CS CAPSTONE PROJECT DOCUMENT

MAY 30, 2020

AUGMENTED REALITY COLLABORATION

PREPARED FOR

OREGON STATE UNIVERSITY INSTRUCTOR

KEVIN D MCGRATH _____ _____

Signature *Date*

PREPARED BY

GROUP CS 13

ARC

CARSON PEMBLE _____ _____

Signature *Date*

RYAN MIURA _____ _____

Signature *Date*

HAOZHE LI _____ _____

Signature *Date*

Abstract

Our client faces the problem of collaboratively annotating PCB boards from remote locations. There is currently no technology to facilitate this at a reasonable price, so we propose using a Zed Mini camera attached to an HTC Vive headset and our personally developed software to allow this. Our software will allow users to collaborate on board designs and view board physicality independent of one another while sharing visuals, audio, and user-added virtual markups. We will measure our progress through iterative prototypes, allowing our client to test our work. This will result in saving the Oregon State University research department a lot of money, advance affordable AR (Augmented Reality) technology, and help our client further his research.

CONTENTS

1 Introduction to Project

1.1 Who requested the project?

Kirsten Winters and Scott Fairbanks requested that we choose a project for the Senior Software Engineering Project course. Each member of our team applied to projects and the ARC team was created.

1.2 What is its importance of the project?

First, the importance of senior projects is to test the overall knowledge of Computer Science students and to put to test everything we have learned up until this point. Senior projects are also a great way for the students to learn software engineering skills, such as talking with clients and prioritizing our time on our own. Secondly, the importance of this specific project is so our client would be able remotely collaborate with colleagues in a new and improved way.

1.3 Who was our client?

Our client for the AR Collaboration Suite project was Kevin D McGrath. Mr.McGrath is an computer computer science professor at Oregon State University.

1.4 Who are the members of the ARC team and what were the roles?

The first member of our team was Carson Pemble. Carson was the designated team lead who would communicate with the teachers assistant, the course instructors and the client. He was also assigned the roles of working on the Login/Registration, VR Keyboard integration, OpenCV board tracking, and AR laser pointers and functionality. The second member of the ARC team was Ryan Miura who worked on the Networking and the Audio components; although these were moved to a stretch goal, we have good starter code for the next team. The third and final member of our team was Haozhe Li and he worked on the import of VR Keys and the laser/board cross functionality.

1.5 What was the role of the client?

As our client, Mr.McGrath gave us a very open project with vert few hard requirements. We discussed more with him and created our own project requirements. After that point Mr.McGrath's role in the project was to supervise and answer any questions our team had about specific things he might want implemented.

1.6 How Covid-19 affected our project?

In the spring term of 2020, our project was greatly impacted by the coronavirus pandemic. With the shutdown of the Oregon State campus, our team was no longer allowed to meet in person to discuss project plans or to test the development of our project together. We had to split up the team, with one of the members moving back to Texas. This made development and testing two very separate tasks. One member would write code and then we would have to call the other member and screen share in order to explain the code and test the functionality. Due to the many setbacks of these unexpected variables thrown into the project we had to adjust our deliverable to set the network requirement to a stretch goal.

1.7 How to pick up where we left off?

To the next team (ARC 2.0) we recommend that you read this final document, the code comments, and the GitHub Commit logs to help you jump start this project. You should be able to read up on some of the technical resources found later in this document and in the GitHub README.txt on our project GitHub page. For more detailed instructions please refer to Appendix 3.

2 Requirements Document

2.1 Introduction

2.1.1 Purpose

The purpose of this document is to create an agreement between the project team, ARC, and our client, Kevin McGrath, for what will be completed during the time frame of this year. This will cover the basics of what our client will be receiving once our product has been completed. Throughout this paper there will be a layout of what steps will be completed and with the inclusion of a Gantt chart, when those steps will be completed.

2.1.2 Product Scope

The scope of this product is to construct and develop a product that can will be usable and helpful for our client's research and ease of collaboration with colleagues. The time frame for our team's development on this project is one year, ten weeks of research and documentation, fifteen weeks of development, and five weeks of finalization and virtual presentation.

2.2 Overall Description

2.2.1 Product Goal

The overall goal of our product is to have 2 or more AR headsets that are able to share visuals, audio, and virtual annotations with low latency. We would like to have multiple headsets, preferably three, connected through a LAN network with AR working at a single location. Everyone wearing a headset should be able to see what the AR headset person is viewing and everyone should be able to create virtual markups/annotations. The users should be able to switch cameras to create a "shared screen" experience for the other two users. This should all be done in real-time, meaning a live feed from one headset to the other, and this will also need to be done securely and safely. Once the product is complete, Mr. McGrath and his colleagues will have a foundation for a trustworthy and simple collaboration suite where they can simply put on a headset and share their work across the network. This will be the final result of the project. Our goal for this year is to complete all of the previous steps except for the network portion as we had to deal with unpredictable drawbacks.

2.2.2 Operating Environment

As our client has requested the software will be developed on Windows systems and will tested using the computers in CGEL, Batchellor Hall at Oregon State University and our own personal systems. The code will developed in Unity 3D and it will use SDKs from both of the hardware companies, HTC and Stereolabs. HTC Vive and Zed Mini have a code base that will also be used in our development process, with the help of Unity Plugins. We will also use other SDKs as we develop the product.

2.2.3 User Documentation

Documentation is required as a key element to this class and it will be created throughout this term and updated throughout the development process. Developed code will contain comments to explain our work and the logic of our programs. Presentations will be given at the OSU Engineering Expo and to our client directly.

2.3 Hardware

2.3.1 ZED Mini

As with most computer science projects, there is still a large need for hardware. Through the development of this project, we are required to create a few key pieces of hardware. We have been requested by our client to create an AR headset from two pre-existing products: HTC Vive and ZED Mini.

One of those is a 3D camera from Stereolabs called the ZED Mini. This camera allows for stereo pass-through which will be used to help us create an AR experience.

2.3.2 HTC Vive

Another piece of hardware that our client will be providing for us to use is the HTC Vive. This is a VR headset that the ZED Mini will be attached to the front of, to simulate an experience as if the user was looking through the headset. This newly created AR headset will be connected to a PC to power the graphics and two hand-held controllers which will allow the users to rotate and mark up the Virtual PCB board which are the key required features of this project. All hardware construction/combining will be done very early on in our development process and will be completed as a team.

2.4 Augmented Reality

2.4.1 Visuals

The required project before us is to create a product that will overlay virtual items onto what the ZED Mini streams as reality. This means there is not a full virtual reality created but instead an augmentation of reality. This will be done through collecting visuals and audio and then overlaying virtual items on top of those.

The Unity3D software we use for development is able to work with the Zed Mini camera, and it supports AR markups on the camera feed. Development can start with getting video streaming to work from the camera to the headset screen. Connecting the Zed Mini to a single headset and displaying the camera feed to the user provides an indication of success. From here, virtual markups should come into the visuals, allowing for writing virtual text on the display. This text does not have to be scaled or positioned correctly at the first development, but will have to work by the final prototype for the expo. Being able to draw to the screen and view it is the next required step to our project. Then, scaling and positioning will become a focal point. Using ArUco images, the markups will be scaled to accurately indicate the components they are referencing. These marker images would be used to indicate which side of the board is currently being looked at and to reference the size of the object. Since the markers will have a known size, they can be used as constants. Once these have all been implemented into our project, we will have reached all the visual requirements. There are more requirements such as audio and more defined virtual markings.

2.4.2 Audio

Another key element to our project is that our software must support audio communication, as we are trying to build a collaboration product, audio communication is very important for proper communication. The HTC Vive has headphones built-in, which could be used if the user does not want to use personal headphones. There is software within the headsets to allow for audio to be passed from the PC to the headsets. This software may work for this purpose, but it may also require manual construction of an audio connection. It is also important the audio is not lagging behind the video or collaboration could be a struggle. Because of this, we are aiming to have an audio latency of less than one second for this project. Our hopes are that we can cut this down even farther. Regardless, creating an audio connection is key to this project's success.

2.4.3 Virtual Markings

One of the key features that our client is requesting is that all users should have the ability to circle or annotate specific components in the PCB in the virtual space. The first step in implementing this feature is to first start by being able to add some sort of visuals to the screen at the user's command. The user will be able to use their controller to point somewhere and create a virtual marking, like a custom note on the PCB. We will then enable the user to make more elaborate markups such as text placed anywhere in the virtual space. This could be done with the addition of a virtual keyboard, or possibly voice dictation if we have time to implement it. As there are more and more annotations added to the visuals, we will have to implement some sort of 3D visual management techniques to create clean layouts. We could follow something similar to the hedgehog labeling [4] technique to keep the visuals clean and organized.

2.5 Data Transmission

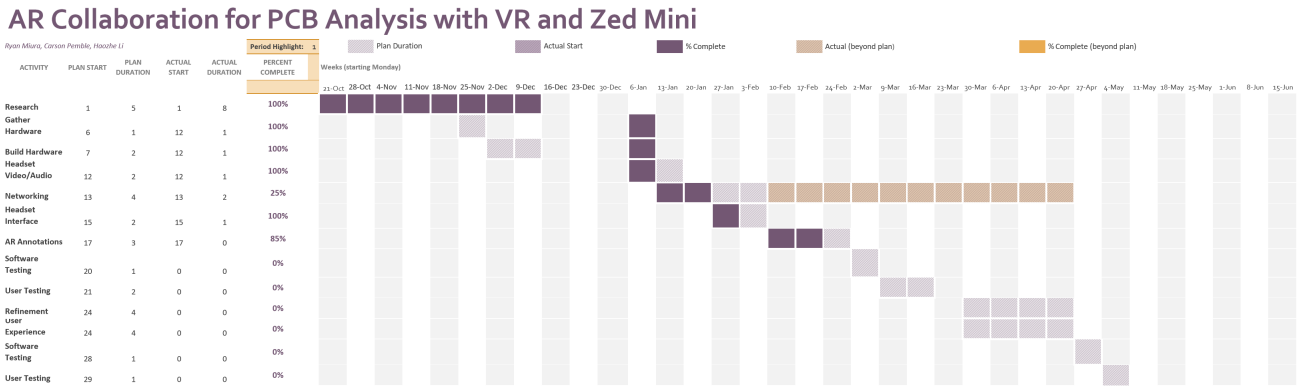
2.5.1 Networking

This project requires some data transmission to be able to collaborate with multiple headsets. We will make attempts to achieve some sort of networking, however these will all be stretch goals for the project. That being said, the first step towards this goal is testing networking through a single PC. Getting our headsets to view the same thing is a good starting point that gives a clear sign of functionality. From there, getting audio to send is the next step. This sets up basic communications between users. These headsets should view the same thing, with a single user broadcasting and the other viewing. A stretch goal should be moving the headsets to completely different networks, with a cloud-hosted server. Refining the code and getting a low latency is another stretch goal. The last stretch goal is getting the project working in Netcode, the new Unity networking standard. This would meet and exceed our clients requirements for network functionality.

2.5.2 Security

Since we are building a PCB collaboration program, there could be some trade secrets transferred across the network during the collaboration process. This illuminates the need for network security. It's necessary to make sure that the data won't be stolen or lost. Our client, Kevin McGrath, is an expert in network safety and he would like our project to have a secure network when transferring data. A key security feature that will be implemented into our program is a user login system that only allows registered users to join the collaboration suite. This registration will contain hashing and salting of the users passwords to prevent the files from being read. One of the steps that we will try to implement as a stretch goal is encrypting the transferred data and decrypting the data at the other user's end. We would be able to secure the data from the ZED Mini by encrypting the hex data with a hash table. We would like to implement some sort of AES (Advanced Encryption System) that would allow us to confidently send data from user to user without having to worry about it been viewed by outsiders.

2.6 Gantt Chart



2.7 Bibliography

[1] “Hardware.”HTC VIVE, https://www.vive.com/us/product/vive-virtual-reality-system/.

[2] “Bring Your Imagination to Life.” Stereolabs, https://www.stereolabs.com/zed-mini/.

[3]“Comparison of Audio Network Protocols.” Wikipedia, Wikimedia Foundation, 3 Oct. 2019, en.wikipedia.org/wiki/Comparison_of_audio_network_protocols.

[4] Tatzgern, M., Kalkofen, D., Grasset, R., & Schmalstieg, D. (2014). Hedgehog labeling: View management techniques for external labels in 3D space. 2014 IEEE Virtual Reality (VR), 27-32.

3 Design Document

3.1 Introduction

The purpose of this document is to provide a detailed guide on how to implement the project. The document will identify specific components of the project, explaining the steps taken to implement each.

This document is limited to the nine key components discussed in the tech review document. These were determined to be the most vital elements of the project. All nine components will be discussed in detail and explain how the implementation of each are vital to our project.

3.2 Hardware Design

The first steps in developing our project includes the creation of our specific Augmented Reality (AR) headset. This includes the combination of a 3D Stereo camera and a Virtual Reality (VR) Head Mounted Display (HMD).

3.2.1 AR 3D Camera

In this project, we will be using a Zed Mini camera that can capture 3D stereo images and video. Our client has worked with this camera before and thus felt it was a good choice for this project. The Zed Mini will be able to stream video to other users at a base level, but includes many other useful features that will be helpful. These will be accessed through the Zed SDK.

To capture video from the Zed Mini, a Camera object is created using the Zed SDK. The Camera object is configured with an InitParameters object, allowing for a large number of configurable parameters [14]. The camera is then opened with the open function [15]. To stream video, a StreamingParameters object must be created, allowing for configuration of properties such as the port and bitrate. Then, the enableStreaming function is called to enable the Zed streaming module. With streaming enabled, the grab function used to capture video will also send the frame over the network.

To receive video from another Zed Mini stream, an InitParameters object must be initialized with the IP and port of the sender. The camera is then opened with the open function and the InitParameters, allowing the application to function as if the sender's Zed Mini were attached to it. The grab function can be used to grab the current frame and the application can send it to the HTC Vive for display.

The Zed Mini also includes depth sensing, which gives a relative Z value to each object captured by the camera. This information can be passed from the API to our program. The depth mode and units can be configured in the InitParameters object. When the current frame is grabbed, the retrieveImage function is called to grab the image seen in the left camera eye. Then, the retrieveMeasure function is called to grab the depth map generated by the camera. The retrieveMeasure function can also be used to get a 3D point cloud or surface normals. The surface normals can be used by the program to determine the orientation of the PCB board at any given time. Other features included with the Zed SDK are positional tracking and spatial mapping, but these won't be used in this project.

Testing of the cameras will start with simple functionality tests. The cameras will be given to us by the client, as he already has a number of them. Then, the video stream could be created locally and looped back to the headset, allowing the user to see the world around them. This would be a successful test of the streaming functionality. Receiving the video would require multiple headsets attached to the same machine, then moving them to separate machines and networks. If problems are encountered with the camera itself, the Zed SDK includes a diagnostics tool that checks the camera and reports any problems with it. The Zed Mini support department will have to be contacted if the SDK cannot be repaired by the SDK.

3.2.2 VR Headset

3.2.2.1 Description: The VR headset comes as a complete unit, and thus does not require much assembly. However, this project requires a few modifications to the base unit, so those will be detailed here. This part of the project will be done at the end of Fall term, beginning the week of November 25th and ending three weeks later. This is due to the importance of the headset in the rest of development. While unit tests can be run on code and visualizations can be seen on a monitor, testing on the headset is the easiest way of indicating whether the software works.

3.2.2.2 Hardware: First, the headset itself must be acquired and examined. The client will work alongside the team to secure an HTC Vive. The headset itself will only require a test run to make sure that it works. First we will find a room large enough to set up the base stations, and with a computer that is powerful enough to run our VR headsets. Then we will set up the link box to our computer by connecting an HDMI cable to the link box and the PC's graphics card, and by connecting a USB cable from the link box to the a port on the PC. Then the we can connect the headset's 3-in-1 cable into the link box with the HDMI, USB, and power ports. At this point the VR headset is set up and ready to be tested.

The next step for developing our project is to connect our 3D Stereo camera to our VR headset. A ZED Mini camera

must also be secured by the team and our client. The headset will be modified by mounting the ZED Mini on the front of it. The ZED Mini comes with an attachable mount compatible with the HTC Vive. This mount will be placed onto the HTC Vive followed by the ZED Mini [12]. A USB-C cable will connect the ZED Mini to the headset, providing power. This is the only physical addition we will have to make to our headset. After these modifications, the headset is ready to be used for testing of the software.

3.2.2.3 Software: Developing software for the HTC Vive will be done in Unity. An SDK that will make VR/AR development easier is the SteamVR SDK, which comes with many useful features.

One of the features it gives is different views. The view can be changed within Unity to view how different physical components are monitored by the system. By putting the headset on, the headset and controller movement can be viewed through small markers on screen. This will be useful in finding and using motion to interact with the app. Within Unity, actions can be defined based on controller input via buttons or motion. These actions get stored in a JSON file, but the GUI provides a user-friendly way of editing them. Individual actions are put into action sets and bound to a controller. Again, Unity provides a GUI for this, giving a list of possible points of interaction for each button. These actions will be used to map buttons to functionality within the program, such as making, editing, and deleting annotations.

While this meets most of the requirements we need for the project, another useful feature that can be incorporated is a laser pointer. Using different colors for different users will allow our clients to indicate a specific component to discuss without adding an annotation. A cube object is created and stripped of most of it's original functionality. A C# script is used for the action to show or hide the laser. It also handles displaying the laser and scaling it so it looks correct to the human eye. The laser is then set to one of the controllers so it appears as though it's coming from the user's hand. This gives the user a good indication of what they're pointing at. For other users' lasers, a hovering point on the board would clean up the clutter of multiple lasers on screen. This can be achieved by using the ZED Mini's SDK to grab depth data. Then, the point can be drawn at the point the laser crosses the board's depth.

3.2.2.4 Testing: To test the connection itself, the ZED SDK will be used. Once the SDK is downloaded, the ZED Explorer program will be used. This will allow the ZED Mini to stream it's video feed to the HTC Vive headset. Once a video feed can be seen, the functionality of the headset can be confirmed. There will be more on this process later on in the document.

3.2.3 AR Headset Audio

Our software will have multi-user voice communication capabilities, and thus will need to take audio hardware into consideration. The HTC Vive includes a 3.5mm stereo audio headphone jack, so any standard headset will work with it. In addition, the Vive includes headphones with each unit, so that will eliminate the need to purchase audio hardware. There is an integrated microphone for audio input on the headset that can be used to capture voice communications, thus external microphones will not have to be used. This design will allow users to choose their own audio hardware without having to worry about a microphone.

In order to ensure that the audio equipment works properly, a software called SteamVR is perfectly compatible with the Vive VR headset. SteamVR can troubleshoot the hardware device and if the software detects any problems, steamVR will provide an error code explaining what is wrong. There is a lot of documentation and troubleshooting information

for the HTC Vive but if the user still can not find a solution to the particular problem; it may require the user to send a system report to the SteamVR mailing list so Valve can diagnose and hopefully solve whatever is plaguing the system or device setup[13]. It means that if the user encounter other difficulties, they can submit the software generated report to the manual support center by email.

3.3 Software Design

The second and more challenging piece of developing our project is the creation of our software application. A majority of our project work will be focused on this section. This will include the setting up of our development environment, creating the basics of our application, improving the user experiance, securing the transferred data, and configuring the network connection.

3.3.1 Visual Software and AR Software Development Kit

After researching many options for our 3D visual software development, we have decided to use Unity 3D for many reasons. The software is free, easy to learn, and the ZED Mini has a plug in that works well with Unity. The first step to creating our AR collaboration application will be to download Unity 3D onto our developer and testing machines. If we use the computers in Batchellor Hall at Oregon State University they have powerful graphic cards and the latest version of Unity 3D already installed. Therefore, all we would have to do is download the ZED Plugin for Unity which will allow us to actually use the stereo pass through from the ZED Mini and start development of our AR application. This plug in can be installed as a Unity package and immediately we will have access to assets, scripts, and sample scenes [1]. Once the Unity basics are set up the next step would be the AR Software Development Kit.

Not all visual software programs can work in 3D and not all 3D programs allow for development in AR. Research shows that Stereolabs has a SDK specifically developed for the ZED camera's allowing us to add depth and motion sensing to our application and many other features all for free. The software is available from the ZED website as well as the documentation for this kit. We will download the latest version of the ZED SDK onto our developer personal computers (PC) and import the ZEDCamera.package into Unity [1]. We can then read through the documentation to get caught up to speed on how the ZED Mini is developed and how we can use that to benefit our project.

Once the SDK is installed we can open and run the ZED Explorer. This will give us high definition (HD) video at 720p with 60 frames per second (fps) [1]. This will also automatically recognize the HTC Vive controllers and allow us to use them flawlessly. After all of that is set up and complete, we will be able to launch ZED World. This is the piece of the software that will allow us test and experience applications in the mixed reality world. We will start actual coding development on our project by creating a new 3D Unity project and making sure all of the ZED packages and plugins are downloaded and installed. Then we can create a new scene and add the AR camera prefab to the scene and enable positional tracking (which can be done in the inspector panel). These are key to starting our AR application.

A large requirement that our client has requested from us is that a user can place an object, like a circle, on the board to identify a pin or chip. To do this we need to use plane detection and object placement techniques. We will use the ZED mini to perform a scan of the real-world environment and create a mesh plane on the PCB. This will be done by creating a plane detection manager object that will create other GameObjects based off of different surfaces and add rigid bodies to these newly created objects. Then we will create arrows, circles, and other annotation objects that the user can place with the click of a button. All other annotations will be a variation of this.

3.3.2 Audio Software

To broadcast audio through the program, an audio broadcasting software must be used. WASAPI will be used for this project, as it's supported by Windows which is our developer OS (Operating System) of choice. A separate class for handling audio broadcasting will be used. All functionality will be abstracted to this class, allowing for all audio concerns to be handled by this one class for ease of code management.

The process of sending an audio message will begin by creating a WASAPI capture client object, activating it, and initializing it. This initializes an audio stream to be used to collect an audio sample. The audio stream will be connected between the software and the microphone on the headset. The capture client will get the size of the next buffer object using a built-in method. The buffer will be stored in a variable of the correct size. Then, the buffer will be released. Each buffer variable will be sent to the other users through a UDP package transfer. This process will continue until there is no more audio to be sent. However, since this will be an audio stream, this process will continue throughout the duration of the call.

The process of receiving an audio message will begin by receiving the sent UDP packages. These packages will contain the audio information to be played from the headphones. A WASAPI render client object will be created, activated, and initialized. Then, a built-in method will be used to get the default playback device of the computer. This will be the headphones used with the headset. The client will then use another built-in method to get a pointer to the next open buffer space in the returned playback device. The buffer will be filled with the next UDP package, and then the buffer will be released to play the audio back. This process will continue until there is no more audio to receive. However, since this will be an audio stream, this process will also continue throughout the duration of the call.

Both processes will be continuously running on all devices. The UDP package sending will have to send to all users, while the receiving part will have to receive from all users. This will be handled using a users array that keeps track of the current members of a session. This allows for the number of members to increase or decrease without affecting package distribution. Another problem can arise from receiving and playing audio from multiple users at the same time. This will be mitigated by adding packages to the buffer by arrival time. Each message sent will already be received in terms of arrival time, so this will not take much effort. Also, since the application is not focused on audio streaming, some distortion will not be cause for concern.

Testing this will require two or three endpoints to be set up. Testing with two connections at first will result in simpler tests overall. These tests will simply record for 5 seconds, capture the data, and send it to the receiver. This will be a simple way to test the entire audio capture class outside of unit tests. Then, scaling these tests out to more than two users communicating in real-time would simulate the conditions for a completed project. The UDP package transfer will also have to be tested on its own. This will mainly consist of ensuring the data is not changed during the transfer process.

3.3.3 Securing the Data

Security will be implemented using the AES (Advanced Encryption Standard) class by Microsoft. This is a symmetric cipher with a single password/key. This class works by having a function that will encrypt the data and a function that will decrypt the data. Encryption is done by converting the data into Hexadecimal format, divided into blocks of data,

combine the blocks with the key, and then passed through the AES algorithm to create ciphered blocks. We will be using a 128-bit key to balance security with performance speed of our software.

This software package has code written for encryption and decryption in the AES protocol, making it easier to implement and use. Within our project, there will not be a separate class will contain the security class, but it will be injected into all other classes it's needed in using dependency injection.

The first part of using AES is getting the key. This key will have to be known by all users of the software. If a user does not have access to the key, the software will exit. This key can be kept in a user secrets file, with each user owning a copy of the file. This seems to be the easiest way to manage this key. It also allows our client to distribute the key to other users as they see fit, allowing for scalability in the future.

The encryption process will occur every time before the data is sent between two users. Obviously the data must be encrypted before it is sent to another user to keep it secure. The data to encrypt must be in string form, but once converted it will be passed to an encryption function where the data will be encrypted and returned to the user. This is done using the encryption key and the built-in Microsoft AES class. The data will then be sent to any other users in the session.

The decryption process will occur every time data is received from another user. The data must be decrypted before it is used in the application. The data to decrypt must also be in string form for it to be passed to a decryption function where the data will be decrypted and returned to the user who received the encrypted data. The process is the reverse of the encryption process. After decryption, the data can then be used in the application.

Testing security will include unit tests and visual inspections of files being transferred. A simple visual test of security would be simply writing the package out to the console before and after encryption. If the contents before encryption match the contents after decryption, this shows the encryption and decryption process is working as intended. Also, visually inspecting the contents before and after encryption allows for verification of encryption. Passing in a readable string to the encryption function would be an easy test, since verification would only involve printing the encrypted string to the console and visually inspecting it. If the encrypted string is impossible to decipher quickly, the test will pass. Much like the audio software, since security is not a main concern in this project, the extensiveness of tests will be limited to ensuring the AES security class is functioning as intended. Trying to hack the software for testing will be outside the scope of this project.

3.3.4 Creating a Network

The transmission of 3d media streams will involve knowledge of the network. How to establish such a network connection is a problem. Our research will start with the data type API of 3d media stream. Study whether there are network interaction methods and software that can be used.

In order to exchange information, we need to find the right way to connect to the network. First, we need to figure out how to transfer the information among the internet. Data travels across the internet in packets. Each packet can carry a maximum of 1,500 bytes. Around these packets is a wrapper with a header and a footer. The information contained in the wrapper tells computers what kind of data is in the packet, how it fits together with other data, where the data came from and the data's final destination.[7] That means we can insert the information in a packets with

different protocols and then edit the format of the information. The format always depends on the data type and API (a Application Programming Interface). Before we actually establish a network connection, we need to choose the right network software to detect the network protocol and the transmitted data.

Some of the more popular examples of networking software include Logic Monitor, Datadog, Vallum Halo Manager and ConnectWise, among others. As always, it is recommended that the user takes the time to explore all the various options available and consult customer reviews before deciding on a particular product.[8] Most of these software either ask for some personal registration information or is not free. In another article called "10 Best Network Monitoring Tools & Software of 2019", it suggests some better network software.[9] We choose the software called: PRTG Network Monitor from Paessler. With the network monitor software, we will know if we send or receive the correct information by our software in the future.

Finally, based on different data transmission types, the corresponding network protocol should be selected. The data transmitted by this software should be a video audio stream. So we did research on protocols for media streaming. Unlike TCP, will not spend extra efforts on fixing errors with delivery, it'll proceed with sustaining uninterrupted flow of information. This feature makes UDP more suitable for live video streaming. However, due to the fact that TCP is widely used for various activities on the web, UDP transport protocol might be blocked by some firewalls. Furthermore, TCP is a preferable option for streaming video on demand or for those broadcasts when small delays do not make a big difference. [10] Therefore, my conclusion is that in the future we should use the UCP network protocol that does not accumulate transport packets and use PRTG Network Monitor software to detect and improve network connectivity.

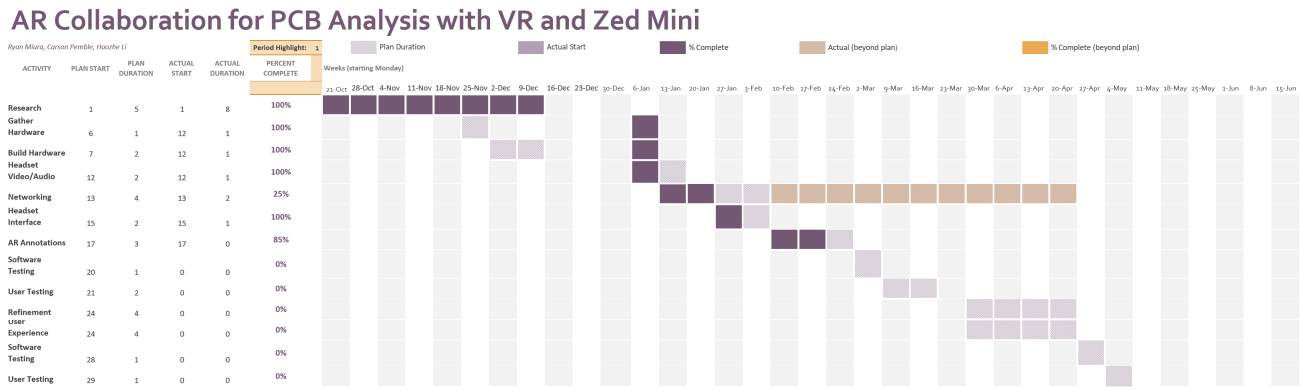
3.3.5 Virtual PCB Information

One of our stretch goals is for the user to be able to import a specific board schematic file and then have our software convert that into usable data for the user to identify the pieces and see information by just selecting the chip. If we want our application to be able to identify the key chips on the board, we will have to first use CAD (Computer-Aided Design) software to backtrack the files and related them to the boards we will be observing. After researching it appears that the best technology for this project would be to use DipTrace. This software has very detailed and complete 3D previewing would be extremely helpful for attempting this extra goal.

We can buy and download DipTrace from online from the DipTrace website. The key feature of DipTrace that we would be using is the 3D modeling and real-time preview features. After downloading and installing DipTrace we will then have to download the 3D libraries to allow us to use these features. These libraries are also available from the DipTrace website and we will be able to obtain 3D patterns, components, and all the items required to create a 3D rendering of the user imported schematic file.

We will do this by first having the user import a schematic file to our software. We will then have our software take that ASCII file, or P-CAD file, and import it into DipTrace [3]. Then we just open the tools tab and launch the 3D preview viewer. Once DipTrace creates the 3D model we will then exported and imported into Unity 3D as a new object. This way the user can rotate the board, zoom in on certain sections and see more detailed chips on the board.

3.4 Project Timeline



3.5 Conclusion

This document includes our plans for the development of our project. Things may change as the year goes on and more time is spent in certain areas, while other steps may take less time than what is currently planned. Overall, this is our road map for the setup of our hardware and the setup for our software environment. We will create the desired product for our client by following the steps previously stated within this document and have it completed by the dates above.

3.6 Bibliography

[1] Stereolabs.com. (2018). Unity - Getting Started with Unity — Stereolabs. [online] Available at: https://www.stereolabs.com/docs/unity/ [Accessed 30 Oct. 2019].

[2] Stereolabs. (2018). Building Multiplayer AR Experiences with ZED Mini. [online] Available at: https://www.stereolabs.com/blog/stereolabs-brings-multiplayer-ar-to-vr-headsets/ [Accessed 8 Nov. 2019].

[3] Diptrace.com. (2019). 3D Modeling - DipTrace. [online] Available at: https://diptrace.com/diptrace-software/3d-modeling/ [Accessed 8 Nov. 2019].

[4]. “Stereo Camera.” Wikipedia, Wikimedia Foundation, 25 Sept. 2019, en.wikipedia.org/wiki/Stereo_camera.

[5]. Getting Started - Introduction. (n.d.). Retrieved from https://www.stereolabs.com/docs/getting-started/.

[6]. “Oculus Rift vs. HTC Vive.” Oculus Rift vs HTC Vive — Which VR Set Is Best for You? - Newegg.com, https://www.newegg.com/vr/guides/oculus-rift-vs-htc-vive.html

[7]. Stricklan, “How IP Convergence Works.” HowStuffWorks, 8 Mar. 2010, https://computer.howstuffworks.com/ip-convergence2.htm.

[8]. Beal, Vangie. “Network Software.” What Is Network Software ? Webopedia Definition, https://www.webopedia.com/TERM/N/network_software.html.

[9]. Wilson, Marc, and Marc Wilson. “10 Best Network Monitoring Tools & Software of 2019: FREE - UPDATED!” PC & Network Downloads - PCWDLD.com, 30 Mar. 2019, https://www.pcwdld.com/best-network-monitoring-tools-and-software.

[10] “How to Sort Through the Variety of Streaming Protocols.” Letzgro, 25 Aug. 2016, https://letzgro.net/blog/the-variety-of-streaming-protocols/.

[11] “What Do I Do If My ZED/ZED Mini Is Not Working?” Stereolabs, https://support.stereolabs.com/hc/en-us/articles/360010101213-What-do-I-do-if-my-ZED-ZED-Mini-is-not-working-.

[12] https://www.stereolabs.com/zed-mini/setup/vive/

[13]Hesse, B. (2018, July 9). The Most Common HTC Vive Problems, and How to Fix Them. Retrieved from https://www.digitaltrends.com/computing/common-htc-vive-problems-and-how-to-fix-them/.

[14] “InitParameters Class Reference: ZED API,” InitParameters Class Reference — ZED API. [Online]. Available: https://www.stereolabs.com/docs/api/structsl_1_1InitParameters.html. [Accessed: 27-Nov-2019].

[15] “Getting Started - Introduction,” Stereolabs. [Online]. Available: https://www.stereolabs.com/docs/getting-started/. [Accessed: 27-Nov-2019].

4 Tech Review - Carson Pemble

4.1 Project Overview

4.1.1 Project Role

Throughout this project, I will personally be taking on the role of programming and implementing the visual side of things. I will be in charge of leading and implementing the visuals software, AR (Augmented Reality) software, and possibly the board layout software. We have two main goals through this project, create and augmented reality headset and then allow for the users to virtually mark it up and these are the tasks that I will be leading. If we have extra time I will lead the stretch goal of getting the board layout somehow on the video feed.

4.1.2 Project Goal

The problem we are dealing with is that there’s no easy way for colleagues to collaboratively annotate a PCB board remotely. Annotating these boards involves editing the schematic information as the physical layout is iterated upon. Currently, the process for completing this work requires all parties to be present at the same location. While this works, it restricts schedules and prevents efficient work from getting done. Users can send pictures, take videos, or even video call to show off a board, but this lacks a certain amount of control and efficiency that we would like to improve. Allowing each user to be able to view and interact with the board in near real-time would increase the amount of collaborative work that a team could complete. Current technology capable of this exist, such as the Microsoft HoloLens, but they can cost around $4000. This amount is not affordable to all, which can decrease productivity between colleagues. Without funding to purchase enough headsets, those wanting to collaborate on a board will have to view it in the ways described above. A lower cost alternative could solve all those problems. This is exactly what we will be attempting to complete through the process of this project.

4.2 Tech Responsibilities

As previously stated, I will be in the role of leading the visual software between the 3D camera and the HMD (Head Mounted Display), and also the creation of virtual annotations on the video feed. We will all help with the construction of the camera to the headsets, but then I will be leading the software connection between the two and make sure that we are able to visually see what the camera sees through the HMD. This will have to be done by using both software from the VR (Virtual Reality) headset and the software in the 3D stereo camera. On top of this I am the team lead with implementing the AR mark ups to show up on the video feed from the camera. To write or draw in the virtual realm I will write some custom software that works with the VR headset and markings should then appear on the screen in front of the user's eyes. Finally, if we have enough time we would love to add a stretch goal and I would be in charge of that. We would like to see if we can have the 3D camera scan the board and bring up the schematics or other documents about the board onto the screen allowing all users to see more detailed information about the PCB (Printed Circuit Board).

4.2.1 Visual Software

One of the key technologies that we need to have for this project is software that will allow for us to create visuals that will be displayed on the VR headsets. This software will allow us to combine the video input from the 3D camera and the virtual elements that we add from the users controls.

4.2.1.1 Unity 3D: Our first option for visual software that works with VR would be Unity 3D. This is probably one of the most well-known and widely used for VR development. Because of that reason it has the large community benefits, such as there is documentation and assets from their online store. It is really easy for beginners to start working on VR development with very little practice prior [1]. Unity is a game engine, but can be used for more than just creating games and works with all of the big-name VR headset companies. This software also has a VR preview mode which will be very helpful in testing our code and making sure we are getting the correct results [2]. Finally, Unity 3D is a very solid option for this project because the 3D camera's that we were told we will be using, ZED Mini, has a Unity plugin which would make our time more efficient instead of trying to re-writing the code that others have already done [3].

4.2.1.2 Unreal Engine: The second option for visual software would be Unreal Engine (UE4). Unreal is one of the next biggest and well-known competitors to Unity, and it too is a game engine but can be used for so much more. It is very similar to Unity with it also have a large community, good documentation, and Unreal is known to have even better graphics [2]. Unreal is also known for advanced optimizations in the development of VR applications. Unreal uses C++ which is known to more of our team members, rather than Unity which uses C#. Unreal supports multiple different operating systems such as Windows, Mac OS, Linux, Steam OS, and even iOS. One of the downsides to using Unreal is that there is no ZED Mini plug in available. This means we would mean we have to write code for something that already exists and would be a waste of time.

4.2.1.3 CryEngine: Another option that we could use could be CryEngine. Once again this is another game engine, but used for so much more. The key features about CryEngine is that has very strong visuals and a solid rendering quality. This would be helpful for our project if we were adding more detailed annotations and markings, but because of the simplicity of text and circling objects this isn't necessary for us. CryEngine does work well with the HTC Vive [1], which is the hardware that we were told we will be using, but the learning curve with the CryEngine can be a steep one. The

other downside to this software is that the very basic version is free, but to do more advanced work you will have to pay a fee.

4.2.1.4 Best Option: The best option for visual software for our project would have to be Unity 3D. The software is free, easy to learn, and the ZED Mini has a plug in that works well with Unity. Because of its price this definitely puts it above the CryEngine, not even mentioning that it has more VR features. Unity is very large and has documentation which will make the learning curve easier than Unreal and this will allow us to make more progress throughout this project. Last and most importantly the fact that there is a plug in for Unity which will allow us to create VR/AR applications easier and take advantage of the work that others have already put into the camera. There is also a ZED Mini SDK in Unity which includes many key items such as assets, scenes and scripts that we could use to boost our efficiency when developing our software [3]. Therefore, for this project the best fit technology for our visual software is Unity 3D.

4.2.2 Augmented Reality Software

Once there has been a decision made for the visual VR software, the next step is the AR software. This is the software that is responsible for placing the 3D object or text box into reality. The key features of the software we are looking for is one that contains content management, content editing, and good hardware integration [5]. Content management will allow us to store our assets and models, allowing us to easily use and access those files for development. Along with that it would be really beneficial to be able to edit that content and change it as we see fit. If the AR software can do that, then we will be able to work more efficiently in one software instead of switching back and forth. Finally, the ideal software will work smoothly with the hard ware that we are being provided.

4.2.2.1 ZED SDK: One of the options for an AR SDK is the ZED SDK. This is a software development kit from Stereolabs, the company that made our ZED Mini camera. This immediately checks of the hardware integration as well as software integration. The ZED SDK was developed for the ZED camera's and will allow us to add depth and motion sensing to our applications [3]. This SDK also includes sample scenes and has professional documentation which will simplify the learning curve. The latest SDK releases (2.8.x) also have added local network streaming capabilities which will help us when we are trying to implement the networking aspects of our project. They also have a working plane-detection system already implemented into the software, which will be very important for locating the board and adding virtual annotations on top of it.

4.2.2.2 ARKit: The ARKit is one of the most well-known Augment Reality Software Development Kits. ARKit is an Apple exclusive software kit, but since its launch in 2017 it has helped expand the innovation and development of AR applications worldwide [6]. The amount of apple devices in the world allow the ARKit to reach millions of people and it is fairly simplified for beginners to pick up, but advanced enough for serious application development. As with the other SDKs the ARKit has remarkable floor and wall detection, enabling easy blending of virtual and physical objects. Some other key features that the ARKit contains would be the following: SLAM (simultaneous localization and mapping), ambient lighting, scale estimations, and stable motion tracking [6]. This is such a largely used software for creating AR applications, but because our client is asking us to develop our software for non-apple products we will not be using this kit.

4.2.2.3 Vuforia Engine: The Vuforia Engine is a very handy SDK for web applications or business applications. Vuforia is known for having excellent device tracking which can have up to six degrees of freedom [7]. It also contains most of the key features desired in an AR SDK, 3D image tracking, content management, and it also has a special feature called the "VuMark". VuMark is virtual fiducial marker which can allow the user to orient an object in 3D space and allow for

correct rotations of digital annotations [6]. This is a really helpful feature and reduce the need for us to create some sort of physical sticker to use as a fiducial marker. The downside of Vuforia is that it does cost at least $42 a month for basic AR functionality features and also the fact that it is really focused for development on iOS and Android. Because we know that we are developing on specific hardware from our client, we will not be using this software.

4.2.2.4 Best Option: The best decision for an AR Software Development Kit is the ZED SDK. Once again this is a simple choice because it appears to be a great piece of software and the fact that it was created by Stereolabs for the exact product that we will be using means that this will save us time when developing our personal software. It is free which means we will be saving money and not have to request support from Oregon State. I also believe that we will be able to use the new feature of simple networking integrated into the latest update of the ZED SDK. Since we will be using Unity 3D, the ZED SDK and the ZED Unity Plugin the learning process should be centralized to just refreshing up with Unity and learning the structure of ZED software.

4.2.3 Virtual PCB Details

One of our stretch goals is to have the user be able to import the schematic of their PCB board and have our software point out the some of the key chips on the board. I think that we can implement the file importation on our own without special technologies, but converting the file into data that we can use in our augmented reality will be a little more challenging. After some research into PCBs and their schematic files, I think we could use some CAD (Computer-Aided Design) software to backtrack the files and related them to the boards we will be observing. There are many great CAD applications out there, like AutoCad and SolidWorks, but I think for our project we will used one that is PCB focused.

4.2.3.1 DipTrace: The first option that we have would be DipTrace. One thing that looks really helpful about this software is that it has Real-Time 3D preview. This allows you to view the PCB and all of its components installed and rotate the board along the x, y, or z axis. These images can also be exported in five different file formats. Sadly, DipTrace is not free; you have to pay $75 for the starter pack and up to $995 for the full version [8].

4.2.3.2 Eagle: The second option for PCB CAD software is Eagle from Autodesk. Eagle has a larger community than DipTrace, which allows for more documentation and library content. Eagle contains features such as schematic editors, PCB layouts, and now they also have a simple 3D modeler. The 3D generated model isn't as detailed as the one in DipTrace but because there is a free version of this software it does make it appealing. If desired there is a $100 version which allows for multiple layered boards [9].

4.2.3.3 DesignSpark: The final option is an application called DesignSpark. This is a free software that allows users to create schematics with unlimited size, design PCB with as many layers as desired, and also has a 3D viewer. The 3D viewer for this product isn't as detailed or as refined as the even the Eagle program, but it is very helpful that one already exists. DesignSpark is a software that is more focused on the creation of PCB and not the importation and inspection of schematic files [10].

4.2.3.4 Best Option: If we complete our requirements for this project and we still have time to implement our stretch goal, then I think that DipTrace would be the best option for a PCB CAD software. The very detailed and complete 3D previewing would be extremely helpful for our project. It does come at a cost, but I think it would be worth it. This would allow us to recognize what each component from the PCB schematic is and we can pass that information to the user when desired.

4.3 Bibliography

[1] Kraft, C. (2019). Getting Started With VR: The Best Software Tools Are Free. [online] Make: DIY Projects and Ideas for Makers. Available at: https://makezine.com/2016/03/24/makers-introduction-vr-best-software-tools-free/ [Accessed 30 Oct. 2019].

[2] Bitner, J. (2017). 11 Tools for VR Developers. [online] Lullabot.com. Available at: https://www.lullabot.com/articles/11-tools-for-vr-developers [Accessed 30 Oct. 2019].

[3] Stereolabs.com. (2018). Unity - Getting Started with Unity — Stereolabs. [online] Available at: https://www.stereolabs.com/docs/unity/ [Accessed 30 Oct. 2019].

[4] Dudkin, l. (2019). Unreal vs Unity for VR Development. [online] Skywell Software. Available at: https://skywell.software/blog/unreal-vs-unity-for-vr-development/ [Accessed 30 Oct. 2019].

[5] G2. (2019). Best Augmented Reality Software. [online] Available at: https://www.g2.com/categories/augmented-reality [Accessed 1 Nov. 2019].

[6] Romilly, M. (2019). 12 Best Augmented Reality SDKs. [online] DZone.com. Available at: https://dzone.com/articles/12-best-augmented-reality-sdks [Accessed 1 Nov. 2019].

[7] Library.vuforia.com. (2019). Overview. [online] Available at: https://library.vuforia.com/features/overview.html [Accessed 1 Nov. 2019].

[8] Diptrace.com. (2019). 3D Modeling - DipTrace. [online] Available at: https://diptrace.com/diptrace-software/3d-modeling/ [Accessed 3 Nov. 2019].

[9] Autodesk.com. (2019). PCB Layout Software Features — EAGLE — Autodesk. [online] Available at: https://www.autodesk.com/products/eagle/features [Accessed 3 Nov. 2019].

[10] Rs-online.com. (2019). DesignSpark PCB Software. [online] Available at: https://www.rs-online.com/designspark/pcb-software [Accessed 3 Nov. 2019].

5 Tech Review - Ryan Miura

5.1 Project Overview

5.1.1 Project Role

I'm taking on the developer role in this project. We will all be developing the software together, and thus share the same role. I've been tasked with researching VR headsets, audio software, and security protocols and methodology. The VR headset is a major component of the finished product, since it will be used to view the PCB board and any annotations it has. The audio software will be used to share audio between users, allowing them to communicate to each other outside of annotations. Implementing effective security measures is crucial to ensure the software cannot be corrupted or infiltrated in any way. Even if the subject being discussed isn't proprietary, leaving pathways into the software could lead to a larger infiltration of the surrounding system.

5.1.2 Project Goal

Our client faces the problem of collaboratively annotating printed circuit boards (PCB) from remote locations. There is currently no technology to facilitate this at a reasonable price, so we propose using a stereo camera attached to a virtual reality (VR) headset and software we develop to annotate PCB boards in augmented reality (AR). Our software will allow users to collaborate on board designs and view board physicality independent of one another while sharing visuals, audio, and user added virtual markups. This will result in saving the OSU research department time and money, advance affordable AR and VR technology, and help our client further his own research.

5.2 Tech Responsibilities

5.2.1 VR Headset

The VR headset will be used to view the PCB board and any annotations it has. There are a few options currently available from Oculus, HTC, and Microsoft. These technologies will be examined in this section.

5.2.1.1 Oculus: Oculus has two new headsets: the Oculus Rift S and the Oculus Quest [1]. While the Quest is a self-contained unit that can operate independent from a personal computer (PC), the Rift S relies on the power of a user's PC to boost performance[2, 3]. To avoid performance issues, the Rift S seems like the better option to examine.

The Rift S sits at a fair price of $399, making it a fairly affordable option for the purpose of scholarly research [3]. It features a single LCD display with a resolution of 2560 x 1440 and a refresh rate of 80Hz [4]. There are speakers sitting above the ear that replace standard headphones, however a 3.5mm jack is included in the device, allowing headphone use [5, 6]. This headset also did away with external tracking that Oculus has used in the past, and instead uses 5 built-in cameras on the headset to track the state of controllers and the position of the user [3, 4, 7]. It also makes use of the cameras to provide a feature they call Passthrough+, allowing the user to view the world around them through the headset [8].

5.2.1.2 HTC: HTC also has a number of different headsets, but I will focus on the basic HTC Vive, as this is what was requested from our client. The Vive features two LCD screens, each with a resolution of 1080 x 1200 for an overall resolution of 2160 x 1200. The refresh rate is 90Hz, and much like the Rift S a user can set boundaries of the area to avoid running into objects. It has a 3.5mm jack and includes headphones with purchase of the headset. There are many other ports to connect to HDMI and USB, as well as bluetooth compatibility. There are options to help with eye relief as well as a built-in microphone. However, this headset includes only a single camera on the front [9].

5.2.1.3 Microsoft: Microsoft's headset is the HoloLens 2. This is an existing version of the hardware we are trying to construct. It allows users to interact with holograms through the visor. The unit will cost $3500, making it an expensive piece of hardware to use. It would be harder to justify buying three HoloLens' over a cheaper alternative. It displays holograms on a see-through holographic lense at 2k resolution. There are six cameras total, four used to track head movement and two for eye movement. Speakers and a microphone are built-in to enable communication and voice-activated commands. This headset doesn't require the use of controllers as the others do, making it extremely user-friendly and intuitive [10].

5.2.1.4 Best Choice: For the headset, the HTC Vive will be used for a few reasons. First, our client specified that we use the HTC Vive with the Zed Mini camera to complete the project. It wouldn't make sense to purchase a headset with more cameras if we are attaching a separate one. Second, it is cheap. We were basically tasked with creating a cheaper

version of the HoloLens 2, so purchasing one and using it would not fulfill this requirement. This makes the HTC Vive a good choice for the purpose of this project.

5.2.2 Audio Software

The audio software will be used to share audio between users, allowing them to communicate to each other outside of annotations. There are a few options that work with C#: DirectSound, Windows Audio Session API (WASAPI), and Audio Stream Input/Output (ASIO).

5.2.2.1 DirectSound: DirectSound is a deprecated Windows component. It can be used to grab audio from a microphone and send it using UDP packages [11]. It creates a playback device connected to a computer's sound card, then uses buffers to manage captured sounds. These buffers are able to send the sounds to a playback device, such as a speaker, for communication [12]. It seems easy to use and there are code examples that can be found online, but the service is deprecated. This means it may not be supported in future Windows releases. Therefore, it is risky to use DirectSound, and it may have to be replaced in the future if implemented.

5.2.2.2 WASAPI: WASAPI is another Windows API that is used to manage audio communications from a user to an audio endpoint [13]. It makes use of an audio session, allowing multiple streams to be managed in a single session. An audio engine transports audio from an endpoint buffer to a playback device. This is a newer version of audio stream management, making it a safer option for streaming. There is also no information stating it's deprecated in the Microsoft documentation, so it shouldn't have any of the issues DirectSound has.

5.2.2.3 ASIO: ASIO is a third party audio streaming software development kit (SDK) developed by Steinberg [14]. It is available for free download off of Steinberg's website. It is specifically made for low-latency audio drivers, with smaller buffer sizes to improve performance over DirectSound. It is used to create audio applications for mixing and recording, with low level functionality provided. It is also not deprecated, making it another safe choice for implementation. It is compatible with many versions of C++, C#, and .NET, making it more flexible when dealing with code.

5.2.2.4 Best Choice: WASAPI seems like the best audio software, followed by ASIO. It's hard to decide on a single software since I've never created an application with audio streaming capabilities. The best way to decide on one will be to create some tests to run against both of them. However, WASAPI seems to take on a higher level scope, and since we won't be making an application focused on audio we won't need as much low level access. Also, since it's from Microsoft, it seems like a more reputable software to use.

5.2.3 Security

Implementing effective security measures is crucial to ensure the software cannot be corrupted or infiltrated in any way. However, security is a broad concept. Data encryption itself can be broken down into audio, video, and messages or requests. It is also a complex topic and will take effort to implement correctly. Still, there are a few options that will work for all formats. Advanced Encryption Standard (AES), Rivest Shamir Adleman Algorithm (RSA), and Digital Signature Algorithm (DSA) are all supported by Microsoft.

5.2.3.1 AES: AES offers good performance and good security [16]. It uses the Rjindael algorithm to encrypt the data. It encrypts a message by substituting, shifting, and mixing the data around, then adding the encryption key in a number of cycles. It is a symmetric encryption method, meaning it uses the same key to encrypt and decrypt. This can bring about some security issues, as if that key is exposed to the public anyone with the key would be able to decrypt a message in the middle of transit. It could then be changed and sent on, or that information could be leaked.

5.2.3.2 RSA: RSA is an asymmetric encryption method. Each user generates two keys, a public and private one. To send a message to someone, that message is encrypted using the receiver's public key. The receiver is then able to decrypt it with their private key [17]. Using this method, a message cannot be decrypted part way through without knowing the secret key. Each user will have their own secret key, so if one secret key goes public, all other messages will still be safe. This protocol works by generating keys, distributing them safely, encrypts them using modulus operations and exponent multiplication with a public key, and then decrypts them the same way with a private key.

5.2.3.3 DSA: DSA is also an asymmetric encryption method. This method covers a hole in the previous two: it's able to verify the message was sent by the correct person. It takes the data and sends it through a hash function, which is a way of encrypting the data. The result of this is then sent to a signing algorithm, where a signature is generated with the user's private key. Both the encrypted data and the signature are sent to the recipient, where the data is hashed again to get the original result. The signature is also verified using a verification algorithm and the signer's public key. If the result of the hash function and the verification algorithm aren't equal, the message is invalid [18].

5.2.3.4 Best Choice: I think AES would be the best way to implement security. While the other two algorithms are more secure, performance is a large concern in our project. We want to provide a low latency experience for our clients to make interactions as real-time as possible. Also, AES has a class implemented in Microsoft libraries already. The other two algorithms would require manual implementation. I think using AES will give an adequate level of security, performance, and implementation ease.

5.3 Bibliography

[1] https://www.oculus.com/?locale=en_US

[2] https://www.oculus.com/quest/features/

[3] https://www.oculus.com/rift-s/features/

[4]https://www.theverge.com/2019/3/20/18273152/oculus-rift-s-vr-headset-announced-pricing-release-date-features-gdc-2019

[5] https://www.windowscentral.com/should-you-upgrade-oculus-rift-s-rift-cv1

[6]https://www.roadtovr.com/oculus-rift-s-review-good-choice-for-newcomers-difficult-choice-for-vr-vets/

[7] https://www.youtube.com/watch?v=nrj3JE-NHMw

[8] https://gizmodo.com/the-new-oculus-rift-s-stops-short-of-being-truly-exciti-1833446012

[9] https://www.vive.com/us/product/vive-virtual-reality-system/

[10] https://www.microsoft.com/en-us/hololens/hardware#

[11] https://www.codeproject.com/Articles/19485/A-Voice-Chat-Application-in-C

[12] https://msdn.microsoft.com/en-us/windows/desktop/ee416964

[13] https://docs.microsoft.com/en-us/windows/win32/coreaudio/wasapi

[14] https://www.steinberg.net/en/company/developers.html

[15] https://www.codeproject.com/Articles/24536/Low-Latency-Audio-using-ASIO-Drivers-in-NET

[16]https://www.c-sharpcorner.com/article/introduction-to-aes-and-des-encryption-algorithms-in-net/

[17] https://www.c-sharpcorner.com/UploadFile/75a48f/rsa-algorithm-with-C-Sharp2/

[18] https://www.tutorialspoint.com/cryptography/cryptography_digital_signatures.htm

6 TECH REVIEW - HAOZHE LI

6.1 Project Overview

In the design phase of the product, you will be exposed to situations where remote cooperation is required. We often see Iron Man build a 3D rotating model of a building and communicate with remote teammates in the movie. Nowadays, the 5G era is coming, pure 2D video communication can not meet the truth and accuracy of the experience. PCB boards are often used in industrial design. As the name suggests, it is a collection of all circuit components. When designers want to interact through the network to improve the design of PCB boards, the authenticity and accuracy of communication becomes a big problem. Is it possible to improve the efficiency of remote assistance design PCB boards with VR devices? Mature VR real-time communication function device on the market: Microsoft HoloLens is extremely expensive. And we need a specific software to support PCB board design, cheaper and more efficient. So how do you implement the replacement of Microsoft HoloLens while making the remote PCB design process more efficient and smoother? We first need VR video input devices: Zed Mini and VR video output devices: HTC Vive or Oculus Rift. In addition to the hardware, we will be responsible for the software design

6.2 Tech Responsibilities

6.2.1 3D Stereo Camera

In this project, we will use the 3D Stereo Camera “Zed mini”. This is a camera that captures 3D images. This camera is different from the camera we usually use. Our team decided to do some 3D stereo camera research before getting this hardware from the client to make sure we understand how it works and what it does.

6.2.2 Audio Hardware

Our final software should have multi-user voice communication capabilities. As an interactive software, it ensures smooth VR/AR video streaming while also ensuring real-time voice communication. Before we receive the specific hardware, we want to determine which audio hardware is available for us to use. Which includes voice recording microphones and output headsets. Our products should also have requirements for the quality and speed of audio transmission. Research on audio devices is necessary

6.2.3 Networking software/method

The transmission of 3d media streams will involve knowledge of the network. How to establish such a network connection is a problem. Our research will start with the data type API of 3d media stream. Study whether there are network interaction methods and software that can be used

6.2.4 List possible technologies, methods, or option:

6.2.5 3D Stereo Camera:

First, I searched for definitions of stereo cameras on Wikipedia. A stereo camera is a type of camera with two or more lenses with a separate image sensor or film frame for each lens. This allows the camera to simulate human binocular vision, and therefore gives it the ability to capture three-dimensional images, a process Known as stereo photography. [1] Stereo cameras generally have two lenses, they capture different images and provide 3d object properties through program calculations. For example, in order to achieve the follow-up function of the display content in VR, the developer can analyze and calculate the lens image from the distance of two simulated human eyes captured by the stereo camera, thereby establishing a three-dimensional vector space. In the second step, I searched for the types of stereo cameras that are popular on the Internet. Karmin2 from "Nerian" company. Ensenso 3D cameras and the Zed Mini camera which we will use for our project in the future. Finally, I found some introduction information from Zed Mini website. In the introduction part, it says that the ZED is a camera that reproduces the way human vision works. Using its two "eyes" and through triangulation, the ZED provides a three-dimensional understanding of the scene it observes, allowing your application to become space and motion aware. [2] There are other some data provided by Zed Mini: video stream, depth perception and spatial mapping. Those data will help us determine the location of the Floating 3d object. We can tell that the stereo camera not only provides video but also focuses on content. Calculate the spatial position by calculating the difference between the distances captured by the same object in two camera lens. The software can calculate the threedimensional spatial position of the object based on these vector data. All in all, the zed mini stereo camera is a 3D camera that provides spatial position data. All in all, the zed mini stereo camera is a 3D camera that provides spatial position data.

6.2.6 Audio hardware

In order to achieve real-time voice communication during user communication, we need to first determine the audio hardware used. First, I want to determine if the VR/AR device we are using has audio input and output devices. I searched this question online. Vive includes a forward-facing camera that provides "augmented reality" within the Vive's virtual reality. At the moment, the augmented reality prevents you from running into real world objects while in a virtual environment. You can also "see" the world around you while in the Vive, allowing you to pick up a glass of water and take a drink. Unlike the Rift, the Vive does not come with headphones so you'll need to provide your own. [3] According to the search results, HTC vive doesn't have a mic and headphone because it usually allows the user to hear the environment sounds. So we need to find an adapted audio hardware for HTC Vive. Then I also did some research about the recommend headphone for VR/AR headset. Most answers suggest to change the audio output setting by the vive software. That means there is no specific microphone or headphone designed for VR/AR headset. People just switch the output and input setting of the software or system to change the hardware. Therefore, users can replace the default device of VR/AR headset with any of their existing audio recording and playback devices. The only think matter is the audio connection made by our software. Once the software itself has an output input port, any form of audio device can be used. Later I found that other versions of HTCvive have audio recording and output capabilities. Therefore, there is no need to worry about audio hardware until I am sure of the specific VR / AR hardware provided by the client. Even if the AR VR device provided by the client does not have any audio hardware, we can replace it with an external speaker or earphone. My conclusion is that for audio hardware, our software provides an interface that allows users to use any audio hardware.

6.2.7 .Network software/method

In order to exchange information, we need to find the right way to connect to the network. First, we need to figure out how to transfer the information among the internet. Data travels across the internet in packets. Each packet can carry a maximum of 1,500 bytes. Around these packets is a wrapper with a header and a footer. The information contained in the wrapper tells computers what kind of data is in the packet, how it fits together with other data, where the data came from and the data's final destination.[4] That means we can insert the information in a packets with different protocols and then edit the format of the information. The format always depends on the data type and API (a Application Programming Interface). Before we actually establish a network connection, we need to choose the right network software to detect the network protocol and the transmitted data. I searched for some suggested network software. Some of the more popular examples of networking software include Logic Monitor, Datadog, Vallum Halo Manager and ConnectWise, among others. As always, it is recommended that you take the time to explore all the various options available and consult customer reviews before deciding on a particular product.[5] Most of these software either ask for some personal registration information or is not free. In another article called "10 Best Network Monitoring Tools & Software of 2019", it suggests some better network software.[6] I choose the software called: PRTG Network Monitor from Paessler. With the network monitor software, we will know if we send or receive the correct information by our software in the future. Finally, based on different data transmission types, the corresponding network protocol should be selected. The data transmitted by this software should be a video audio stream. So I did research on protocols for media streaming. Unlike TCP, won't spend extra efforts on fixing errors with delivery, it'll proceed with sustaining uninterrupted flow of information. This feature makes UDP more suitable for live video streaming. However, due to the fact that TCP is widely used for various activities on the web, UDP transport protocol might be blocked by some firewalls. Furthermore, TCP is a preferable option for streaming video on demand or for those broadcasts when small delays do not make a big difference. [7] Therefore, my conclusion is that in the future we should use the UCP network protocol that does not accumulate transport packets and use PRTG Network Monitor software to detect and improve network connectivity.

6.3 Bibliography

[1]. "Stereo Camera." Wikipedia, Wikimedia Foundation, 25 Sept. 2019, en.wikipedia.org/wiki/Stereo_camera.

[2]. Getting Started - Introduction. (n.d.). Retrieved from https://www.stereolabs.com/docs/getting-started/.

[3]. "Oculus Rift vs. HTC Vive." Oculus Rift vs HTC Vive — Which VR Set Is Best for You? - Newegg.com, https://www.newegg.com/rift-vs-htc-vive.html

[4]. Stricklan Some of the more popular examples of networking software include Logic Monitor, Datadog, Vallum Halo Manager and ConnectWise, among others. As always, it is recommended that you take the time to explore all the various options available and consult customer reviews before deciding on a particular product. d, Jonathan. "How IP Convergence Works." HowStuffWorks, HowStuffWorks, 8 Mar. 2010, https://computer.howstuffworks.com/ip-convergence2.htm.

[5]. Beal, Vangie. "Network Software." What Is Network Software ? Webopedia Definition, https://www.webopedia.com/TERM/N/

[6]. Wilson, Marc, and Marc Wilson. "10 Best Network Monitoring Tools & Software of 2019: FREE - UPDATED!" PC & Network Downloads - PCWDLD.com, 30 Mar. 2019, https://www.pcwdld.com/best-network-monitoring-tools-and-software.

[7] "How to Sort Through the Variety of Streaming Protocols." Letzgro, 25 Aug. 2016, https://letzgro.net/blog/the-

variety-of-streaming-protocols/.

7 Blog Posts

7.1 Carson Pemble's Blog Summary

Week	Positives	Deltas	Actions
1	N/A	N/A	N/A
2	Chose a fun project	Submit Project Bids	Choose our project
3	First team meeting	Write Problem Statement	Talk with our client to understand the project
4	Agreed upon team standards	Team mate left and failed meetings	Try Slack for communication
5	More insight from TA	Failed LaTex formatting	Research LaTex and understand how it works
6	Early start on Tech Review	Failed client meeting	Improve communication between the team
7	Met with our client again	Need to improve our writing	Take more time to improve our first drafts
8	Improved our writing method	Failed client meeting	Improve communication with client
9	Team revision session	No communication with our client	Speak with professors about communication
10	Revised all papers	Haven't touched hardware	Get the technology from our client
11	Scheduled our meetings for winter term	CGEL is locked with Admin	Get the technology still
12	Acquired the two key components	Reschedule meetings and client out of town	Start development on project
13	Starting basic development	CGEL still doesn't have correct software	Get second set of hardware
14	N/A	N/A	N/A
15	Created good poster ideas	Team struggles to understand existing code	Finish the alpha requirements
16	Poster rough draft complete and project progress	I was really sick this week	Work on the security portion
17	Completed alpha and design review	Struggling with network	Improve security
18	CGEL has required software	Unable to contact client	Use Regex to allow passwords
19	Got something down for networking	Vuforia won't work with out project	Look into new VR lab
20	Updated Login Screen and Fixed SteamVR Update	GitHub mistakes were made	Fix teammates mistakes
21-30	No more blog posts	Re-approved our client requirements	Finished Project

7.2 Haozhe Li's Blog Summary

Week	Positives	Deltas	Actions
1	N/A	N/A	N/A
2	N/A	N/A	N/A
3	N/A	N/A	N/A
4	we are working on the second draft of the requirement	We can't getting connect with client	Keep emailing our client Kevin and then schedule another meeting with him
5	I attend the meeting with the whole group and the client Kevin.	I don't know what to start researching for the AR/VR API and SDK.	I will complete the tech research of my three topics.
6	I finished my tech review and start organizing everyone's work into one single file.	I don't know how to create a reference table in Latex	Learn about Latex online and ask Richard for suggestions. Reschedule the meeting with our client weekly.
7	I helped combine our tech review papers into one design paper. So far this project is going well.	My weekly meeting is in conflict with the meeting with the client.	Keep working together and meeting with each other. We plan to reschedule a new time for meeting with the client.
8	update the design document	We need to get in touch with our client to have him review our project	reschedule the meeting time with client
9	Meet with teammates and work together.	Still haven't heard back from our client	Keep working and keep the connection on Slack. Resend more e-mails to the client.
10	N/A	N/A	N/A
11	N/A	N/A	N/A
12	N/A	N/A	N/A
13	N/A	N/A	N/A
14	N/A	N/A	N/A
15	Working on the project, create a virtual keyboard	working along at home is so slow with a complex VR headset	we are going to meet together face to face to work on this project
16	connect with teammates about the details of alpha patch	We haven't meet together as the whole group to test our program	try to meet together and then try out our entire program and debug together
17	Finish the group design review	Can't have a place for the whole group to work together	keep connecting each other more frequently to make sure every one is on the same stage.
18	we meet each other with the equipment to work on the project	Still need ideas for the networking part	e-mail professor to ask for suggestions about the networking part
19	Keep working on developing the unity software	no solution for networking part	work hard
20	keep working for video	can't push changes on git properly	discuss with teammates on slack and research

7.3 Ryan Miura's Blog Summary

Week	Positives	Deltas	Actions
1	N/A	N/A	N/A
2	N/A	N/A	N/A
3	Formed groups	N/A	Meet with group
4	Requirements document	Failed client meeting	Start tech review
5	N/A	N/A	N/A
6	Met with client	N/A	Finish the design document.
7	Completed first draft of design document.	N/A	Gather hardware and hopefully get started on building the headset.
8	Revised documents	N/A	Meet with client
9	Revised documents	Failed client meeting	Set consistent meeting time
10	Was sick	CGEL computers	Obtain hardware
11	Obtained hardware	CGEL computers	Work towards alpha features
12	Began week's implementation tasks	Hard to contact client	Finish laser and audio implementation
13	N/A	N/A	N/A
14	Worked on networking functionality	Networking is more difficult than expected	Finish networking
15	Worked on networking	Unity networking transition	Finish networking
16	Completed presentation	N/A	Continue networking
17	Planned beta	Contacted client about networking issues	Work through beta tasks
18	Partially worked through networking example	No contact from client	Continue working on networking
19	Got networking audio working	Audio is laggy and choppy	Get integrated with main branch
20-30	No more blog posts	Re-approved our client requirements	Finished Project

8 Poster

COLLEGE OF ENGINEERING

Electrical Engineering and Computer Science

CS 13

AR Collaboration Suite

Augmented Reality Collaboration Application for PCB Analysis Using HTC Vive and ZED Mini.

Project Motivation

- Kevin McGrath, a CS/ECE professor here at OSU realized the need for a better way to work with his colleagues in Texas and other states.
- Our objective was to create an AR application enabling them to share a live visual stream from one of the locations. Both members with headsets should be able to markup and annotation the video stream as desired in a shared environment.

Project Requirements

- Our client requested that we use an HTC Vive or Oculus Rift as our VR headset, and that we used the ZED Mini as our stereo camera
- The software should allow multiple users (2+) to view and make notes on the live video stream recorded by the other headset in an almost real time experience.

Fig 1. Attaching ZED Mini to an HTC Vive.

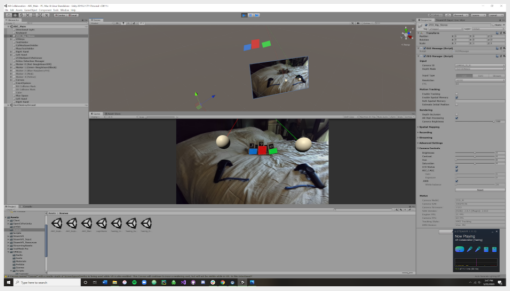


Fig. 2 Unity Project Screenshot

Unity Project Development Process:
There was a curve for all of the member as we had very little experience with Unity prior to this project, and there was many hours dedicated to reading different SDK documentation and understanding existing code. We began with basic AR features and integrated the more complex ones with time. Login/Registration uses password salt and hashing. User input is received through control input and a virtual keyboard. Audio is captured the HTC Vive and Video is captured via the ZED Mini. There is still work to be done, but the two year project is near completion.

Implementation

AR Hardware Design:
The camera was preselected for us to use, so we based our VR headset around the ZED Mini. We chose the HTC Vive due to connectivity and availability. The two pieces of hardware attach easily due to a mount from Stereolabs.

Software Design:
We chose to use Unity for the development of our project for many reasons. Unity is free, relatively easy to learn, and Stereolabs has an SDK plugin for Unity. We downloaded many packages from the Unity asset store including Steam VR 2.0, OpenCV, and VR Keys.

Outcome

Due to the global Covid-19 pandemic our project was impacted greatly by the shutdown of the OSU campus and separation of our team and hardware.

The outcome of this project was that our client, Kevin D McGrath has a great start towards the AR collaboration suite. With this application, users are able to annotate in the virtual world, track boards with ArUco images, identify key components with laser pointers and more. Networking development was slowed due to the inability to have in-person testing with the hardware, but still this project will be a large improvement to their previous remote collaboration, of Skype.

Oregon State University

Our Team

This team is a collection of senior CS students who all had the desire to work with VR technology and advance the forefront of the AR industry

Members: From Left to Right

Carson Pemble
pemblec@oregonstate.edu

Ryan Miura
miurary@oregonstate.edu

Haozhe Li
lihaoz@oregonstate.edu

Client is Kevin D McGrath
d.kevin.mcgrath@oregonstate.edu

Thank You

Our team would like to thank the following:
Kevin McGrath for letting us start this project.
Scott Fairbanks for helping us acquire hardware.
Richard Cunard for assistance along the way.
Kirsten Winters for keeping the project on track.

Fig 3. HTC Vive

9 Project Documentation

9.1 How does this project work?

What is its structure?

The structure of our project is the combination of two pieces of hardware, the ZED Mini and the HTC Vive. We used the zed mini stereo camera as video input device, and used the Vive virtual reality headset as the output device.The compilation and operation of the software requires the support of the unity physics engine and many unity C# script libraries.

What is its Theory of Operation?

Use a ZED stereo camera to capture and establish a three-dimensional coordinate system, use OpenCV reality enhancement function to calculate the target object space, decorate and edit the virtual PCB object and present it through the virtual reality headset.

9.2 How does one install this software

Currently unity OpenCV asset is not paid so we cannot build the program.

1.Please use git clone project repository. (https://github.com/CS-Senior-Project/AR-Collaboration)

2.Download and install the unity editor free version (https://unity.com/products/core-platform)

3.Import project folder with unity hub. Download and install the zed SDK for your system(https://www.stereolabs.com/developers/

4. Install the steamVR program(https://store.steampowered.com/app/250820/SteamVR/)

5.Then you should be able to open the program from the unity hub and run it by clicking the play button

9.3 How does one run it?

The Github README.txt has all the instructions for how to run our project. A brief description of how one can run our project is to clone the repository to a local machine. Download all the required software, and then the user can open the unity hub and finally the project. Once it is running, it is as simple as clicking the play button in the center at the top to run the program.

9.4 Are there any special hardware, OS, or run time requirements to run your software?

Please make sure you have zed mini camera and Vive VR headset properly connected to your computer first
The VR device setup may take a lot of time, please follow many online tutorials for deployment. Open the unity hub and open the project, click the play button in the center at the top.
Recommended Computer Specs Processor: Intel® Core™ i5-4590 or AMD FX™ 8350, equivalent or better Graphics: NVIDIA® GeForce® GTX 970 or AMD Radeon™ R9 290, equivalent or better. View the complete list ¿ Memory: 4 GB RAM or more Video out: DisplayPort 1.2 or newer USB ports: 1x USB 3.0 or newer port Operating system: Windows® 8.1 or later, Windows® 10. Upgrade to Windows® 10 for the best results with dual front facing cameras. Recommended graphics for the best experience is NVIDIA® GeForce® GTX 1070/Quadro P5000 or above, or AMD Radeon™ Vega 56 or above.

9.5 Any user guides, API documentation, etc. This needs to be detailed enough to recreate and/or use your project!

https://docs.unity3d.com/ScriptReference/Physics.Raycast.html(Unity's raycast method is used to detect the object pointed by the laser handle and calculate the coordinate point crossing the object)
https://www.stereolabs.com/docs/unity/
https://github.com/ViveSoftware/ViveInputUtility-Unity/wiki/Getting-started
https://www.raywenderlich.com/9189-htc-vive-tutorial-for-unity
https://valvesoftware.github.io/steamvr_unity_plugin/articles/SteamVR-Input.html

10 Technical Resources

10.1 Helpful Websites

[1] https://docs.unity3d.com/ScriptReference/ - Provides descriptions of many useful Unity classes.

[2] https://www.youtube.com/channel/UCa-mDKzV5MW_BXjSDRqqHUw - Youtube channel with many useful Unity tutorials.

[3] https://www.vive.com/us/setup/ - Getting started with the HTC Vive

[4] https://www.stereolabs.com/zed-mini/setup/vive/ - Setting up ZED Mini with HTC Vive

10.2 Reference Books

N/A

10.3 Helpful People

Mike Bailey is a graphics professor on campus that has access to CGEL, a potential development lab for VR projects. While our efforts to get the correct software installed on the CGEL computers failed, this may be a route for development in the future.
Richard Cunard was our TA and helpful friend in the development of this project. He answered any questions we had about the senior project or even about our AR development.

11 Conclusions and Reflections

11.1 Technical Information

11.1.0.1 Unity: Unity-based project skills were learned over the duration of the project. Simply managing a Unity project's dependencies was one, as this project depended on a few packages. Utilizing many of Unity's classes and getting familiar with them was another thing learned. Since Unity is heavily based on C# scripts, experience with C# was also gained.

11.2 Non-Technical Information

How to start and structure a project were important skills in this class. The entire first term was dedicated to planning the project, which showed how important this step is to successful development. The different documents helped examine various aspects of the project, ensuring some amount of planning was done for each before any code was written.

11.3 Project Work

Project work is easier to manage when broken into smaller pieces. When splitting into smaller tasks, each task becomes more manageable and this helps work get done. If these tasks aren't clear enough, or if development goes off the researched plan, project work can become more difficult. This can be avoided by testing available options prior to development when possible. Some complications encountered later in development may have been caught sooner had this been done.

11.4 Project Management

Managing a project requires consistent and clear communication between all members involved in the project. Without this, there are often breakdowns and lulls in progress.

11.5 Working in Teams

Working in a teams can be very hard at times, but overall it helps more than when working alone. As a team you should be able to get more work done. It gives all team members other people to rely on and to go to when stuck on issues. It also helps that everyone has about the same knowledge of the project. In teams, members are able to use their strengths in different tasks to create an overall better product. It's helpful to hear different perspectives and ideas on problems.

Working in teams can also be difficult at times. With other responsibilities, there are times when deadlines will be missed or communication will be late. This can lead to setbacks in development and progress. We have all learned a lot about how to handle communication errors and what to do when members aren't on the same page. This was almost as large of a task as our senior project was.

11.6 Things to do Differently

Meeting up in person to do paired programming or to work on the project together may have helped in making sure everyone was on the same page and that all work was being completed.

12 APPENDIX 1

If there is still questions about this project, this section should help clear some things up.

Setup instructions:
Clone Repo to your local machine.
Download ZEDSDK.
Download and Set up Unity (2019.2.17f1).
Download ZEDUnity plugin and import into Unity.
Download and install Steam (for SteamVR).
Download and run SteamVR.
Download and install Visual Studio 2019.
Open Server.sln file in Visual Studio.
Start the Server using the Run button at the top of the window labeled Server.
Set up HTC Vive and base stations.
Attach the ZED Mini to Vive and then connect the combined hardware to the machine (usb 3.0).
Open the AR Collaboration project in Unity.
Select LoginScene (Project / Assets / Scenes / LoginScene) and press Play.
To view audio, build and run the project (File / Build And Run).
Select Main Scene (Project / Assets / Client / Scenes / Main) and press Play.

Troubleshooting:
Controller Input Not Working at all?
Select the "Window" / "SteamVR Input Live View" and you can see if the controller is registering the input. If it still doesn't work, make sure you have "Steam" open and running on your system. Steam has constant updates and SteamVR must be up to date to work properly.

If Laser ON/OFF is tracking the wrong hand?
Click on the "Right Hand" object and switch the device to 4 (if it is currently 3) or 3 (if it is currently 4). Do the same for the "Left Hand" object.

AR Image Marker isn't tracking correctly?
It could be that the size of your image is incorrect; AR Images are set to the default size of 5cm. If you print off or use

ArUco images that are a different size, you can change this size within the project by opening the “ArUco Detection Manager” object and adjust the “Marker Width Meters” from 0.05 to the correct size.

Virtual AR Image plane isn’t the correct size?

Currently Marker 0 (ArUco Marker ID 0) is set to the correct size for a Beagleboard XM. Marker 1 is set to a Beagleboard Black, and Marker 2 is set to a Raspberry Pi 4. If you want to assign an image to a new board, you can adjust the size of the corresponding cube, in each Marker object. To add a new board select the “Marker 3” object and then select the “Cube” child object of “Marker 3”. You can , then adjust the “Transform” and “Box Collider” components to fit the correct size and placement of your board.

13 Appendix 2

Videos:

Unity SteamVR Laser Pointers:

https://www.youtube.com/watch?v=TyEzaj06Xao

Unity VR Keyboard:

https://www.youtube.com/watch?v=oWSy6tNAFjo

ARC Login and Registration Demo:

https://www.youtube.com/watch?v=8XOWeidmCT8&t=33s

ARC Tracking, User Input, and Laser Pointer Demo:

https://www.youtube.com/watch?v=vKveGMF2JH0&t=39s

ARC ArUco Image Tracking:

https://www.youtube.com/watch?v=EE4Pp5UHN30

ARC Project Clone, Build, and Demo:

https://www.youtube.com/watch?v=cJsmnEvMhjY&t=2s

14 Appendix 3

Here I have included the criticisms from the code review and our responses to each criticism in the table shown below.

Category	Description	Reviewers Comment	Action taken by reviewed group
Build	Could you clone from Git and build using the README file?	Cloning was fine but could not build due to hardware limitations. README is clear and understandable.	N/a
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	Flow was sane and understandable. Some files were missing comments.	Comments were added to the files that were missing comments and all other comments were improved to be more descriptive.
Implementation	is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	Using relative file locations or dynamically determining file location would be better for login. Also, moving ControllerInput class functionality into other functions.	User login info stored in a relative file, making it safer and less error prone. ControllerInput class removed.
Maintainability	Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?	Did not see any unit tests.	Decided that, due to the nature of the project, viewing the scenes through the headset would prove more useful than unit testing.
Requirements	Does the code fulfill the requirements?	The requirements established in the requirements document are met.	N/a
Other	Are there other things that stand out that can be improved?	Using hash functions like BCRYPT for login.	Stuck with SHA-256 as it seems to meet the needs for our project.