Augmented and Virtual Reality Telecommunication Tool

Nicholas Wilson

Follow this and additional works at: https://lib.dr.iastate.edu/creativecomponents

Part of the Communication Technology and New Media Commons, Digital Communications and Networking Commons, Graphics and Human Computer Interfaces Commons, and the Software Engineering Commons

Recommended Citation
https://lib.dr.iastate.edu/creativecomponents/561

This Creative Component is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Creative Components by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
Augmented and Virtual Reality Telecommunication Tool

Abstract

The objective of this creative component is to design and develop a mixed reality telecommunication application. Video conferencing platforms have seen an incredible increase in popularity over the past few years and have become an essential tool in most modern industries. The importance of these platforms as well as their limitations provide an opportunity for advancement through the integration of mixed reality. With this goal in mind, I was tasked with developing a Unity application that must; connect multiple players through a network connection, support both VR and AR players, and allow players to exist in the same relative 3D space.

Final Submission Requirements

- Application Design Document
  - Application overview and intended use case
  - Description of important project features
  - Highlighting of significant project challenges
- Demonstration Screenshots
  - Screenshots from the application
- Unity Project
  - Zip file of complete Unity project file
Application Design Document

Project Overview

Over the past 30 years, mixed reality communication and collaboration has been a popular topic of Human Computer Interaction research and development (Ens et al., 2019). In support of this research, a wide variety of applications have been developed. Most of these applications support connecting only VR devices or only AR devices, while a few of them have supported both. Supporting both AR and VR devices proposes the unique challenge of combining the physical and virtual world in a selective way. Too much virtual content and the AR user would be overwhelmed, but too little virtual content and the VR user would be underwhelmed. A common solution to this challenge is to capture the AR user’s physical environment and virtually recreate it for the VR user. This approach can vary based on the technique used for capturing the physical environment. Some applications have live streamed 360 camera footage while others have generated 3D point clouds using more advanced cameras (Piumsomboon et al., 2019). Due to the disadvantages of these approaches and the fact that none of the applications were open source, I decided to design a new approach to this challenge.

The goal of the application is to have the AR user virtually recreate their physical environment, so the VR user(s) can connect and occupy that same 3D space. Both users will occupy the AR user’s physical environment with the VR user(s) experiencing it completely virtually through a desktop HMD like the HTC Vive and the AR user experiencing it both physically and virtually through the Microsoft HoloLens (1 and 2).

The application developed for this creative component connects VR and AR users through the following set of actions.

- **AR User**
  1. Takes a 360 photo of their environment
  2. Opens the application on the HoloLens
  3. Logs the location and rotation of the camera where the 360 photo was taken
  4. Connects to the network

- **VR User**
  5. Connects to the network

Once these steps have been completed, the VR user is placed in the 3D environment in the location where the 360 photo was taken and the users can see the following:

- **AR User**
  - Their own physical environment
  - An avatar representation of the VR user in the location where the 360 photo was taken

- **VR User**
  - The 360 photo of the AR user’s environment
  - An avatar representation of the AR user
The following renders were made to demonstrate the concept of the application.

Because the AR user logged the location where the 360 photo was taken from, the application is able to track the AR user’s location relative to location of the 360 photo. Furthermore, because the VR user is placed in that location, the AR user’s avatar is shown in the correct location relative to the 360 photo. As a result, as the AR user moves around their environment, the avatar appears to move around the 360 photo of that environment in the same way.

While this application could prove to be useful in various scenarios, it was developed to be used as a remote training tool. In almost every industry, there are specific skills or processes that are too complicated to teach using videos or photos. An example of this in the world of mechanical engineering is all of the skills covered in the labs that students are required to take as a part of some of the classes. The skills covered in these labs are ones that could not be properly taught during lecture using slides or demonstrations. Similar scenarios often occur in the business world, but providing face to face training often means having everybody travel to one location. The goal of this application is to be used in this scenario. The AR user would be the Instructor.
and the VR user would be the Apprentice. By capitalizing on the immersive benefits of mixed reality, this application aims to provide the Apprentice with a higher quality training than one they would receive over a video teleconferencing platform, while still eliminating the need for travel.

Another scenario I see an application like this being used is in the various situations that a person would want multiple users to occupy their own environment. A specific example of this that I envision is when a company would like to host a conference meeting with multiple individuals that are located at various locations around the world. In a situation like this, the current solution would be to either have all the intended participants travel to one location, or to host a conference call via the phone or an application like skype or zoom. An application like this would allow the hosting user to take 360 photos in each of the chairs in their conference room and have each of the attendees share the 3D space with avatars representing each one of them. They could look around and see each of the other people in the meeting, talk to them, and even point at things. Ideally, it would give each member the sense of presence that exists in a real meeting, without having to deal with the hassle of traveling.
Project Features

There are three primary features of this application. First, this application supports the use of desktop virtual reality headsets like the HTC Vive and the Oculus Rift. This feature is made possible by incorporating the Virtual Reality Tool Kit (VRTK) Unity package. VRTK is a powerful and widely used open source SDK that provides a wide variety of tools that help with developing Virtual Reality applications with Unity. By incorporating the package in a Unity project, the application can be built to easily run on things like SteamVR, which is what was used to test for the VR user. For this project I used VRTK v3.3.0 with SteamVR v1.2.3.

For the second primary feature of supporting the Microsoft HoloLens, a similar toolkit was used. The newly restructured Mixed Reality Tool Kit (MRTK) is another powerful SDK that is primarily used for developing applications for the HoloLens. MRTK was formed by completely revamping the old Holo Tool Kit in preparation for the release of the HoloLens 2, which is currently being slowly released. Microsoft developed the MRTK so that Unity developers can relatively easily build custom HoloLens applications and show off the incredibly powerful uses for the augmented reality headset. For this project I used MRTK v2.1.0.

The third primary feature of this application is that multiple players can connect and interact over a network connection. To support this, the Photon PUN 2 Unity asset package was incorporated. The PUN 2 package is a free asset package on the Unity Asset Store that allows developers to build network multiplayer projects after making an account with Photon. The package is widely used and heavily supported and provides developers with plenty of great tools to help get over the heavy lifting of connecting via network. For this project, I used Photon PUN 2 v2.16.

When used by themselves, the three packages mentioned above are extremely powerful and relatively easy to use when building simple applications with low levels of interactions. Since they are all heavily supported, finding example projects and how-to guides online is simple and doesn’t take much work. The difficult part of building this application was trying to incorporate all three. There are not many applications that do this and the few that do are not open source so how-to guides really don’t exist. For this reason, I encountered a lot of challenges while building the application because I needed to really understand the toolkits and use some more advanced tools that they provide. The more difficult ones are mentioned in the next section.
Development Challenges

The most difficult development challenge that I encountered was trying to use the Photon PUN package to connect the users over network. I have never used Photons packages in general, so using it for an application like this forced me to really understand how the package works. There were plenty of example applications for connecting multiple users of the same platform but there were zero examples for how to connect an AR user with VR users I imported and ran a lot of example projects that connected multiple VR, AR, or computer users to try and find the differences in the projects and how the application responded to changes in the code. It was a long process because the MRTK and VRTK assets can be completely broken with just small changes so while some solutions worked for the AR user, the VR user’s avatar would disappear or show no movement. In the end, going through the process of understanding how Photon worked by testing a bunch of different solutions and blending them all together was the key for developing the application.

Another challenge that I encountered throughout the length of the project was keeping the performance restrictions of the HoloLens in mind. The HoloLens is a very powerful device, but in terms of processing power, it severely lacks compared to the well-built computers that the VR headsets run on. This heavily limited the application design because with too many complicated objects in the scene or too many processes occurring in the background, the HoloLens would perform so poorly that the application would essentially be worthless. Throughout the conceptualizing and development of this application, the performance limitations of the HoloLens forced me to develop alternative ideas and solutions many of times.
Future Work

While the bulk of the development for this application is completed, there are still some improvements I plan on making. The first of these is incorporating a live stream of the AR players “first person view”. For the Instructor-Apprentice virtual training use case, it important for the Apprentice to get a live view of how the Instructor is interacting with their environment. By watching the Instructor’s avatar move around the 360 photo, the Apprentice gets the sense of being in the Instructor’s physical environment, but they cannot see what exactly the Instructor is actually doing. My solution to this problem was to attach a camera to the Instructor’s HoloLens and stream the video from that camera to the Apprentice.

I have already incorporated this feature into the application but included this in future work because I was not able to test or demo it. While developing this feature, the first challenge I encountered was how to capture the Instructor’s view. For this, I used some 3D printed parts to mount a webcam on top of the HoloLens. This setup is shown below.

![Camera Setup](image)

The next challenge was being able to live stream the video from that webcam to the Apprentice. While my initial thought was to use Photon for this, the restrictions of their networking forced me to use another package, FM Network. This package is one I found on the Unity Asset Store and it comes with very little support or documentation. After spending some time working with it, I was able to get it working.
Now, with both players connected to the network, the Apprentice has the option of viewing the Instructor’s first person live stream in 2 different ways.

1. On a small display just above their controller:

2. On a large display in a theatre-like setting:
Application Screenshots

The majority of the following screenshots were taken from a demo of a previous version of the application. Unfortunately, due to the circumstances of Covid-19, I was not able to access the equipment needed to test the most recent form of the application.

Instructor

1. Takes a 360 photo of their environment

2. Opens the application on the HoloLens
3. Logs the location and rotation of the camera where the 360 photo was taken

4. Connects to the network
Apprentice

5. Starts VR application and connects to network
While both players are connected, they see the following views:

**Instructor**

![InstructorView](image1)

**Apprentice**

![ApprenticeView](image2)
References


HoloLens Materials


https://docs.microsoft.com/en-us/windows/mixed-reality/recommended-settings-for-unity


https://docs.microsoft.com/en-us/windows/mixed-reality/mrlearning-sharing(photon)-ch1

VRTK Materials

https://vrtoolkit.readme.io/

https://vrtoolkit.readme.io/docs/getting-started


https://github.com/quintesse/PlayoVR

Photon Example Project