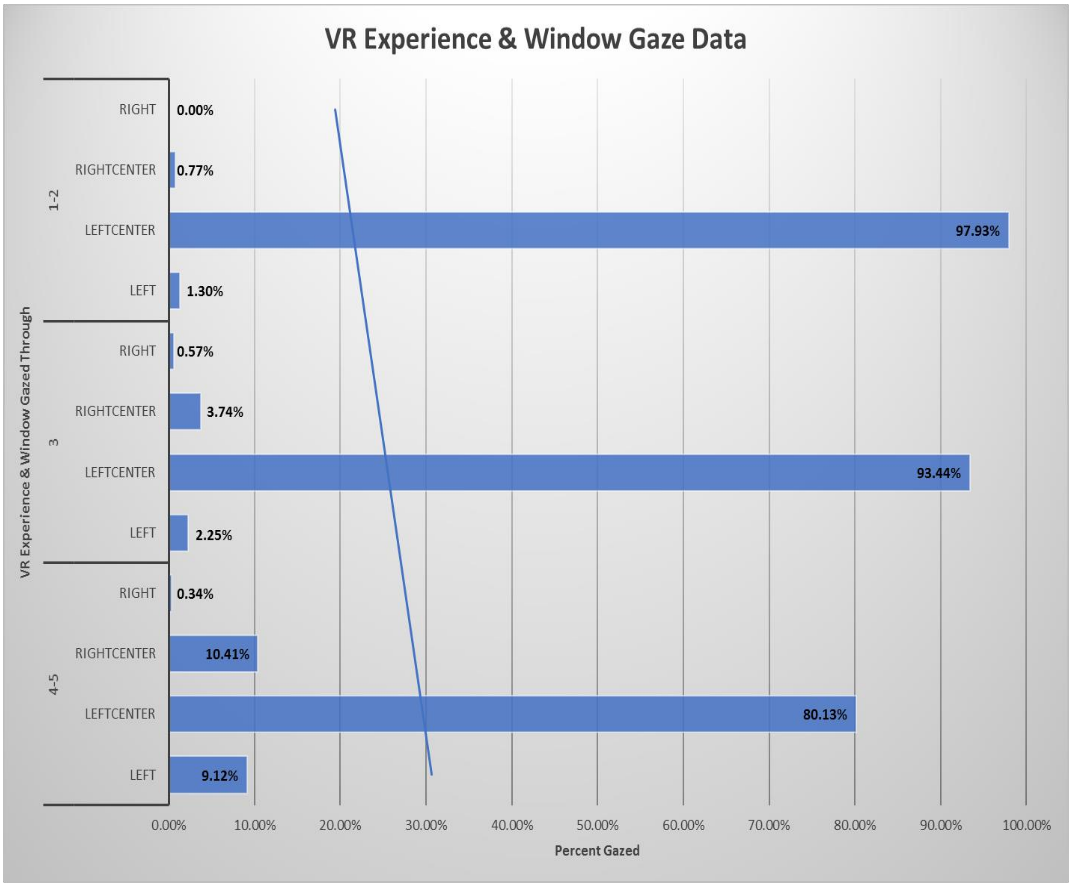
"Eye Tracker," *Pupil Labs - VR AR*. [Online]. Available: http://www.pupil-labs.com/vr-ar/. [Accessed: 2017].

"Tobii Pro VR Integration based on HTC Vive HMD," *based on HTC Vive HMD*, 15-Oct-2015. [Online]. Available: http://www.tobiipro.com/product-listing/vr-integration/. [Accessed: 2017].

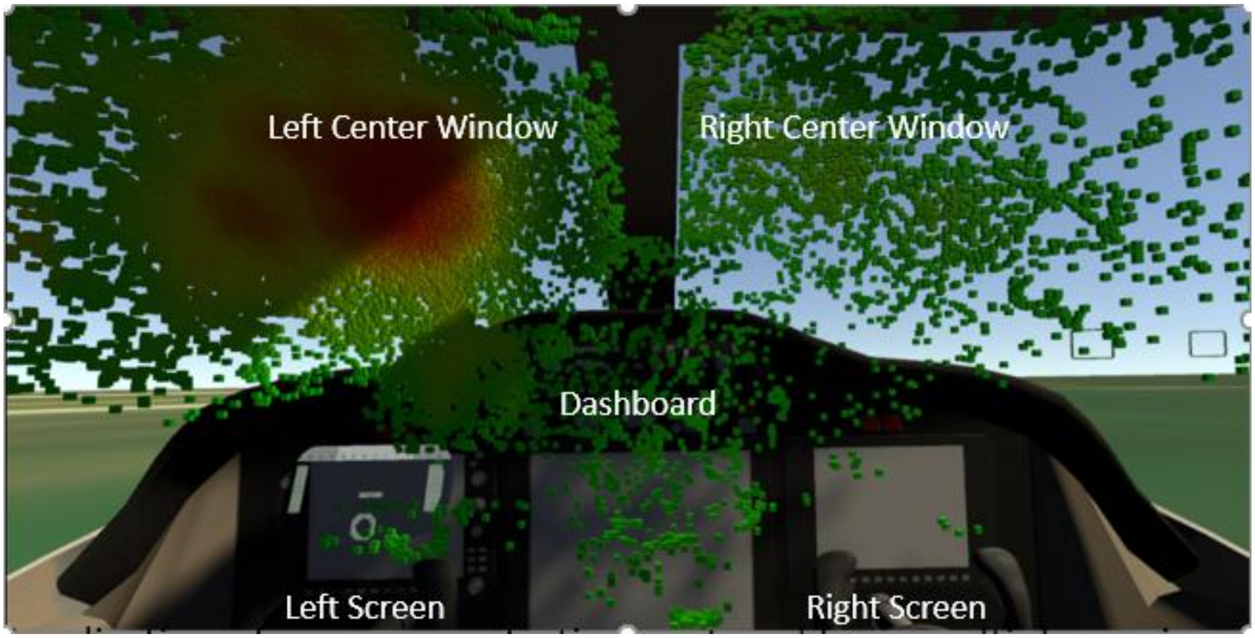
"Using virtual reality technology for aircraft visual inspection training: presence and comparison studies," *Applied Ergonomics*, 26-Nov-2002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000368700200039X. [Accessed: 2017].

# Appendices

## *Appendix A*



*Appendix B*




*Appendix C*