CS 462 Winter Final Report

March 17, 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

Throughout this term our team has completed a majority of our augmented reality collaboration software, but there is still work to be done. All of the main components have been implemented; these include controller tracking, user input and annotations, software security, and audio networking. There are extra features we have still to complete, such as board tracking and video networking. We encountered many problems this term, ranging from communication failure and acquiring hardware to Steam updates and Unity's shift in networking. Thankfully the ARC team has pushed through and been able to overcome most of these issues.

CONTENTS

1 Our Project

The project that our team chose is an augmented reality (AR) Printed Circuit Board (PCB) collaboration software application. With a 3D stereo camera attached to the front of an HTC Vive VR headset, we will create our own AR headset. Our project will allow the user to simulate a first-person interaction with a PCB board in augmented reality for other viewers to see and interact with. The camera will be set at the correct interpupillary distance to mimic what the main user (broadcaster) is seeing, and the camera feed will be transmitted by our software to display the same environment in real-time to the other users (viewers) who will be viewing the displayed feed on their own headsets. The other viewing users will be able to annotate and interact in the same AR environment so the broadcaster can see their annotations as well.

2 Project Purposes and Goals

The purpose of this project is to help our client improve his PCB design through the use of a virtual reality collaboration suite. Our goal is to create software that will allow for our client and his colleagues to present information about various parts of a printed circuit board and possibly other technologies in a 3D virtual reality video conversation. With the addition of augmented reality annotations, near real-time audio, and laser pointers this new form of communication should allow for further advancements in our clients research and improve long distance project collaboration.

Allowing each user to be able to view and interact with the board in real-time would increase the amount of collaborative work that a team could complete for many reasons. Also, being able to handle a single board independent of other workers without making copies would cut back on resources and research expenses, while still increasing productivity.

3 Current State

The ARC team worked hard this last term in developing our product and its current state is still in development. We have completed many steps towards our final project; some of the key features we have completed are setting up the hardware, implementing AR tracking spheres, AR laser pointers, VR keyboard, user created notes, adding security features, audio recording, and steps towards networking, with audio being networked. We also came up with a few drafts of our poster that we will be presenting at the expo.

3.1 Hardware

Currently, we have two HTC Vive Headset units and two Zed MINI cameras. We've mounted the cameras to the front of the HTC Vive headsets to allow for video pass-through in our software. We've also installed the software needed to work with these hardware components, such as the Zed SDK, Zed Plugin, Unity, DirectX11, and OpenGL Core. Also included in the Unity projects is the SteamVR package available on the Unity asset store. This provides functionality for the HTC Vive Headset.

3.2 AR Spheres

The first step in our project was to allow the user to create virtual objects in the AR video. We started with creating spheres and having them track to the Vive controllers. This was done by importing SteamVR and writing code that would attach game objects to the scene and adding SteamVR components to those objects. This showed that we could track the users controllers and add virtual objects to our scenes.

3.3 Laser Pointers

One of the items we thought would improve the user's collaboration is the ability to have a laser pointer. This allows for the user to use a controller to point to something very detailed while conversing with the other members in the collaboration suite. This was done by adding the SteamVR light component and then editing it so that the user could turn it on and off with a pull of the trigger. We added a laser to each "hand" object, each with a different color (See Fig 2).

3.4 Keyboard

Another key feature that we wanted to implement was to allow users to type notes into the software. Through our research, we found an AR keyboard asset from the Unity asset store. Virtual keyboards are not provided in the official SDK for Vive headsets, therefore we chose the current AR keyboard asset because it was free and we felt like we could use that to achieve our goal without having to rewrite existing code. The typing method of this keyboard is more efficient than the traditional laser pointer virtual keyboard input method (See Fig 3). After added the keyboard to our project, our team changed the code so the user could press a button to easily bring up the keyboard and remove it from the camera view. With this feature added the user could now type and bring text into the application without having to go back to the physical keyboard, improving the user experience.

3.5 User Notes

With the user being able to type within our application, we thought it would be really helpful if the user was then able to place that text where they wanted. We wrote code that allows the user to type whatever kind of note they want with the VRKeys. Then the user can press the "grip" button on the Vive controller to place that text anywhere in the VR environment. That text will then stay there at that location even if the user walks around or leaves the room.

3.6 Security

For security, we've implemented a basic login system with the ability to register new users (See Fig 4). With this, we can make sure our system is only accessed by those who are allowed to access it. We have also used regular expressions to pattern match the fields and ensure valid inputs are given by the user. This means the user will have to create a password that is strong with an uppercase, a lowercase, a number, and a special character. Another measure of security we've implemented with this login system is hashing the passwords with salts to make sure they aren't openly readable where we store them.

3.7 Audio

For audio, our team used a Unity component script to capture audio from the microphone already integrated into the HTC Vive headset. This audio is stored in an Audio Clip object and gets sent to a server via TCP every 1 second. The server then takes this clip and sends it to all connected users every 1 second. This ensures the audio is synced right. There are limitations with quality and latency, as the audio seems to come in a bit late. However, it is understandable and clear but it will be improved upon before expo.

3.8 Networking

For networking, we've run into the most trouble here. As explained in our design review, Unity is currently in the midst of a networking system transition from UNet to NetCode. NetCode is currently in alpha without much documentation. Our client would like us to use NetCode with the project, which is something we will be working towards next term. This specification was given to us a bit late in the project, so our existing code uses a C# server with Unity clients. As of now, our audio is the only thing we could get networked. However, we've taken time building up our networking classes so they are ready to receive data such as Vector3 positions and Quaternion rotations, data types often used by Unity. This will allow us to send these values rather than entire objects over the network, simply instantiating them for each player with the given values. The server and clients also include code for sending packages in both TCP and UDP, giving us flexibility in the future.

3.9 Poster

For our poster, we made a rough draft, filling in all the basic information about our project. Once that was graded with feedback we constructed a second draft and have made improvements where we lost points the first time. It's fairly complete, but could use one more look over prior to expo to ensure it's within university standards.

4 Left To Do

We've made a lot of progress on our project this term, but there is still a lot to do. At the technical level, we need to make more inner components compatible with each other in order to achieve the final requirements of the project. First, we want to use Vuforia AR markers to identify the PCBs and record the position information of the PCB in a 3D space (See Fig 5 and Fig 6). This will allow us to orientate the board with the annotations, so all users see the annotations correctly. Next, we need to establish an effective network connection to transmit the 3D video stream with real-time position data of the PCB between the two users. By doing this, users will be able to share visuals and interact remotely without a large time delay. Finally, we want to encrypt the transmitted information to ensure the security of the transmitted information.

In addition to the technical aspect, there are other tasks we still need to complete. Our team still needs to complete the registration process for attending the Engineering Expo in May. We have yet to place our hardware requirements as we are still in discussion with our client to see if we would be able to have all the hardware required to demo with the public or not. Finally, we will finalize our poster and make sure it is print worthy and meets the OSU requirements, and once that is complete we should be done with our project and ready to present at the expo.

5 Project Problems

This project has definitely had some unforeseen problems to it. We ran into individual team member problems, problems with our client, problems with finding a place to work together on campus, and many software issues.

5.1 Individual Problems

This term we encountered more individual problems than we did last term and they made this a harder for the whole team. Ryan M was sick during the first two weeks of the term and Carson P was sick during week seven. This made

TA meetings and communication amongst the team slightly more challenging. Development was slowed during these times. Each team member was also challenged with balancing this class with our other classes. This is our senior year and other classes demand more time than the previous years. Our team members try to prioritize this project, but it is hard with very few deadlines and other professors requesting more and more of our time for their classes. This was more of a personal problem and something that we each had to individually work on throughout the term.

5.2 Communication

One problem we ran into this term was proper communication in general. Both our team and our client had some difficulties with communication. This made it challenging to contact or meet with our client. We had to reschedule both our TA meetings and our client meetings. There were class time conflicts with our original TA meeting time, and our client meets were at a bad time for a client with a young child. Poor communication caused us to miss a few meetings, and eventually we just canceled our weekly client meetings and decided to only use email. Email had a whole set of complications as well.

Email was possibly our largest non-product issue we had this term. We started the term with an understanding that it was hard to reach our client last term and throughout the term we learned that our online meetings weren't really working either as the invites to Zoom were through email. Not being able to get a fast response from our client is still slowing the project development and it makes it hard if our team has any clarifying questions about the project's future.

5.3 Hardware

Last term our team was told that our client would support all our hardware needs. The first few weeks of this term there were some issues with actually getting the hardware from our client as he is only Corvallis once a week. Our client was also out of state at CES which meant we weren't going to be able to start project development until we received hardware, which could have been much longer. Once we had one VR headset and camera we realized that project development would be extremely slow if we have to share the one headset between the three of us. It is possible to work on the project without the hardware, but testing is much harder. This meant we needed to acquire another headset and camera and that process slowed down development again.

5.4 CGEL Lab

Our team spent most of this term trying to find a location where we could all three come together and work on the product together. We thought the Computer Graphics Engineering Lab, CGEL, would be a great place to go as we had already been approved by Mike Bailey, the professor in charge of the lab. Early on in the term we had our ID cards given the ability to enter the lab, but we were not able to download the software for our project as the computers lock installations without administrative credentials. After six weeks of communication with Mr. Bailey and the IT team, we finally had the software we needed installed on the computers. With further inspection we discovered that there was still the wrong version of Unity installed. We requested that the newer version was installed, but never heard back. If this process takes another six weeks, we may never use the CGEL lab for team development.

5.5 Software

Another set of problems that we had to deal with this term was all the problems encountered with our code. One issue we have is that SteamVR, a large component of our user input and tracking, is always being updated with new code being added and old code being altered. This has been an issue a few times in our project when the code is working perfectly one week, and then after a required update the project is full of errors and warnings. We also ran into a problems with how the project restarts every time it is run. This means that every time the user runs the program, you need to check the controller bindings and make sure they are set correctly. There doesn't seem to be a way around this, but it does take a few minutes and impacts the user experience. The biggest problem that we faced this term was definitely figuring out how we are going to implement the networking feature of our product.

5.6 Networking

Unity is currently in the middle of changing their networking service from UNet to NetCode. This led to many problems when developing our networking code. UNet had the basic setup we needed, with a way to connect to other clients through Unity-hosted servers. It also provided an interface as a built-in component, making it simple to use. However, UNet is now deprecated and the servers behind it are shut down. This meant UNet was no longer an option.

NetCode is currently in its alpha state and won't be fully released until the end of 2020 at the earliest. While some aspects are working, there isn't much in terms of documentation and examples, which made development difficult. While there were examples from Unity, the documentation behind these mainly explained what code to write without much explanation on what it did.

The option Unity suggested for networking during the transition, their low-level API (LLAPI), was also difficult to understand, with the main example containing many scripts and minimal documentation. Another note with the LLAPI was that development would consist of rewriting deprecated UNet code, which would make the project out of date as soon as NetCode comes out. Therefore, the LLAPI did not seem like a good solution either.

6 Project Solutions

As problems arose this term, the ARC team did our best to find the most effective solution as fast as possible. Not all problems have been completely solved, but most of the issues at least have temporary solutions, while others have been solved and no longer hold us back.

6.1 Individual Problems

The individual problems were mostly solved with individual solutions. There wasn't much that our team could do about being sick, except to get sleep and hope for a fast recovery. We were able to jump right back into working on the project once the sickness was over. Our team was also able to spend more time working on the project with the weekly reports and TA meetings being a time to check up and make sure that we were all still allocating the right amount of time to work on the project.

6.2 Communication

The way that we are currently overcoming the communication problem is to use Slack for our team communication and lots of emails for client communication. As a team we are informing each other of our progress and keeping ahead of the

deadlines. When emailing the client all team members are to be CC'd to stay up to date on the information. The issue with our client not responded hasn't been removed but it has been improved upon. About halfway through the term we talked to Richard Cunard, Teachers Assistant, and he informed us that our client is most likely to respond to emails if we constantly resend them. Although this wasn't the professional advice we expected, this has helped our team to get more email responses from our client. This may not be the best long term solution, but as of right now this might be the only solution.

6.3 Hardware

The problem with acquiring the first set of hardware was solved by talking to Scott Fairbanks about getting hardware from him and not going to our client. This allowed us to get the hardware much faster, but we were still missing the ZED Mini camera. The way we solved that issues was to look up our clients office hours and to visit him there and to ask for the camera then. Once we had both pieces we were finally able to start development in week three. With the second set of hardware we had the system down. Scott helped us get the Vive in less than two days and we got the second camera within a week. Now we are able to have two members of the team working every week.

6.4 CGEL Lab

The CGEL lab has been a constant problem and the best option for our team is to stop wasting time with trying to get the room to work for our project. We have considered looking into other labs on campus such as the VR lab run by Rafael Di Amici. If those labs are also locked down by administrative credentials and the installation process with IT takes a long time, the best solution might just be to do our development separate and try to test on our own.

6.5 Software

Software issues were handled with software. As we develop with SteamVR, the current solution is to copy the files and editing them into new files, so they won't be effected with an update. This way our code will stay working and if we want to implement what ever features were added in the update we will have both files still in our project. We are still looking into a way to store the controller presents in our application, but at this time the solution to have a trouble shooting document that goes along with our application and it explains how to bind the controller action sets.

6.6 Networking

To help solve this problem, we brought it up in our design review. The other groups provided feedback and offered solutions that had worked in their projects, but these solutions didn't transfer to Unity in a way that worked for our project. We also contacted our client separately from this meeting to ask for advice as we continued looking for a solution.

In the end, a series of videos online held the solution. The videos walked through creating a C# server with a Unity client. This method worked and allowed for the audio to be networked. This was as far as we got prior to beta as this solution wasn't found until near the end of the term.

Another thing to mention is that our client wrote back to us while we were working on following the video series, requesting that we try and use NetCode as best we could. We decided to move forward with the working solution for now, but will make another attempt at NetCode early next term. If we're unable to find a NetCode solution, we may have to proceed with the C# server.

7 Interesting Pieces of Work

A few things that our team found interesting through our development this term were how SteamVR handles user input,

One thing that was really interesting to work with is the SteamVR 2.0 action sets and the controller binding UI. When using Unity and SteamVR together you can create new lists of actions that you want the user to be able to perform. It becomes even more fun when you are applying those actions to buttons on the controller, through the binding UI (See Fig 1).

Another interesting thing to work with was the process of capturing audio in Unity. Unity provides an easy way to find connected microphones and record audio through them. This audio is stored in an AudioClip object, which provides methods for converting the audio to float arrays. This made sending the audio over a network much easier, as we didn't have to deal with networking a complex Unity object with no Unity support. Hearing our voices played back in a project created by us was interesting, and hearing them over the network was a breakthrough in the networking development (See Fig 7 and Fig 8).

8 Project Screenshots

Fig. 1 Assignment of the ARC action set onto controller binding.

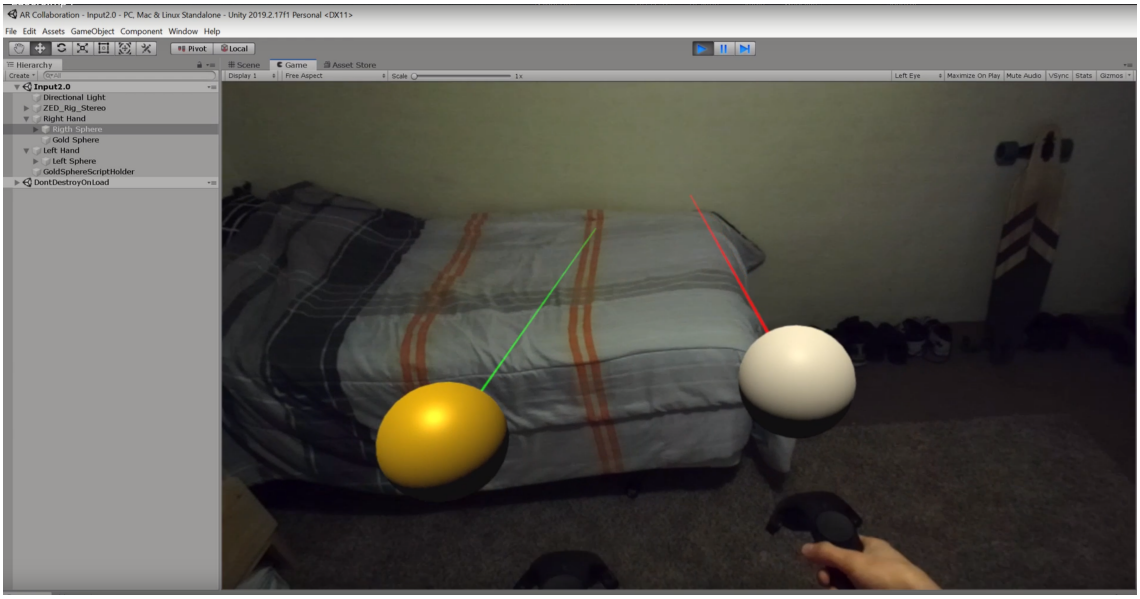

Fig. 2 AR controller tracking spheres with both laser pointers active.

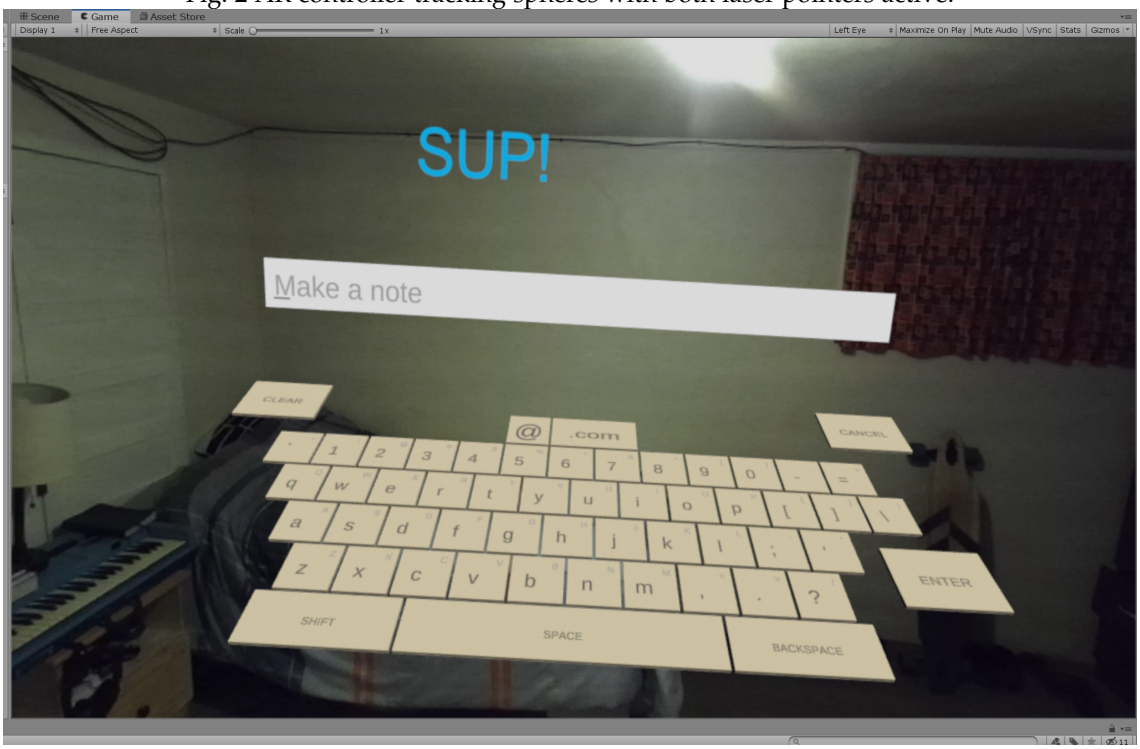

Fig. 3 The VRKeys virtual keyboard and the text place holder in blue.

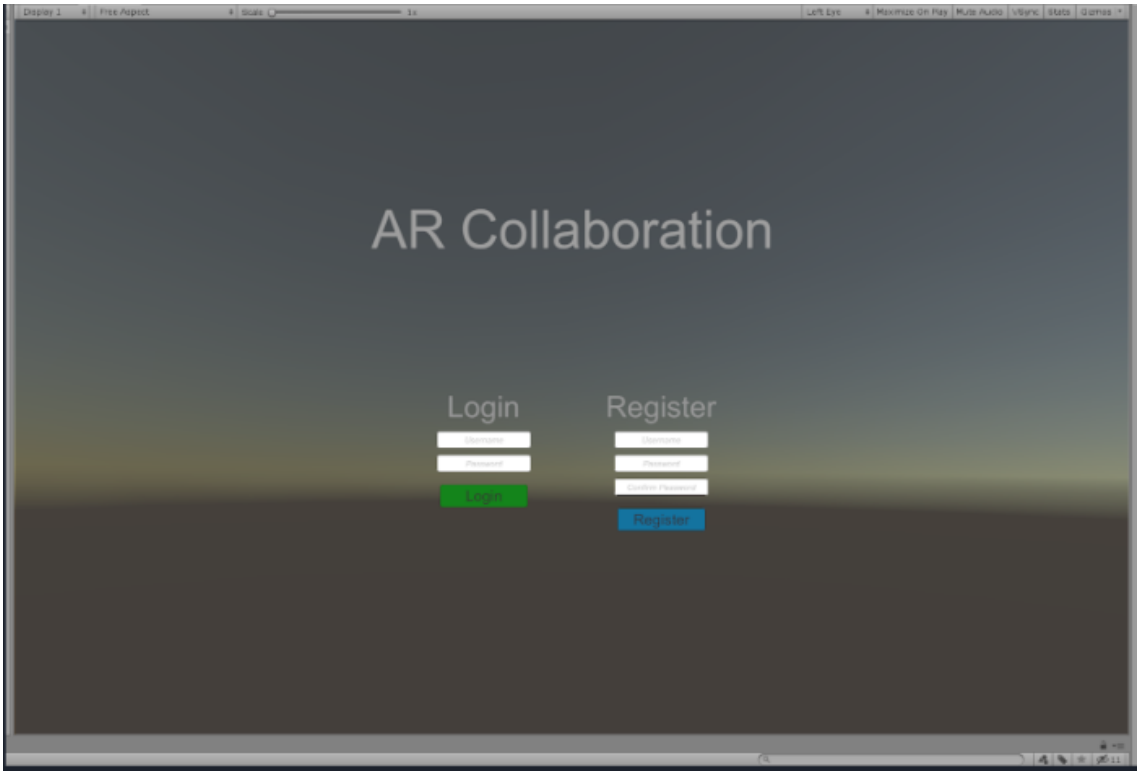

Fig. 4 The collaboration suite registration and login page.

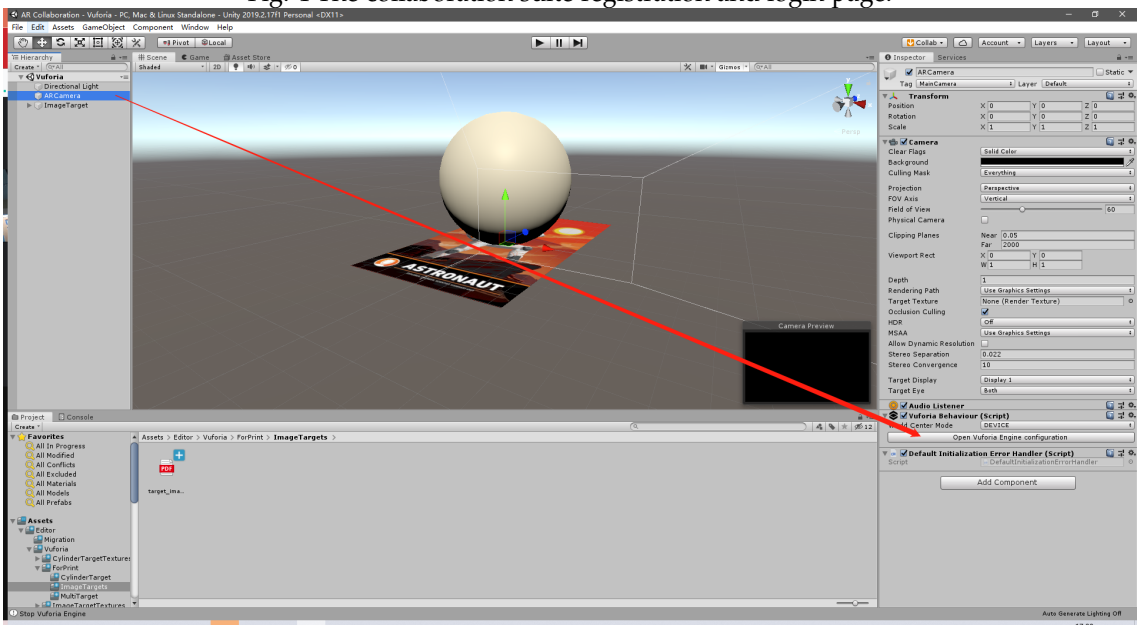

Fig. 5 Vuforia image to be tracked and test virtual object.

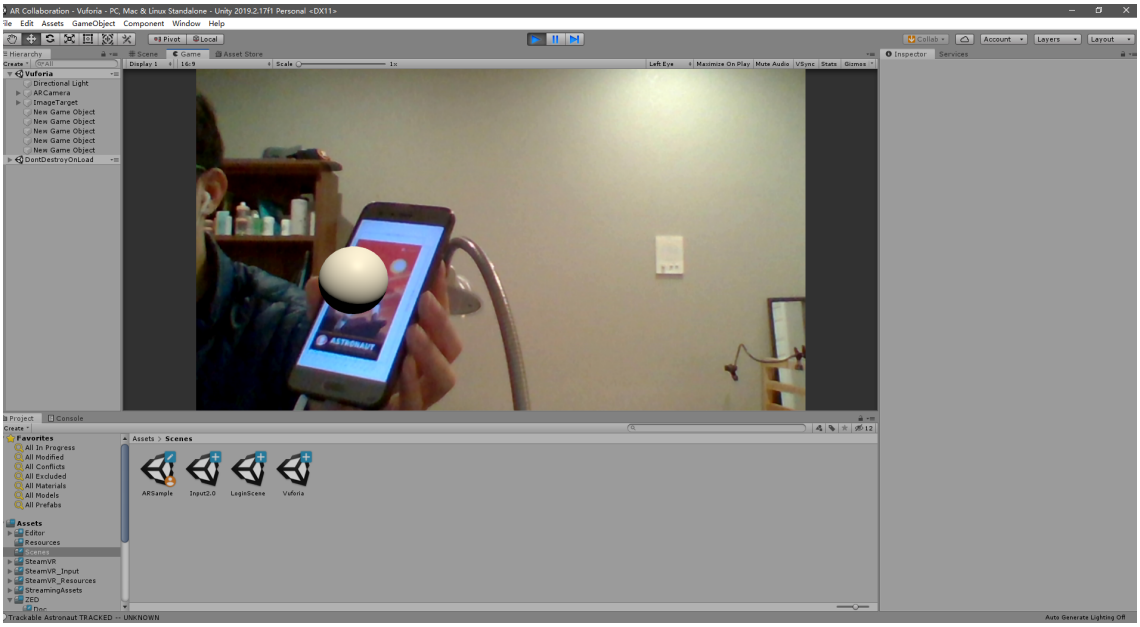

Fig. 6 Vuforia being tested within our project

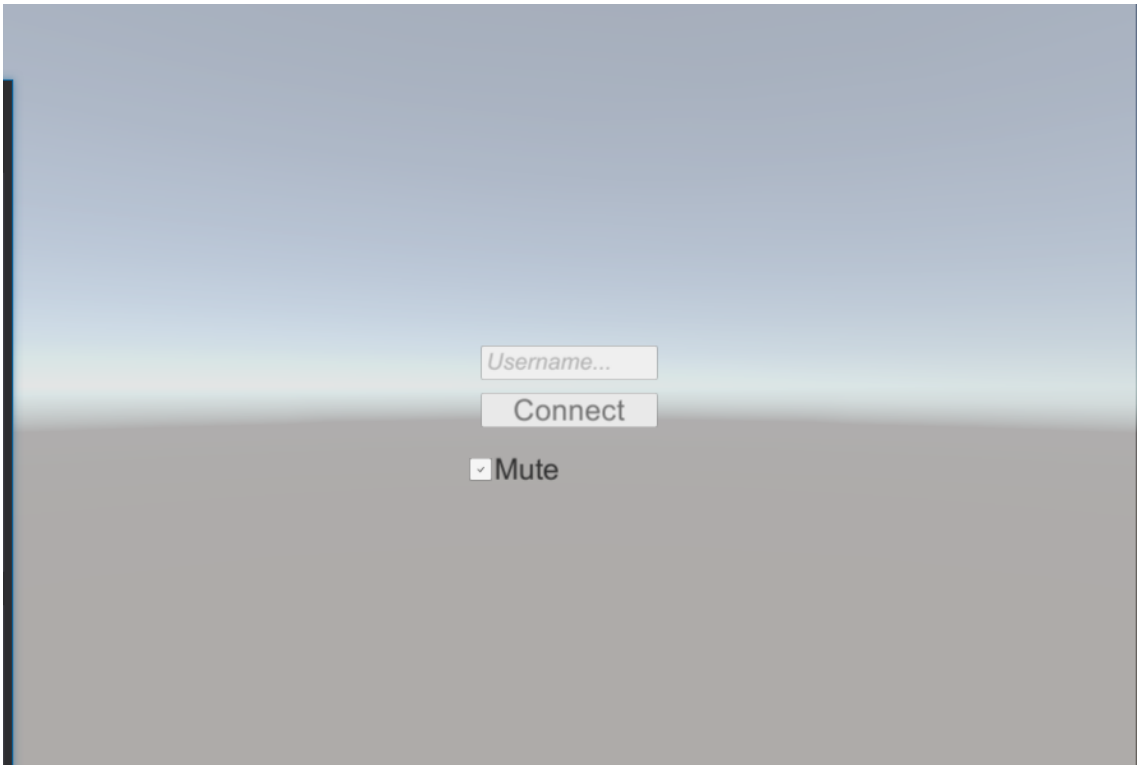

Fig. 7 Client screen for connecting to the server

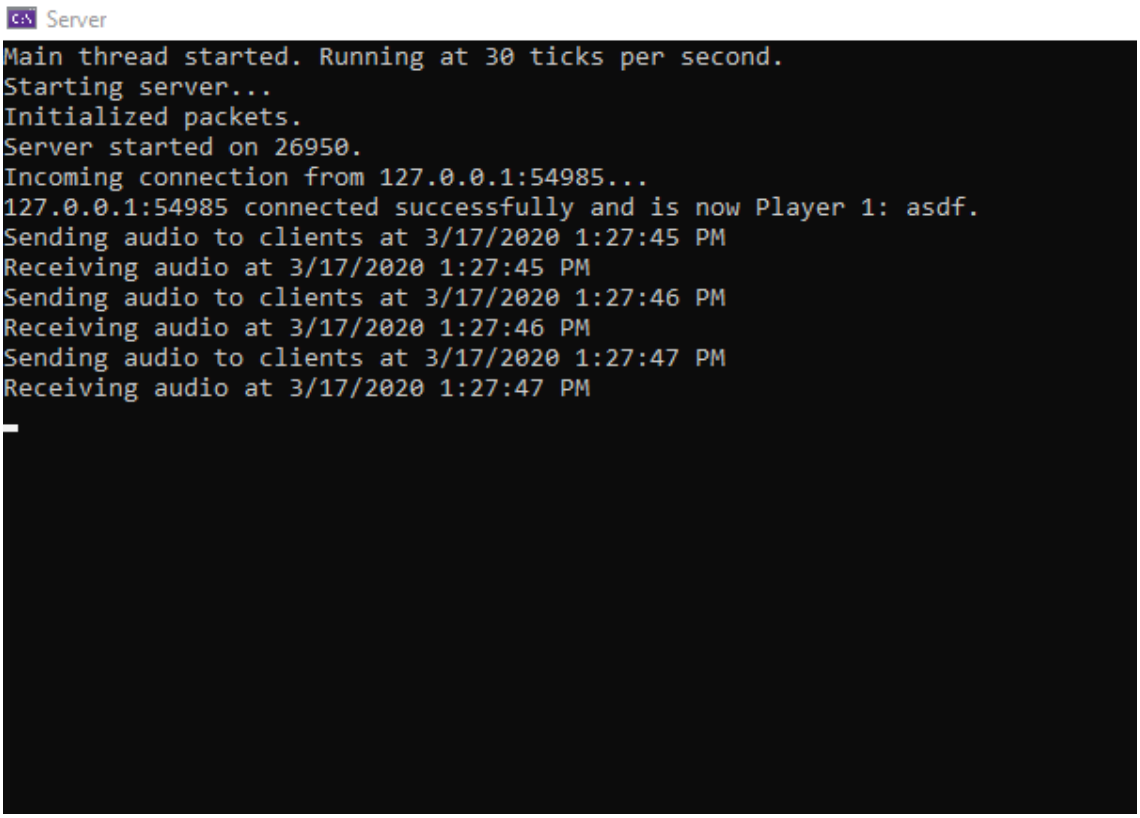

Fig. 8 Server sending and receiving audio