Spring Term CS Capstone Final Handoff Report

May 27, 2021

Augmented Reality for Real-Time Strain Visualization

Prepared by

Team ARH

An-phong Luu Nguyen

Conner Rhea

Fransisco Javier Hernandez

Contents

1 Introduction

The project we worked on was to develop an application using the Microsoft HoloLens that can model the strain put on an object measured by strain gauges in real time. This involved the use of Mixed Reality, which is effectively an inbetween of the Augmented Reality and Virtual Reality technologies. The intent is to allow the user, assumedly a civil engineer, to have a visual reference to the stresses being put on an object by visualizing the feedback given by the gauges.

From our project description:

> Professor Bailey from CS is interested in 3D Augmented Reality. Professor Chris Higgins from Civil Engineering runs stress tests on various concrete and cross-laminated timber structures. He wants to be able to visualize the results from those tests by seeing graphics superimposed on top of the structure being tested. Your job will be to write a program or programs that use a Microsoft Hololens 1 to do that.

2 PROJECT DETAILS

2.1 Project Description

The project needs to be loaded or preloaded into a Microsoft Hololens II. Using the app, the user can view the playback of the test results provided to the team by client Professor Chris Higgins. Wearing the headset, the user is shown a virtual representation of the test site at the Hinsdale Wave Basin Lab. The system shows a green “handle” and 8 “strain gauges” set in a line with the handle in the center.

As the user uses the project, each individual strain gauge representation will visually show the strain level felt by their respective real-life gauge. As the level of strain felt by a gauge increases, the color of the orb will shift from blue to red. In addition, orb’s whose representative strain gauge undergoes increased strain will start to lower in the virtual environment and vice versa. The behavior of the strain gauges is intended to make it easier for the user to notice patterns or anomalies within the testing environment.

2.2 Project Tools

2.2.1 Development Engine: Unity

Development for the project was done in the Unity Engine. For some team members, this was their first experience with using the Unity Engine, so part of the development was learning the workflow and organization of the Unity editor

2.2.2 Libraries: Microsoft Mixed Reality Toolkit (MRTK)

This is the library that needs to be installed in a Unity project to do Hololens AR development. It includes several scripts that allow the user to interact with virtual objects.

2.2.3 Language: C#

C# was the primary language used for the project, as scripting for Unity is done in C#. As with Unity, some team members had little experience with C# prior to the project. However, the learning curve was not too difficult as the language has many syntactic similarities to other languages like C++ and C.

2.2.4 Hardware: Microsoft Hololens II

Development for the project was done in order to create an app deployable to the Hololens II. The team ended up allocating a substantial amount of time to researching the prerequisites and dependencies needed to prepare their systems for Hololens development. This preliminary step took a substantial amount of development time, with some team members eventually able to download the Hololens II emulator while one team member ended up halting major development until they could physically access a Hololens. Because of this, the team was left with much less development time than they had planned for.

3 System Overview

3.1 Strain Gauge Orbs

The virtual strain gauge is the simplest and most independent portion of the system as each gauge in the virtual environment operates on its own. Overall, all the strain gauge object does is idle until it receives a float from an external source. Once it receives this signal, the gauge then independently displaces and recolors itself.

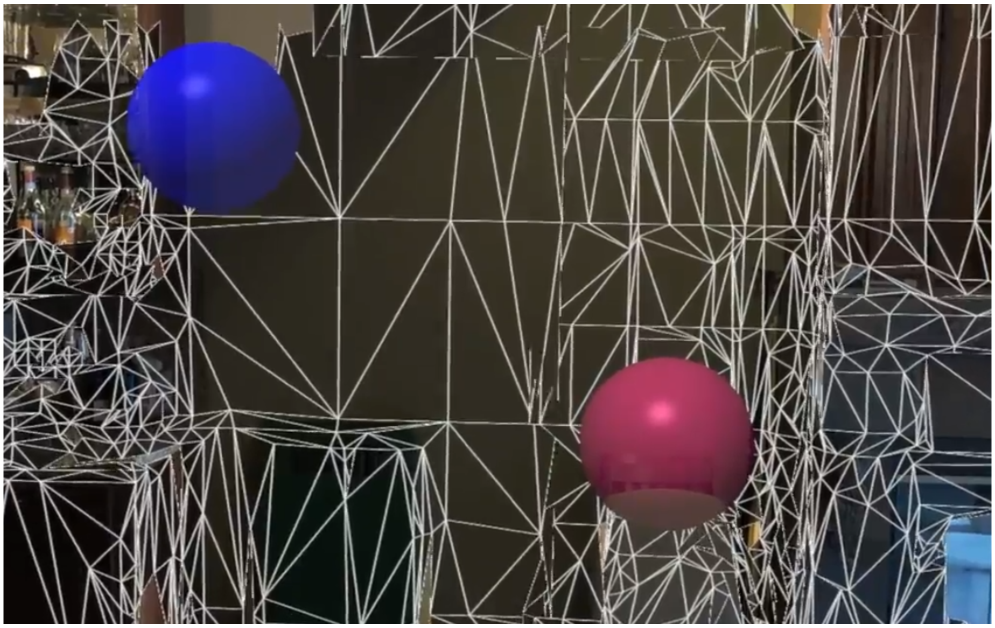

Figure [AD5]: Example of two strain gauges with differing strain values. The lower (higher strain) orb is also a more red shade than the higher (lower strain) orb.

3.2 Comma-Separated-Value File Parser

The Comma-Separated-Value File Parser or CSV Parser is actually broken up into two separate parts: the CSV Reader and CSV Signaler. One half of the system reads a CSV data file and records this data while the other half reads this data and signals the strain gauges.

The CSV Reader and CSV Signaler are put into the same component to improve system efficiency. As the CSV Reader writes to the same data structure that the CSV Signaler reads from, having this structure readily available to both components reduces how much inter-object communication the system needs.

3.2.1 CSV Reader

The CSV Reader takes in the .csv formatted data file that comes from the physical test. It reads in each line in the data file and creates a data structure to hold the relevant information including strain values and time of recording for each entry.

3.2.2 CSV Signaler

The second component of the CSV Parser, the CSV Signaler is in charge of reading the data structure populated by the CSV Reader and outputting this data to the individual strain gauges. The Signaler does this by constantly cross referencing the current time and time value of the data structure's entries.

This component is crucial to the project as it is in charge of signaling strain gauges with strain values. Without the CSV Signaler, strain gauges would simply remain idle.

3.3 Gauge Handle

This system appears as a green box placed in between the two groups of four strain gauges. By pinching this object, the user is able to move the entire array of strain gauges. This allows the user to reallocate the gauge array to a more desirable position and makes it so the user does not have to pre-plan loading the application to properly place the strain gauges correctly.

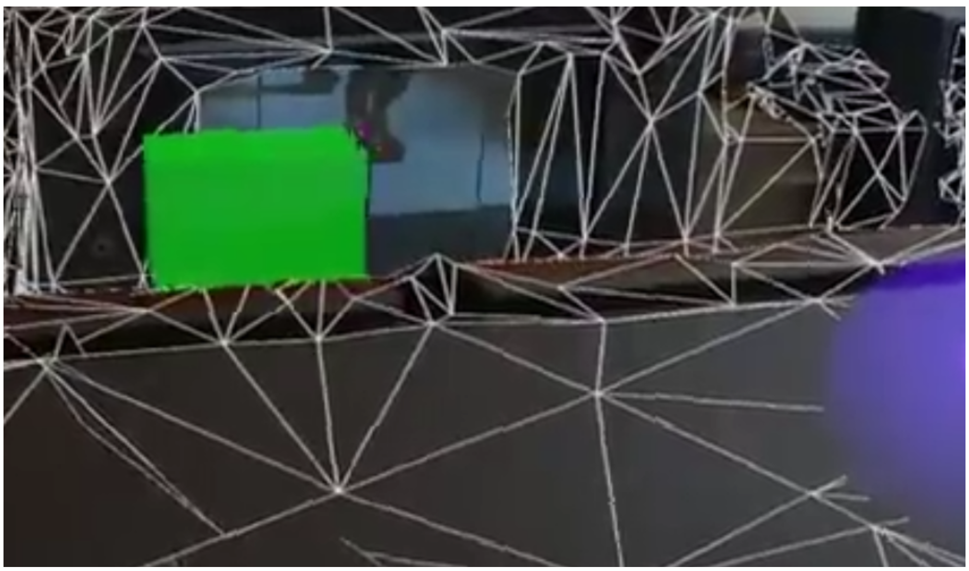

Figure [AD6]: The Gauge Handle and its position relative to the immediate rightmost strain orb.

While the Gauge Handle appears separate from the strain gauges, in Unity the object is actually the parent container for all eight strain gauges. With this relationship, displacement on the GH affects all strain gauges while displacement on any one strain gauge has no effect on the GH or any other gauges. So, when the user moves the GH to a new location, each gauge follows, but when each gauge moves itself due to a strain value, it has no effect on any object besides itself.

3.4 Hololens II

The Hololens II is the only physical component of the system. After placing the virtual system, the Hololens is in charge of managing the merging between the physical and virtual environment as well as detecting user movement and gestures.

Once the virtual system is placed, the Hololens' cameras and sensors are used to track the user's movement through the physical environment. Physical movements are then translated into digital movements, moving the camera placed in the virtual environment. This makes it so virtual objects appear “fixed” in physical space. Without this functionality, the system would be unable to fix the gauge array over a testing environment.

The Hololens also detects gestures. In the project, this includes the pinching gesture used to grab and move the Gauge Handle.

4 Project Usage

The following steps include instructions needed to properly compile and use the project on a Hololens II. Sections 5.1 to 5.3 include the actions needed to compile the project from the build folder without any source code while step 5.4 details how to relaunch the project on the Hololens and how to interact with the project.

4.1 Loading the project into Visual Studio

To start, the build folder needs to be loaded onto a desktop computer with Wi-Fi access. From this folder, the “Capstone MRTK.sln” file should be opened with Visual Studio. If necessary, update Visual Studio and ensure the following workloads and components have been installed:

- .NET desktop development
- Universal Windows Platform development
- Desktop development with C++
    - IntelliCode
    - C++ Device Connectivity
    - C++ (v142) Universal Windows Platform Tools
    - Graphics debugger and GPU profiler for DirectX Windows 10 SDK (10.0.18362.0)

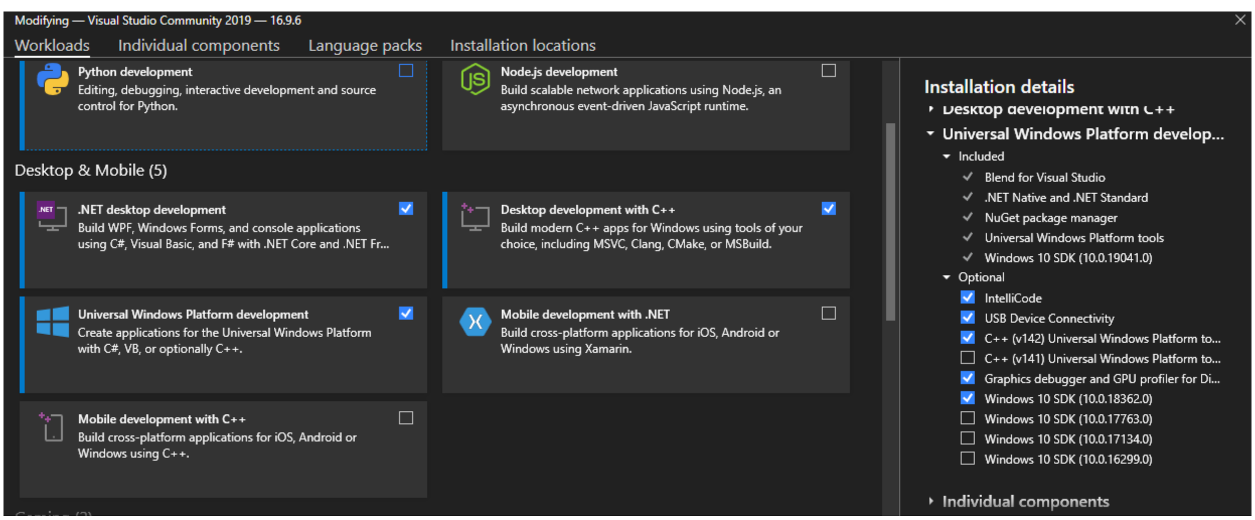

Figure [AD3]: Within the Visual Studio Updater, the shown workloads must be installed and for Universal Windows Platform Development, the shown features must also be included.

4.2 Deploying the project

In Visual Studio, ensure the following build settings are set as follows:

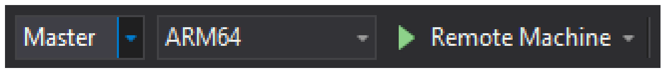

Figure [AD4]: Image of the deploy settings needed in order to properly load the project into a Hololens over Wi-Fi

Navigate to Project > Properties > Configuration Properties > Debugging. Ensure that the Authentication Type field contains “Universal (Unencrypted Protocol). In the Machine Name field, enter the ip address of the target Hololens II unit.

Close the Project Properties window and navigate to Build > Deploy. The project should start to build and prepare to deploy to the Hololens unit.

Turn on the Hololens unit and log in. After Visual Studio finishes deploying, the project should immediately begin to load on the Hololens.

4.3 Loading the project on the Hololens

If the project has already been loaded onto the Hololens unity, simply navigate to Apps and press the “Capstone MRTK" icon. The project should begin to load and playback should begin.

4.4 Using the project

When the project is loaded, ensure that there is ample space in front of the Hololens. The project will automatically load the strain gauge set roughly 1ft in front of the user. Any obstructions may make the project load the set into a physical object, making the gauge set unreachable or unviewable.

When the project loads, it will automatically start the playback of the project test demo. From there, the user can freely move around the environment without altering the position or orientation of the set of strain gauge orbs or the array handle.

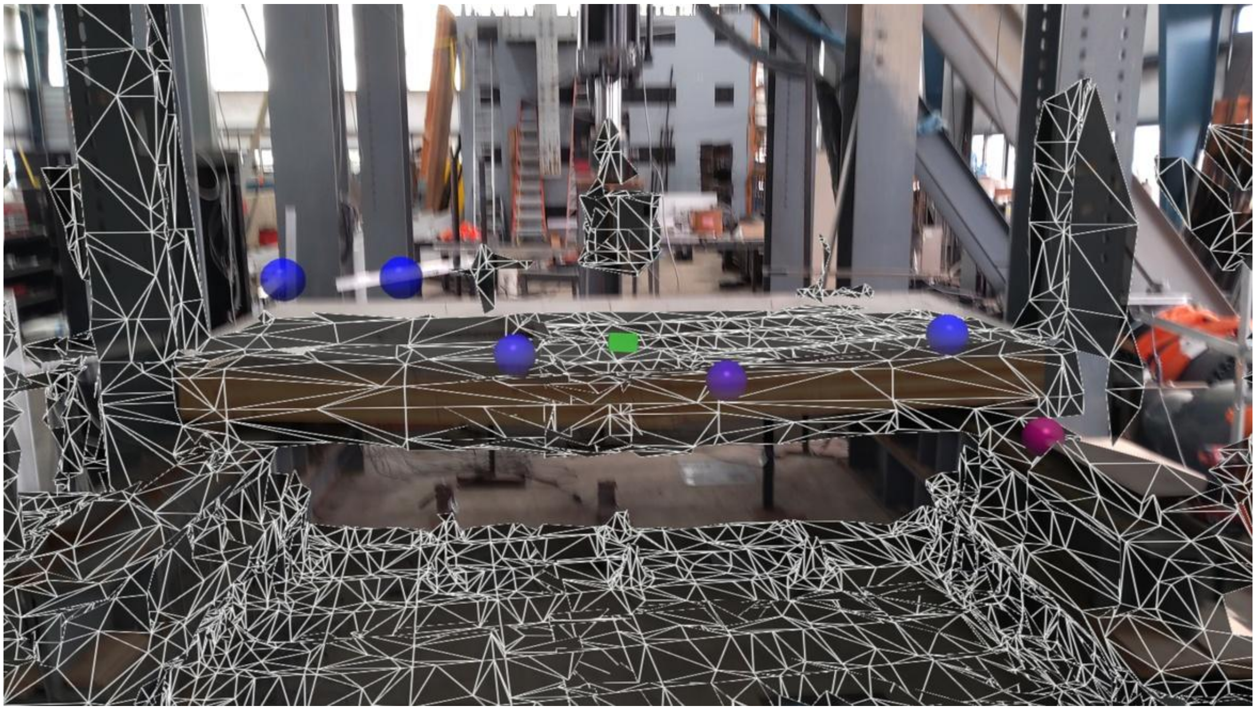

Figure [AD1]: The project in use at the Hinsdale Wave Basin center. The virtual system has been physically moved over the actual site where strain testing takes place.

After the gauges appear, the user can easily relocate the gauge set. By holding their hand out and pinching the green array handle with the thumb and index finger, the user can “grab hold” of the gauge array and freely move the set around the environment. This means that it is not necessary for the user to preposition themselves to ensure the gauge set is placed correctly. After loading the set into an arbitrary location, the user can move it into a more desirable position.

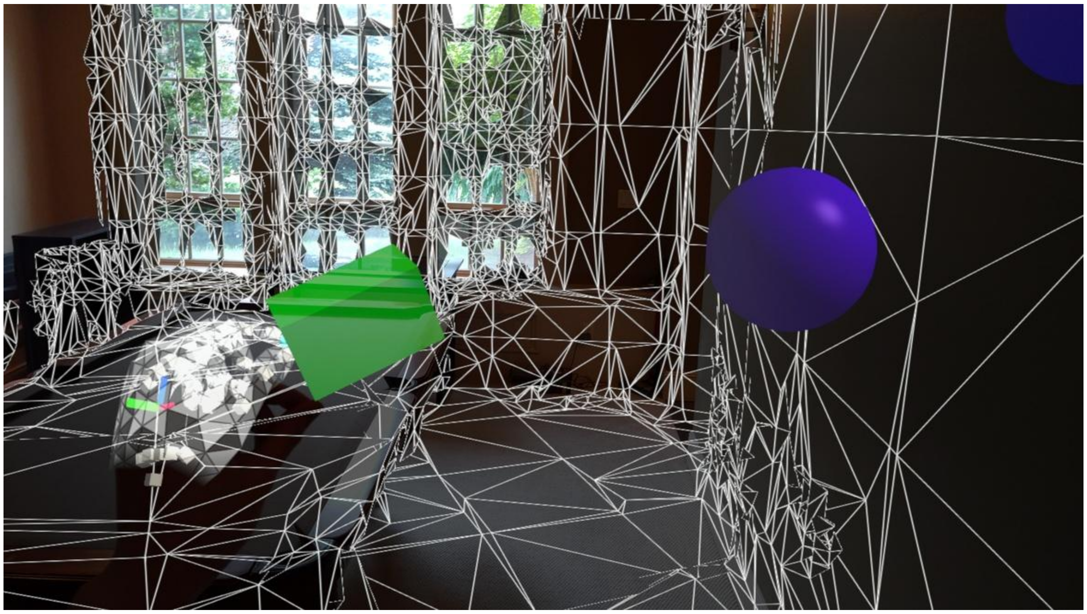

Figure [AD2]: Image of a user using a pinch gesture to hold onto the handle and reallocate the system. The Hololens II will highlight the user's hand to indicate that they are being properly tracked.

The Hololens roughly shares the same field of view as the user due to how the cameras are mounted onto the headset. Because of this, the following will apply

- When making a pinching gesture, if the user cannot see their fingers making the motion, the Hololens likely cannot either
- When moving an object out of the user's view, the Hololens will lose view as well, meaning the virtual hand will let go of the object even if the user sustains the pinching motion

5 Project Deviations

5.1 Primary Setbacks

Over the course of development, the project has been scaled back, many of the features and functionalities initially planned for have been either stripped back, removed, or modified.

In order to develop for the Hololens II - a relatively new platform - and MRTK, the Hololens' exclusive library, the team had to reallocate a substantial amount of time meant for development to further research. As a result, many features had to be simplified or removed.

5.2 Modified Features

5.2.1 Digital Tracking System

The initial plan to track the strain gauges onto the physical environment was to utilize the Hololens' camera to automatically track the system. This would have been done using either mo-cap style tracking with tracking markers placed in the physical environment or by saving the physical layout at the Hinsdale lab so that the system can identify where it needs to place strain gauges.

In the final product, tracking is instead done manually by the user. Instead of having the system automatically track where gauges need to be, the system arbitrarily places the gauges and the users then use the green “handle” to relocate them.

While time did play a part, the main reason for this change was the availability of tools and limits to how much the team could affect the physical testing procedure. In order to have the Hololens map and know where to put gauges, the team needed Microsoft Azure's Spatial Anchors, which they could not access. The team also decided against using tracking markers as they could be a safety hazard during physical tests.

While not the initial plan, allowing the user to physically move the gauges manually offers additional flexibility. By arbitrarily placing the gauge array and allowing users to place them, the project is now able to be used outside of the Hinsdale center. With more time, the team would have liked to implement a system that utilizes both manual and automated placement. This way, the project is quick to set up at the Wave Basin, but is still able to operate at other locations.

5.3 Removed Features

5.3.1 Realtime Feedback

The initial plan for the project intended for the system to be able to read input directly from strain gauges and output that data in realtime. With this functionality, the user would be able to view test results as they happen during a physical test.

During development, the team shifted from realtime feedback to post-test feedback. This means the removal of several features including tracking of strain gauges and any of the other features that were necessary with realtime feedback. Rather than outputting results as they are recorded by the test, the system is instead fed the complete data file after the test. In this way, the user is only able to view the results of the test with the system after the test.

The main reason for this change was time constraints. A large portion of development time was needed to simply implement the basic functionality of the project. To build onto the current project to implement this feature, the team would have needed a few more weeks to implement the I/O system that included the data recording system at the Hinsdale lab.

5.3.2 Idle Mode

One of the features included in the initial plan was to include an additional system state where the Hololens stops outputting visuals to the user, allowing them to navigate the physical environment without distractions.

Currently, the only state in the system is “playback mode”. There is no way for the user to hide the digital constructs from their Hololens view.

Time was also the main factor leading to the removal of this function. The team did not have enough time to develop the gesture handlers needed to implement parts of the system to allow for more than moving the “handle”.

5.3.3 Swapping System States

As the original system included three different states. One of the intended features were to be able to swap between them freely between the following:

Synchronization, Idle, Heat map

Since we did not implement Strain Gauge Data Transmission in real time, and many other features due to time constraints.We instead used processed data from an excel sheet instead.

6 Project Timeline

Below is the list of research and development tasks done by each team member throughout the Winter and Spring terms. As fall term primarily consisted of planning and research, tasks for that term were left out.

6.1 Anphong

Winter Week 1:	**Research:** - Learned more about Mixed Reality Toolkit(MRTK) - Learned more about general Unity/C# development
Week 2:	**Development:** - Created a data format for “mock data” so other portions of the project can be simulated
Week 3:	**Development:** - Created mockData.c to create/update mockData files
Week 4:	**Research:** - C# file IO - C# timing functionality API
Week 5:	**Research:**

	- Hololens Emulator alternatives **Development:** - Created the data format for the systems “config” files.
Week 6:	**Development:** - Created configReader.c to read config files and create gauges - Created configWriter.c to update config files to provide flexibility
Week 7:	**Research:** - Unity triggers, buttons, and input fields **Development:** - Created baseline template for strain gauge script “guageData.c”. - Created csvReaderMock.c. - Connected currently existing project components to run as a single system
Week 8:	**Research:** - Unity scene manager API - Unity scene functionality **Development:** - Set up framework for multiple Unity scenes that will be used in the project - Created sceneManageer.c to swap between scenes
Week 9:	**Development:** - Set up the distinct scenes that will be needed for the project
Week 10:	**Development:** - Created csvReaderTestResults.c to read from provided example data
Spring Week 1:	**Development:** - Ported project beta to Hololens II
Week 2:	**Development:** - Created Gauge Handle - Created Gauge Array
Week 3:	**Development:** - Modified Strain Gauge to work with Hololens II
Week 4:	
Week 5:	**Development:** - Made changes to project to allow for Hololens II deployment
Week 6:	
Week 7:	**Development:** - Created CSV Signaler - Modified CSV Reader to work with Hololens - Implemented zeroing/scaling in CSV Reader
Week 8:	**Misc:** - Went to Hinsdale Wave Basin Lab to demo project to client and record expo materials

6.2 Fransisco

Winter Week 1:	**Research:** - Research on MRTK and Unity
Week 2:	**Development:** - Configuring Unity in order to develop on it. **Research:** - Research on MRTK and Unity
Week 3:	**Research:** - Research on MRTK and Unity and how to configure them. - Research on how to show displacement - Mock data to create displacement
Week 4:	**Research:** - Look into how we will manage our csv changes.
Week 5:	**Research:** - Figuring ways out to emulate the HoloLens efficiently on my system - Further configuration issues being looked into - Researching how to reverse displacement(failed) or calculate displacement - Research the data we received
Week 6:	**Research:** - Research on data we received - Help find alternatives for emulating the AR teammates
Week 7:	**Research:** - Research on data we received
Week 8:	**Research:** - Still working on how to read the excel file, but still waiting on a meeting with our sponsor.
Week 9:	**Research:** - Discussed with the team about how we will create the new solution.
Week 10:	
Spring Week 1:	**Research:** - Researching Issues we had with Unity last term.
Week 2:	**Research:** - Researched how to create spatial anchors.
Week 3:	**Research:** - Continued research on spatial anchors and saving mapped locations.
Week 4:	**Research:** - Researched about MRTK.
Week 5:	**Research:** - Method to effectively emulate HoloLens on lower end PCs.

Week 6:	
Week 7:	
Week 8:	- Missed the Hinsdale Wave Lab Demo with Dr. Higgins, Mike Bailey and Anphong.

6.3 Conner

Week 1:	**Research:** ● Researched Mixed Reality Toolkit (MRTK) ● Researched C# development environment.
Week 2:	**Development:** ● Begun development on original csvreader code
Week 3:	**Research:** ● Research on .csv format and reading it in C#
Week 4:	**Development:** ● Csvreader basic concept finished for code review
Week 5:	**Research:** ● Methods to emulate the HoloLens ● Cost of Windows 10 Pro Upgrade ● OSU Azure Server for WIndows 10 Education Edition
Week 6:	**Research:** ● Research attempt to discover how to read in the new data file format
Week 7:	**Research:** ● Excel Document Reader initial foray **Development:** ● Restarted attempt to read .xlsx file for project
Week 8:	**Development:** ● C# pathing details handled
Week 9:	**Research:** ● Handling C# DataSet Class.
Week 10:	
Spring Week 1:	**Development:** ● Begun prep for migrating to Windows 1`0 Education for Hyper-V
Week 2:	**Development:** ● Installed Windows 10 Education edition and Hyper-V ● Updated Unity in preparation to move over the Excel Reader Code. **Research:** ● C# and Unity package compatibility and csc.rsp files
Week 3:	**Development:**

	• Attempted to make Excel Reader code compatible with the Unity project space
Week 4:	**Development:** • HoloLens Emulator installed for testing
Week 5:	**Development:** • HoloLens emulator compatibility test with excel reader code and Unity debugging **Research:** • Unity Streamreader and handling external files **Misc:** • Lost time due bad reaction to Covid Vaccine
Week 6:	**Development:** • Shifted away from Excel to CSVReader do to time constraints and issues with HoloLens
Week 7:	**Development:** • Completed CSVReader code and passed it off to Anphong for implementation
Week 8:	**Misc:** • Hinsdale Wave Lab Demo with Dr. Higgins, Mike Bailey and Anphong. • Created materials for Virtual Expo

7 Conclusion

Though we did not get to have a fully finished project due to the multiple constraints we faced of learning the new technologies, covid, and other issues. The team produced what both the client and project sponsor saw as a good proof of concept. Though we do regret we did not get to finish the project in its entirety we are happy that both the project sponsor and client are satisfied with what we have created thus far. One of the things we greatly appreciate though is the chance to use and work with the HoloLens and MRTK technology.