Final Year Project

Extension of MAAP Annotate

Finn O'Neill

Student ID: 17367986

A thesis submitted in part fulfilment of the degree of

BSc. (Hons.) in Computer Science

Supervisor: Anthony Ventresque

UCD School of Computer Science
University College Dublin

August 25, 2021

Table of Contents

Project Outline

Interim Report Chapters

These chapters were included in the interim report and some alterations have been made to account for changes to the project.

- Chapter 2: Project Specification
- Chapter 3: Related works and Ideas

This section was originally included in the interim report, but as it is not relevant to the final report it has been moved to the appendix.

- Chapter: Project Workplan

Final Report Chapters

- Chapter 1: Introduction
- Chapter 4: Data Considerations
- Chapter 5: Outline of Approach
- Chapter 6: Conclusion and Future Work
- Chapter 7: Acknowledgments

Chapter 1: **Introduction**

Megalithic art comprises of intricate stone carvings made by Europe's first farmers. They provide us with an insight into the lifestyle and beliefs of our ancestors and thus are an exciting area of study for Archaeologists. MAAP Annotate is a Mixed Reality annotation tool from the Megalithic Art Analysis Project (MAAP). It provides an innovative method of interacting with megalithic art, combining cross-disciplinary research in digital heritage, 3D scanning and imaging, and augmented reality.[1] This project seeks to expand MAAP Annotate's current HoloLens program.

Given the existing MAAP Annotate project developed in 2017 by Johanna Barbier, et al.[1], the project plans to expand upon their work based on feedback from their test group and from the future work section discussed in their 2017 paper. The feedback suggests a number of potential improvements to existing features which seem to primarily focus on the user interface and the interactive tools - namely the drawing tool and lamp tool. The future work section discusses improvements and brings up some ideas for potential new features, such as the viewing of multiple stones.

This project hopes to experiment with and test the limitations of MAAP Annotate. It will also seek to develop new features for MAAP annotation, primarily: a feature to view multiple stones at once, and collaborative annotation to allow for more than one user to annotate a stone.The previous MAAP annotate project was developed for the Microsoft HoloLens and, as such, this project will also be developed for the HoloLens. Unity with the HoloToolkit add-on and Visual Studio Code will be used for the development. The Unity scripts will be written in C#. Other software such as Holographic Remoting Player will be used for testing and development. Testing and deployment will be done on a physical HoloLens. These tools are seen as the industry standard for HoloLens development and are recommended by Microsoft themselves.[2] I will also be using the existing code that was previously developed for the MAAP Annotate project.[3]

Chapter 2: Project Specification

2.1 Core Requirements

- Expansion of existing interaction tools.
- ~~Improvement of the user interface based on test group feedback.~~
- Performance testing of MAAP Annotate.
- Development of multi-stone viewing feature.
- Implementation of collaborative annotation feature.

2.2 Discretionary Requirements

- Updating of project code to be compatible with the most recent Unity release.

2.3 Advance Requirements

- ~~Testing of new and improved features by a test group*.~~
- Video demonstration of new features.
- Deployment of MAAP Annotate program on Oculus Quest.

*It is understood that given the ongoing pandemic there may be some limitations when organising a test group, hence its placement in advance requirements.

Chapter 3: Related Work and Ideas

3.1 MAAP Annotate

The MAAP Annotate[1] paper discusses the original implementation of the MAAP project. The early sections of the paper showcase the developed features of MAAP Annotate and gives the reader an introduction on how to use the program. Section 4 of the paper details an evaluation of the program and section 5 discusses the potential future work of the project. These sections are of notable interest as they will guide this project in the direction its development will take.

To evaluate the program a test group of archaeologists was gathered from University College Dublin. They were given an introduction on how to use the HoloLens and the MAAP program. They were then let explore the program on their own. Their feedback from the experience was gathered by a questionnaire that made use of the System Usability Scale[4]. Examining the results, the overall response to the program was very positive and it averaged a score of 80.75 out of a possible 100. The participants also offered some suggestions regarding the usability of the program.

In the evaluation, 60% of the participants rated the drawing tool at a three out of five. The drawing tool allows users to draw an annotation onto the stone. It is modelled by casting a ray through the wearer's head from a fixed point behind it to a virtual representation of their hand that is offset from their own hand. The feedback suggested that this tool is somewhat difficult to use and could use some tweaks. This project would seek to make two additions to the drawing tool that should improve its usability. Firstly the addition of a size slider that would allow the user to select different width sizes for their stroke. This would function similar to the Adobe Photoshop brush size tool [5] but would be simpler in its functionality.

The second improvement would be to add sensitivity control. With this, users could alter the level of sensitivity of the drawing tool. They could do this by varying the distance of their hand from the HoloLens. For faster, less precise movement, the user would hold their hand further from the HoloLens, but for slower, more precise control they would hold it closer. This idea would require some testing but may result in more intuitive drawing tool control. If testing reveals that this is an over-complicated solution then a similar result could be achieved with a slider for increasing and decreasing the sensitivity. When implemented, these additions would allow users to have greater

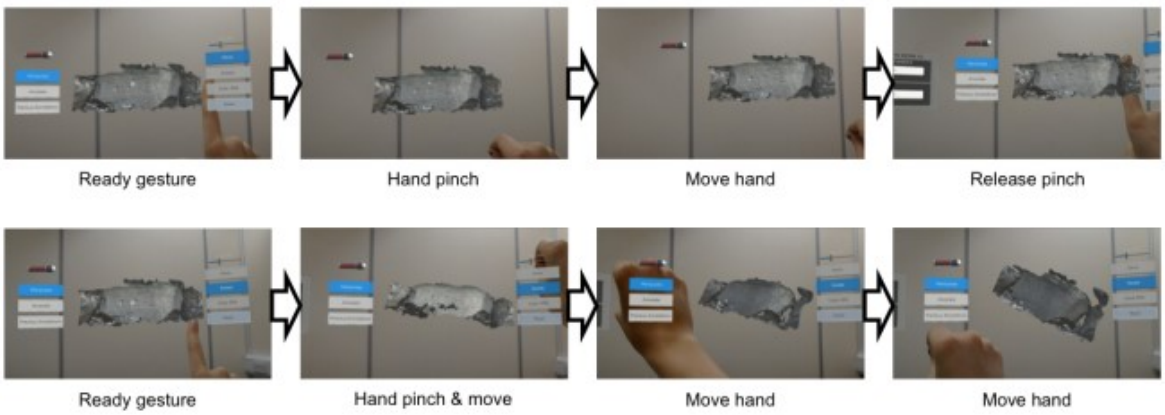


Figure 3.1: Images demonstrating gesture control in MAAP Annotate. Source: [1]

control over the drawing tool to ensure their annotations are clear and correct.

The Lamp Tool, which allows the user to shine a virtual light onto the stone, also received some feedback. The first suggestion was to increase the brightness of the lamp to improve the contrast it provides. A simple resolution that this project will implement is the addition of a slider that allows for brightness control. The second suggestion was to include more than one lamp so that more light sources can be shone on the stone. This will require experimentation within MAAP Annotate as processing the lighting effects of multiple sources may be an issue for the HoloLens' limited memory as is discussed in the MAAP Annotate Paper.

The last piece of feedback that this project will seek to address is given for the Rectangle Selection tool. The tester suggested having more shapes available for selection e.g. circle, triangle, etc. While this is possible, it would be interesting to explore the possibility of adding a feature similar to Adobe Photoshop's polygonal lasso[6]. This tool allows a user to place points between which straight vertices are drawn. The implementation of a similar tool in MAAP Annotate would give users the freedom to place the shape they wanted. If experimentation determines that this is not a viable option, then a selection of shapes will be added instead.

The discussion section of the MAAP paper breaks down the results of the test groups study and discusses the possible future works that could be developed for the project. The idea of viewing multiple stones at the same time was brought up by a participant in the test group. They had suggested the idea of a carousel that would be used to view a set of stones in a series. However, as discussed in the paper, this is limited by the HoloLens' on-board memory. In place of the carousel two or more stones could be shown side by side. This would allow for comparison of the stones or viewing of the same stone from different angles at the same time while also working within the memory constraints.

3.2 User Interface Design for AR

Given the feedback from the MAAP Annotate test group, one area this project will focus on will be an improvement of the user interface. Designing effective user interfaces for AR platforms presents some unique challenges. In the case of the HoloLens, the estimated FOV of 30° by 17.5° [7] can make it difficult to display all of the relevant information to the user at once. This can result in users missing content as it is outside their field of view[8] if they have been given no indication of its existence. While this could be solved with an implementation of a tutorial it can also be mitigated or avoided with intuitive UI design.

In the paper "Evaluating the Microsoft HoloLens through an augmented reality assembly application"[9], Evans, et al. discusses that while there is limited research on the topic of UI for AR platforms, we can use lessons learned from designing virtual reality interfaces to help develop more user-friendly AR interfaces. Their paper highlights some important considerations that have to be made. Due to the 3D nature of an AR platform, UI elements have to be positioned a correct distance from the user to allow for comfortable interaction. Positioning elements at 1 meter from the subject is understood to be the best distance for users to interact with and manipulate elements[8]. Anything under 0.5 meters is considered an uncomfortable placement[9] and is too close for effective interaction. It will be important to keep these constraints in mind when improving the UI of MAAP Annotate so as to ensure users can comfortably access all elements of the UI.

The limitation of having to place elements at distance from the user presents challenges in making sure text is clear and legible at such distances to reduce the effects of eye strain. Evans [9] found that a font size of 20pt and the use of a sans serif font allowed for a clear text that rendered well

on screen. The colour of the text and its background can also have an impact on readability. In 2004 Hall and Hanna found that where readability was concerned, black on white text or a closely related combination was the best option.[10] Their research also looked at aesthetic factors and found that chromatic colours were considered to be the most visually pleasing to users. Given that MAAP Annotate is an educational tool it is best to favour readability and thus the black text on a white background. However, due to the holographic nature of AR elements, further aspects must be taken into account. White can appear very bright on adaptive displays and darker colours can appear transparent.[11] Due to the bright nature of the colour white on an adaptive display, it makes for very clear and legible text. For the best results, a darker backplate should be placed behind the text to ensure it can be differentiated from the user's environment.[12] MAAP Annotate currently uses a black on light grey design [1] so a revision of the colour scheme could be considered in the UI redesign.

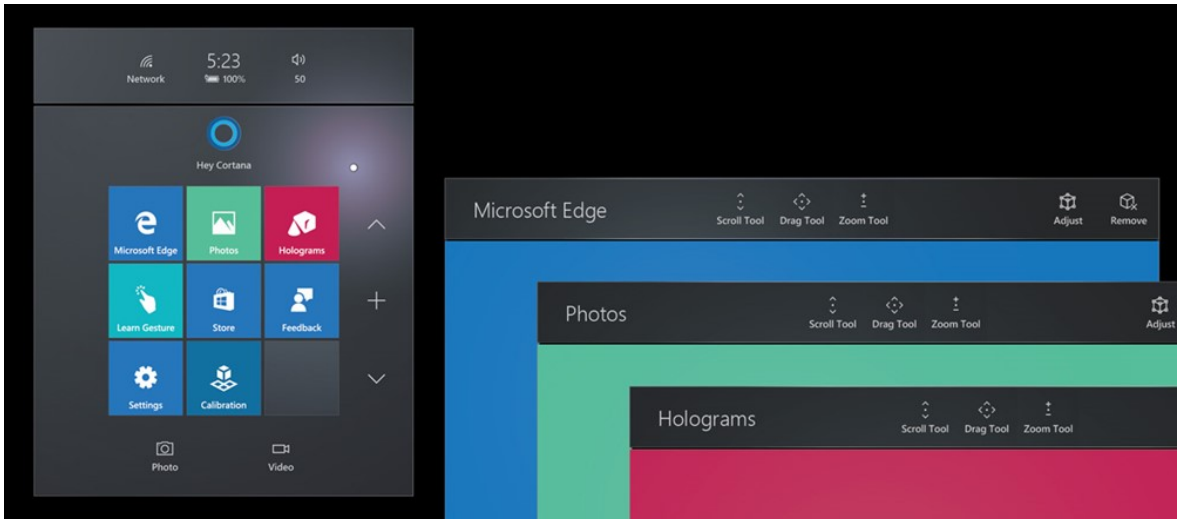


Figure 3.2: Depiction of white text on dark background along with chromatic colours Source: [12]

"Feedback is a way of assuring the user that an action has been performed" [9], providing a user with feedback as a result of their actions is an important part of a UI's design. Feedback can be given in a number of forms but in the HoloLens' case, visual feedback and audio feedback are the primary options. The HoloLens itself provides feedback to the user upon use of a recognised gesture. For example, the air tap gesture results in a cursor animation and a click sound being played. MAAP Annotate already has feedback in the form of highlighting the currently selected feature[13]. Audio feedback could be used to enhance the users experience further. The suggestion is to use audio feedback for events that are not initiated by the user.[14] This may make it applicable in the collaborative annotation feature where there will be multiple users interacting with the stone. A notification could play when another user adds text or annotation to the stone.

3.3 Collaborative AR

In 2003 Hannes Kaufmann defined collaborative augmented reality as "multiple users ... access[ing] a shared space populated by virtual objects while remaining grounded in the real world."[15]. Applying this idea to MAAP Annotate would suggest multiple users being able to view and possibly annotate the virtual stones. There have been a few examples of other works that developed ideas and methods to achieve this.

A project in 2018[16] developed a proof of concept for HoloLens collaborative augmented reality in

the form of a disaster management application. It made use of spatial anchors[17] which in essence renders the object in the form of a coordinate system. When a HoloLens loads the application, the program will check the server to see if an anchor exists. If one does not then the current HoloLens would upload the coordinates of its object to the server. When a second HoloLens connects to the server, they will be sent this object as the anchor and will see the object in the same orientation with the same spatial points. In practice, this worked well and they successfully demonstrated the concept in their disaster management app by having multiple users view the same object. One issue the paper did discuss was how sub-grouping too many objects to an anchor could result in problems in spatial locations.

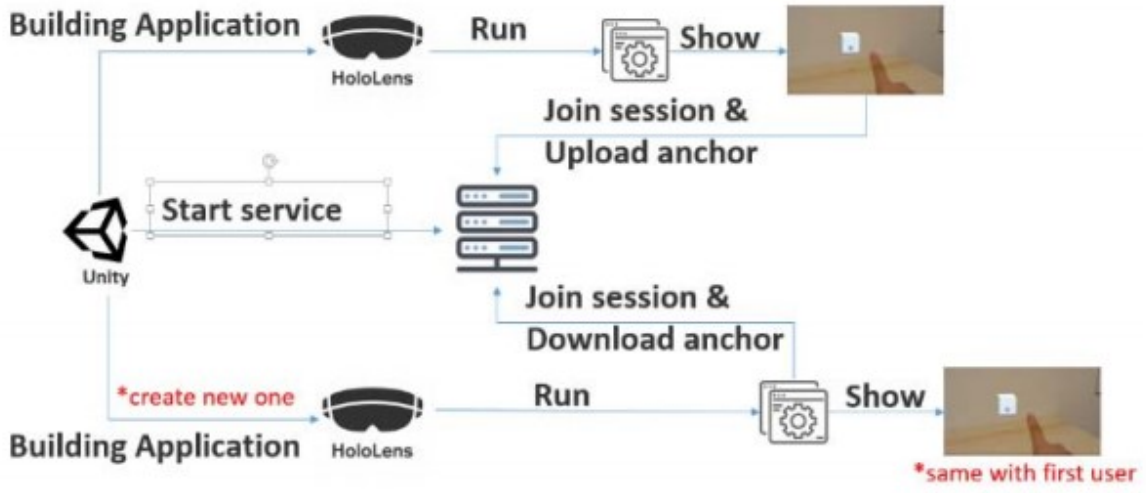


Figure 3.3: Diagram of the system used in Chisetthargan's et al. approach. Source: [16]

The approach discussed here could work very well with MAAP Annotate. The spatial anchor would allow for the positioning of the stone in a spatial location where multiple users could view it. One concern would be how applying annotations to the stone would affect the spatial locations and performance of the HoloLens. With two people annotating there may be issues such as the one discussed in the MAAP paper [1] where larger meshes would cause the application to crash. A possible solution would be to allow only one person to annotate and have the other user's as spectators. For now, this is merely speculative and further research will be needed here to ascertain how viable this approach is.

Considering the uniqueness of the HoloLens, it is fair to say that most people do not have access to one. Hence, collaborative annotation is made more difficult by lack of access. A paper by Henry Chen et al. in 2015 [18] outlines a method to solve this by using the program Skype in place of multiple headsets. They built an app on the Skype client that works by having the HoloLens user act as the explorer of the virtual space and having the other collaborators "piggyback" on the view that the HoloLens user sees. The Skype client users have tools to lock the view in place and add annotations to the environment. A variant of this solution could be applied for MAAP Annotate. This would allow for one user to use MAAP Annotate on a HoloLens and for collaborators to connect in to see their view and to provide annotations. This option may also be less taxing on the HoloLens itself.

Further research and experimentation will be done on collaborative AR during the next portion of the project to discern the best approach. However, from a preliminary point of view developing collaborative AR with spatial anchors would be the preferred method for this project as it could possibly allow all users to annotate in augmented reality.

3.4 Hands-on Research

Outside of the research of academic papers, hands-on background research was done with a HoloLens and an Oculus Quest. Through this research, it was discovered that the previous MAAP annotate project does not compile on some of the more recent versions of Unity. This is due to it using legacy code from previous versions of Unity. There are ways to update the project to a more recent version, however, through discussions with the project supervisor it was discerned that this issue may be beyond the scope of this project. The possibility of updating the code to a more recent version of Unity was added as a discretionary requirement as further research is required to discover the degree of work required to fix this issue.

3.5 Summary

To summarise the research outlined above, the goals this project aims to achieve for each of the selected features are as follows:

- ~~**Drawing tool** - Implementation of a brush size slider. Experimentation with a sensitivity control option and its potential development either through gesture input or slider control.~~ Unfortunately this tool was not implemented during this project, it is hoped that this may be explored again in future projects.
- **Lamp tool** - Development of a brightness control slider. Testing of multiple light sources and, provided testing is successful, the addition of a multiple lamp feature.
- ~~**Shape selection tool** - Trialling of a variant of the polygonal lasso tool. Depending on the success of this trial, expanding the type of shapes offered by the selection tool in place of or in conjunction with the lasso tool.~~ Although this did not end up being a part of this project, it may be an avenue to reexplore in future MAAP Annotate work.
- ~~**UI redesign** - Reorganising and redesigning the layout of the MAAP Annotate UI, keeping augmented reality and human-computer interaction design principles at the core of the redesign.~~ Although the UI redesign did not end up being a part of this project, the information gathered as part of this research was still applied to the UI elements that were implemented during the development of the other features.
- **Multi-stone viewing** - The implementation of a new feature that allows users to view two or more stones at once.
- **Collaborative Annotation** - Performing additional research on ways of implementing collaborative annotation and their respective costs. Finalization of the method of implementation and the subsequent implementation of the collaborative annotation feature.
- **Performance Testing** - Testing of MAAP Annotate's performance running on a Microsoft HoloLens 1.

Chapter 4: Data Considerations

Although this project primarily focuses on software development, data collection and examination will be done to aid in development. The HoloLens is limited in both its processing power and onboard memory. These factors can impact development, and thus, it is important to discover and understand the limitations of the HoloLens. While being beneficial to this project, this data will also be useful for any future development done on MAAP Annotate.

4.1 Data

To test the performance of the HoloLens while it was displaying multiple stones, a number of metrics were considered. Measuring the CPU utilisation can indicate the level of processing the HoloLens was performing at a time. Looking at the GPU utilisation can also show the current load on the HoloLens GPU while rendering the current scene. Memory utilisation can also show how much of the limited RAM that the HoloLens has was in use at a time. While most applications on the HoloLens are GPU bound [19], the CPU and memory are important to consider when measuring the device's performance. On this basis, the decision was made to record and measure the average frames per second (FPS) of the device while it was running MAAP Annotate. Measuring the FPS gave a more complete measurement of how the HoloLens was performing, as if a bottleneck occurred in either the CPU, GPU, or memory, the FPS would be impacted, and the results could be shown. While the other individual metrics can also be examined, the focus was placed on the average FPS.

4.2 Data Collection

The Window Device Portal displays a wide range of metrics for evaluating the performance of a connected device[20]. It can show a graphical representation of the HoloLens FPS over the previous seconds, however, there is no easy way to export this data. An "export to CSV" button was noticed in the page source code; however, it appears that this is not functioning at this time and thus the FPS values cannot be retrieved from here. The Device Portal will still be used to examine the individual performance of the CPU, GPU, and Memory. If any bottlenecks occur it will be able to show where they are occurring.

In place of the Windows Device Portal, an FPS counter was developed in Unity that can be deployed onto the HoloLens. Its development is discussed further in the Outline of Approach section of this paper. To ensure the tests were as accurate as possible, the minimum number of variables were changed between tests.

- The tests were performed in an application deployed to a HoloLens 1 as opposed to being run using the Unity emulator. This was to ensure this was an accurate test of the HoloLens 1's performance.

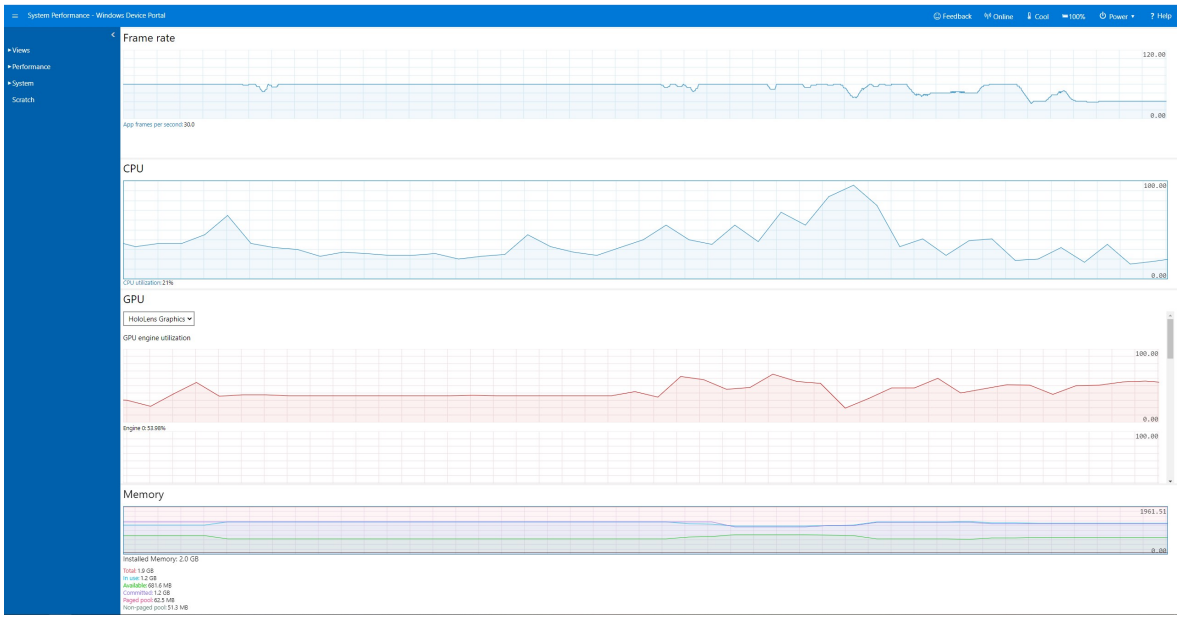


Figure 4.1: Screenshot of the Windows Device Portal

- The test environment was not changed except from the number of stones that were displayed; this was so that no other factors could potentially influence the results.
- The same stone model was used for each test, the Kern15Low stone, this is a low-resolution render that consists of two meshes.
- For each test, the FPS was recorded over a period of 3 minutes.
- During the first minute, basic manipulation of the stones, movement in the scene, rotation, and scaling was performed.
- In the second minute, the HoloLens was kept stationary in order to display all of the current stones at once.
- For the final minute, the HoloLens was tracked from left to right so that the stones left and re-entered the display.

At the end of the test, the collected FPS data was saved locally to a CSV file.

Chapter 5: Outline of Approach

5.1 Setup and Initial Work

At the beginning of the project, the initial setup and gaining familiarity with the HoloLens was undertaken. This involved learning the HoloLens interaction gestures and how to navigate the device.[21] Key programs for development were also installed on the HoloLens at this stage such as Holographic Remoting Player. This program allows for emulation on the device through Unity which provides much faster prototyping and deployment in comparison to the ten minutes or more it can take to build and deploy a solution to the device.

After becoming familiar with the device and its usage, the setup of the development tools was done. This is where an issue arose. The version of MAAP Annotate on the project's GitHub was unable to be built. A duplicate file was discovered that was causing the build errors. Removal of this file allowed the project to be built in Unity, however, the created Visual Studio solution could not be deployed to the HoloLens.

This error appeared to be the cause of a conflict between my testing machine and MAAP Annotate. To try to solve this, a combination of different Unity, Visual Studio, and MixedReality ToolKit versions were trialled. See the table below for all tested versions. Unfortunately, no combination of these programs worked.

	Original MAAP development version	**Newest release version tested**	**Other tested versions**
Unity	5.6.1f1	2020.2.1	2019.4.25, 2018.1.34
Visual Studio	2017	2019	2015
Mixed Reality ToolKit	HoloToolKit for Unity 5.6.1f1	MRTK 2.5.3	MRTK 2017.1.0f3

To fix this, the machine that MAAP Annotate had originally been developed on was sourced and acquired. Using the software with the versions that MAAP was originally developed on meant that MAAP was successfully built. However, again, the Visual Studio solution still could not be deployed through Visual Studio. A fix that had been discovered on the first machine, but that had not worked on that machine, was to edit the created solution files. Specifically, any project.lock.json file would need to be edited so that the UAP version was changed from 10.0.10240 to 10.0. After doing this the solution could finally be deployed and MAAP Annotate could successfully be deployed to the HoloLens.

Code generated in project.lock.json:

```
1{
2 "version": 3,
3  "targets": {
4       "UAP,Version=v10.0.10240": {
5      "Microsoft.ApplicationInsights/1.0.0": {
6        "type": "package",
7        "compile": {
8          "lib/portable-win81+wpa81/Microsoft.
```

```
   ApplicationInsights.dll": {}
9        },
10...
```

Replacement code:

```
1{
2 "version": 3,
3  "targets": {
4       "UAP,Version=v10.0": {
5      "Microsoft.ApplicationInsights/1.0.0": {
6        "type": "package",
7        "compile": {
8          "lib/portable-win81+wpa81/Microsoft.
   ApplicationInsights.dll": {}
9        },
10...
```

5.2 Expansion of Tools

Given the unforeseen difficulties with deploying MAAP, the expansion of the tools was reduced in its scope. Based upon the feedback from the test group, suggestions were made to improve the lamp tool.[1] This project sought to make two improvements in the form of adding another lamp tool and the addition of an intensity controller.

The addition of another lamp tool was simple in nature. The pre-existing lamp tool meant that the second lamp could be based on it. The second lamp allows for more complex lighting of the artwork as two light sources can cast more dynamic shadows. It was positioned above the original lamp tool to be immediately visible to the user.

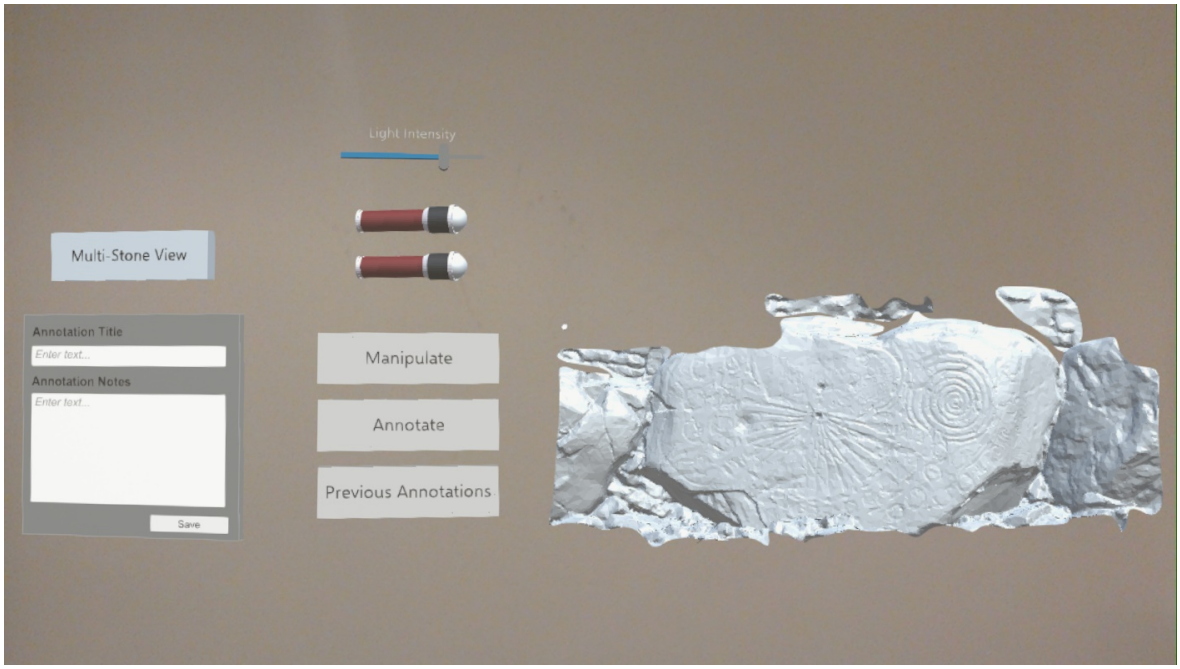


Figure 5.1: Lamp tools and light intensity slider

The light intensity controller allows the user to control the intensity of the light from the lamps. Initially, a two-button design was considered, with a "+" button for increasing the intensity and a "-" button for decreasing it. While this allows for finer control, it also makes large increases time consuming as the user would have to hold the button for a number of seconds. Given the nature of the holograms, small increases in light intensity are not as noticeable, thus fine control of the light intensity is not a requirement. Taking this into account, a slider controller was decided upon. This allows the user to make large changes to the intensity quickly while also having some level of fine control, albeit not to the same degree as the button solution.

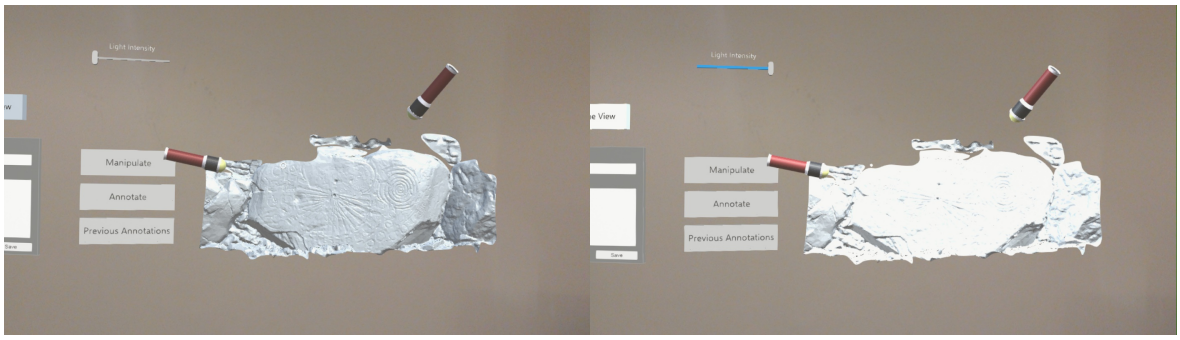


Figure 5.2: Lamp tools activated with different light intensity

In Unity, the lamp tools and slider controller are children of a parent GameObject that positions them above the selection buttons. The existing LampScript.cs was expanded to include intensity adjustment. On each frame the update method is called, the current position of the slider is checked and the intensity is changed in relation to its position. The minimum slider value is set to 0.3 in place of zero. This is to ensure the user does not completely turn out the lamp when adjusting the intensity.

5.3 Peformance Testing

FPS counter and Test Enviroment

For the performance testing, both an FPS counter and a test environment needed to be developed. The counter would be used for recording the current FPS of the application and outputting this to CSV files. The test environment would ensure that the tests were done fairly with no other influences.

The FPS counter is found in the script FPSCounter.cs. On the application start, it creates a persistent path to store the outputs from the counter. Each second, the time taken between the last frame and the current frame is added to a list within the script. When the save button is pressed, this list of FPS readings will be formatted and outputted to a CSV file for that test.

The FPS test environment consists of some basic manipulation controls, the FPS counter, and the number of stones that are being tested. Each test was created as a new scene in Unity. The stones were positioned centrally to ensure that they were what the device was primarily rendering. In the scene, the FPS counter had two buttons. The "Clear FPS" button clears the FPS list, and this is used at the beginning of a test to ensure the list is empty and there are no other values that can pollute the results. The "Save FPS" button outputs the current FPS list to the CSV file for the test.

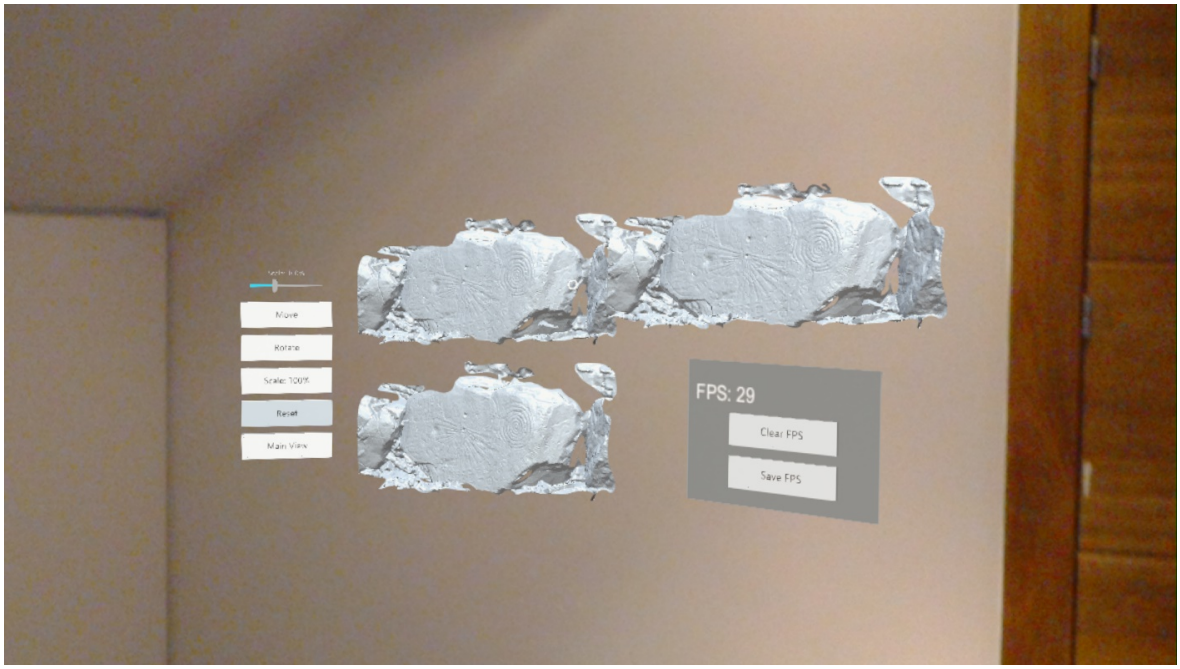


Figure 5.3: Testing environment during 3 stone test

Results

The results of each 3-minute test were averaged and the graph below was produced.

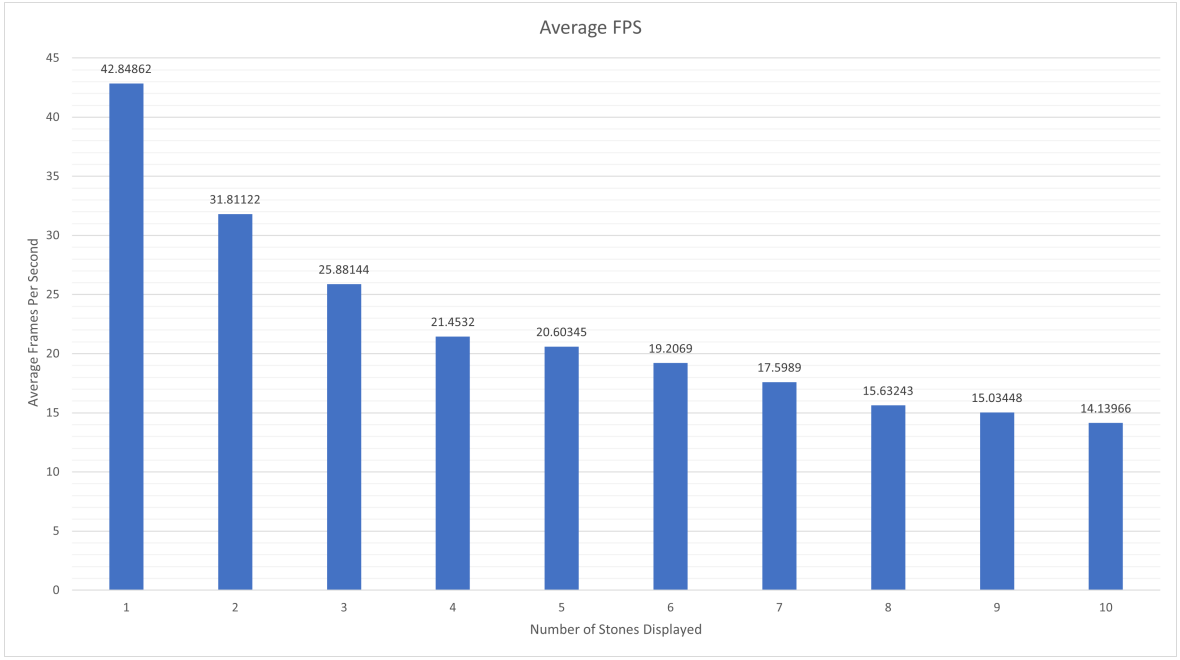


Figure 5.4: Average FPS of the HoloLens during each performance test

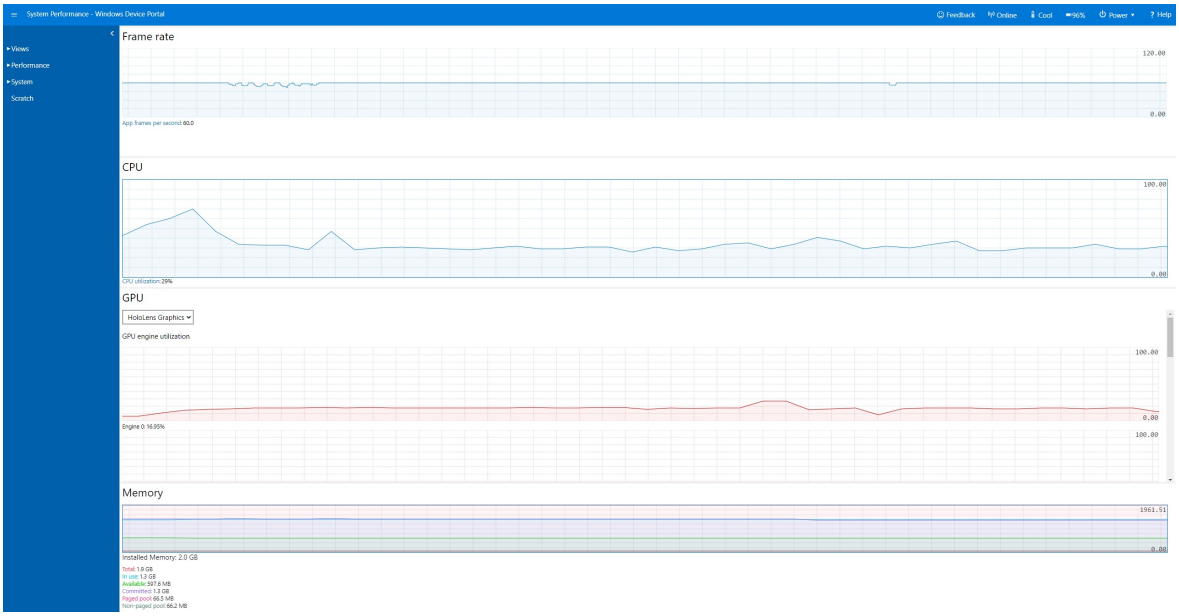
The graph has a downwards trend in FPS as the number of displayed stones increases. This was what was expected as each stone that is displayed requires more resources from the HoloLens. The device manages to handle up to five stones without too large of an impact on the user experience. While twenty frames per second is a third of the refresh rate of the device's displays, this is still with an acceptable limit. Beyond this, however, there are some notable losses in the performance of the HoloLens.

At seven stones displayed, the cursor seems to lag behind the user's gaze as they traverse the screen. This is due to the lower FPS as the cursor's position is not updated as often and, therefore, it feels like it is lagging behind. During this test, the UI elements had a more pixelated look as they are rendered at a lower quality to increase performance.

At nine stones displayed, the HoloLens is under such a load that there is a noticeable delay in rendering. When the user turns their head there are occasional delays in stone renderings, causing a "pop-in" effect where the stones suddenly appear in front of the user.

At ten stones displayed, the delay in loading was noticeably worse and there were also issues in displaying colours correctly. This was not possible to capture with a screenshot or recording as it is believed that this is an issue occurring with the display. When the user turned their head, red and orange colours would appear to "bleed" into the scene at the corners of the screen.

Looking at the Windows Device Portal, we can examine where the performance loss is occurring. Below are graphs showing the resource usage of the HoloLens while idle (for control), while displaying one stone, and while displaying ten stones. We note that there is a significant increase in GPU utilization both when comparing the idle statistics to one stone displayed and when comparing one stone to ten stones displayed. From this, we can see that the GPU is where the bottleneck is occurring.



From the collected data, it is recommended that MAAP Annotate only display a maximum of six stones simultaneously to ensure that the user experience is not impacted. With six stones displayed, there will be some performance lost, however, most of the more noticeable effects seen during the higher tests will not be present. This recommendation is also made due to the fact that these tests were performed with a lower resolution stone. With higher resolution stones that consist of more meshes, performance loss could be seen with even fewer stones displayed on the screen. A separate test was also conducted that consisted of seven stones total, 3 low-resolution Kern15Low made of 2 meshes, and 4 higher resolution Loughcrew TC8 stones which consist of fourteen meshes. This test saw performance comparable to the ten stone test performed here. This shows that high-resolution stones can have a greater impact on the HoloLens performance.

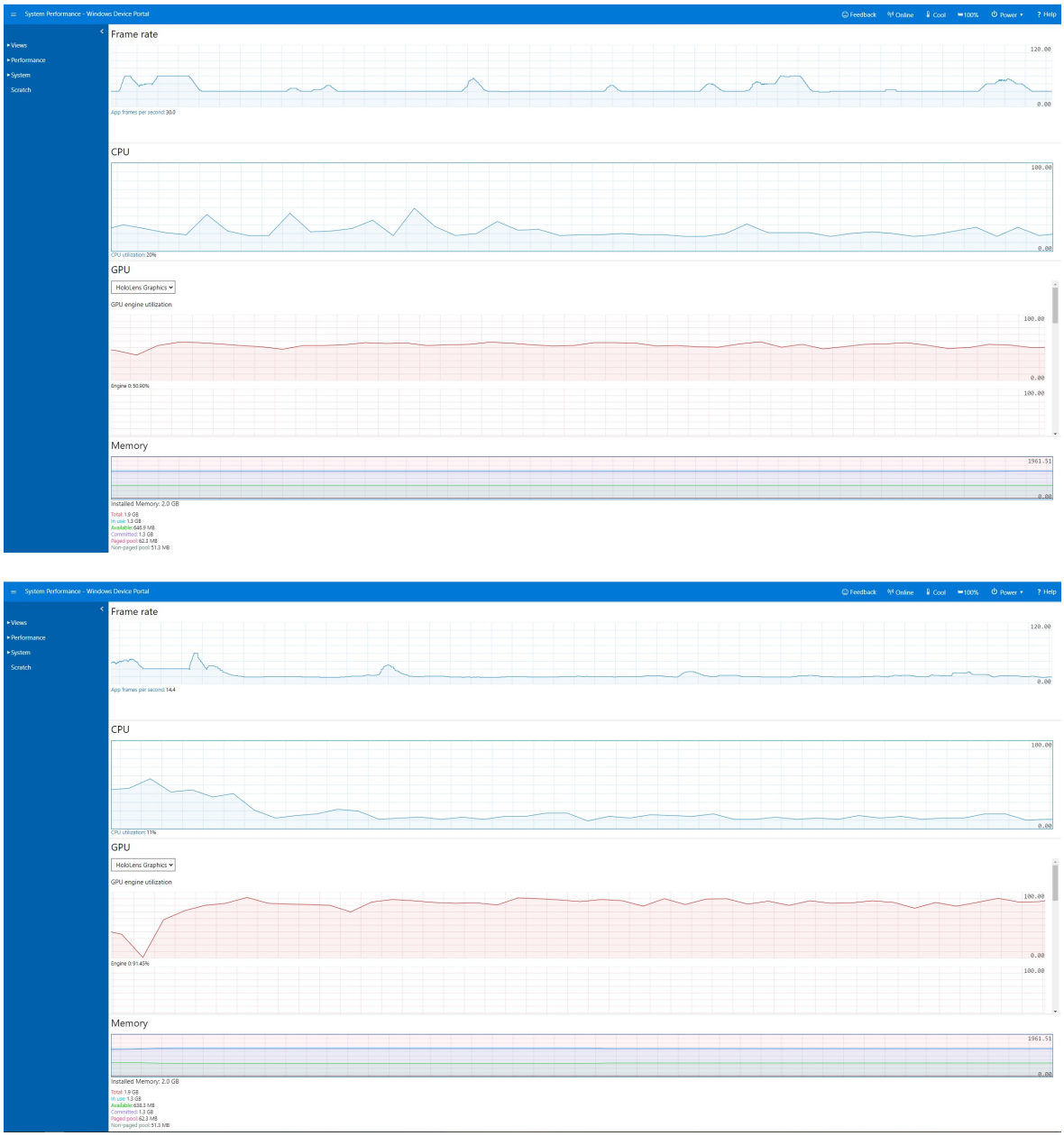


Figure 5.5: Windows Device Portal resource usage of the HoloLens while Idle, displaying 1 stone, and displaying 10. Note increase in GPU utilization.

5.4 Multi-Stone Viewing

One suggestion from the focus group that reviewed MAAP Annotate previously was the ability to view two or more stones at once. The original suggestion was a carousel-style viewer where different stones could be rotated through and viewed. Given the performance testing done prior, it was decided that this approach would not be feasible with the current implementation of MAAP Annotate. The approach was altered and instead a menu style system was prototyped and developed.

From the main view, a user can press the "Multi-Stone View" button. This will load the multi-stone Unity scene. Here the user is presented with options for which stones to view. They can make their selection and when "Next" is pressed the stones will be loaded and the user can manipulate and interact with them. The decision was made to only allow for manipulation options and not full annotation options like in the main view due to performance concerns. In the 2017 MAAP

Annotate paper, Barbier outlines how, if a large or complex annotation is made, it can result in the application crashing. Given that displaying multiple stones at once already results in a performance loss, there was a concern that the annotation issue would be compounded.

Figure 5.6: Multi-stone Selection Menu

The manipulation controls provided to the user are similar to those on the main view with some minor tweaks to improve the user experience. The "Move" option only moves the selected stone and not the UI elements as well unlike the main view. This is to allow repositioning of the stones in relation to each other. The "Rotate" button allows for rotation of a stone to the desired orientation. The "Scale" tool scales one stone at a time, again so that stones can be positioned for better comparison. The "reset" button returns all stones to their starting position and resets their scale and rotation properties. The remaining buttons "New Selection" and "Main View" allow the user to make a new selection of stones and return the user to the main view respectively.

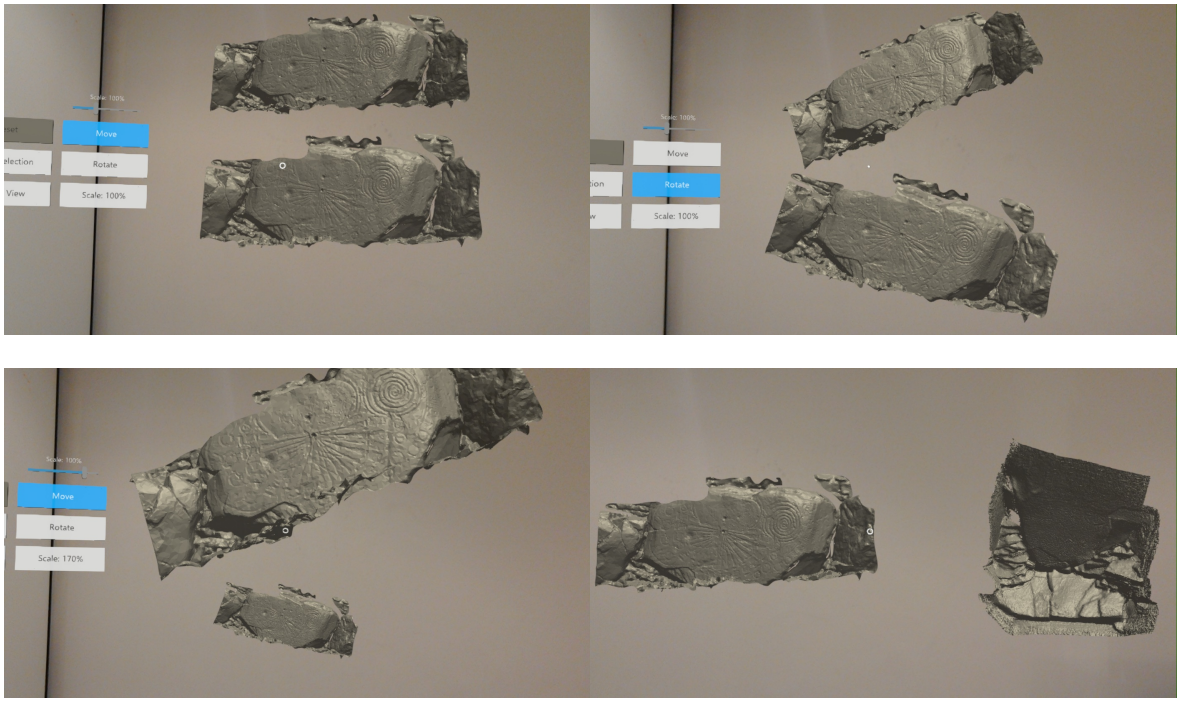


Figure 5.7: Clockwise from top left: Multi-Stone view with Kern and Loughcrew stones, rotation tool, scale tool, new selection made.

The scripting for the multi-stone view is done in the MultiStone.cs file. When the multi-stone view is opened, 4 spawn points are initialized. In Unity, these positions are represented by empty game objects that hold the positions for the spawn points. These spawn points are added to an ArrayList "spawnPoints". A second ArrayList "stoneList" is also initialized. When a stone is selected in the menu it is added to the "stoneList", if it is deselected, it is removed.

When the user presses the "Next" button, the stoneList is looped through. The first stone in the

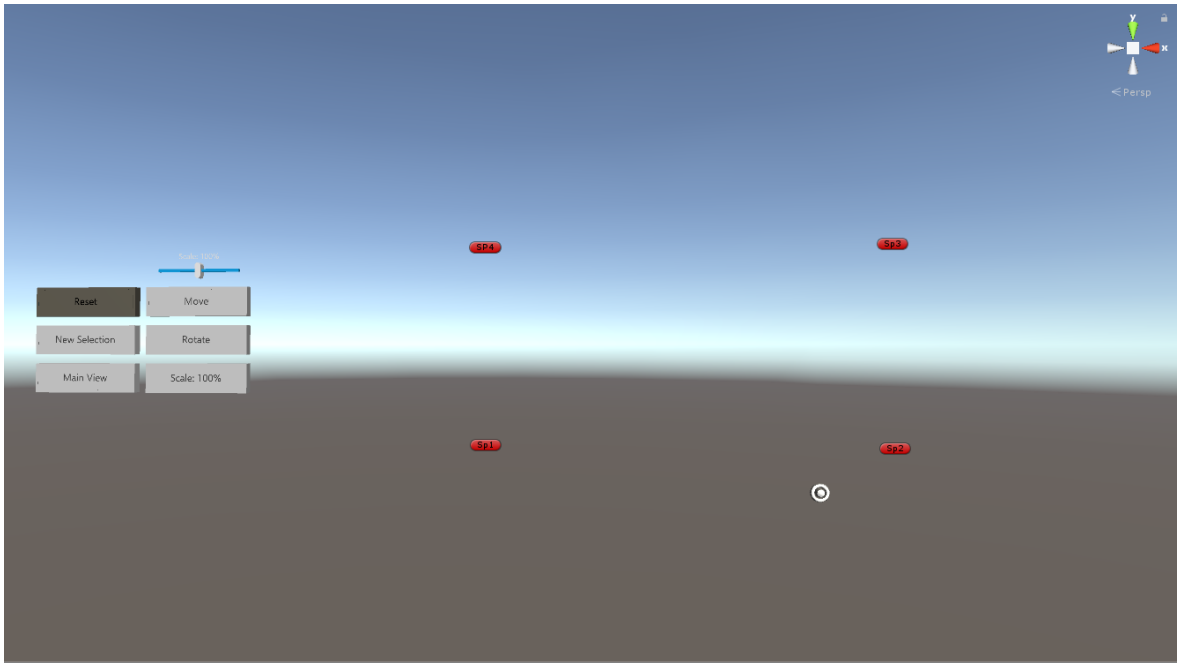


Figure 5.8: Spawn point locations in Unity

ArrayList is positioned at the first spawn point, the second at the second spawn point and so on. This allows for the stones to be positioned in the order that they were selected by the user. For this project, only three stones were added to the menu for selection for demonstration purposes and to ensure adequate performance. In future, more stone options and more spawn points could be added to allow for a greater selection. Again, from the findings of the performance testing, it is recommended that a maximum of six spawn points be used to ensure a good user experience.

5.5 Collaborative Annotation

As the research for this feature was performed it became evident that the goals of this section would have to be adjusted. Initially collaborative annotation was intended to make use of spatial anchors to allow two or more devices to visualise the same stone in an environment. Unfortunately, research showed that spatial anchors were a feature of later Unity versions[22]. Given the inability to update MAAP to a newer version of Unity, a reinterpretation of this feature was made. While MAAP annotate currently had the ability to make annotations, it had no way of saving these annotations. The implementation of saving and loading functionality and the option of ranking these saves was decided upon as the replacement for the intended collaborative annotation.

Saving Annotations

Upon startup of MAAP, *SaveSystem.Init()* is called. This method checks if the path for the Save folder exists. If it does not, one is created. The path is persistent, so even after a restart the save data will remain.

To save an annotation, the annotation tile, text, and data would need to be saved. Since Unity has functionality that allows for the serialization of data into JSON format, it was decided to convert

and store these objects in a JSON format. When completing an annotation, the user presses the save button, Unity collects the text inputs from the annotation window and the positional data of the annotation. This data is serialized and written to a text file. This file is then saved in the previously created save folder. Each saved file is saved sequentially with an ID in place of the user entered title. This ensures that there are no conflicts if a user creates an annotation with a duplicate title. One quirk that was noticed is that the position data from the *drawing tool* is saved as a List of LinerRenderer objects. Unity is unable to serialize this type of data correctly, so to work around the problem, the vector array that holds each of the positional vectors is serialized first. This data is stored as a List of strings which can be serialized by Unity.

Figure 5.9: Annotation Input Window

Annotation Window

To display recently saved annotations, the annotation window was created. It holds up to five of the most recent annotations and displays their title, a view button, and the vote options in an "annotation panel". The vote options allow the user to increase or decrease the score of an annotation. To retrieve the five most recent annotations, the script LoadSavedAnnotations.cs reads the contents of the save folder. As the file IDs are sequential, it retrieves the five most recent and returns them. The annotations title is retrieved from the file and this is returned to Unity to display on the annotation window.

If there are less than five saved annotations, the annotation window will only display a number of annotation panels equal to the total saved. The excess, empty annotation panels, are disabled in the scene.

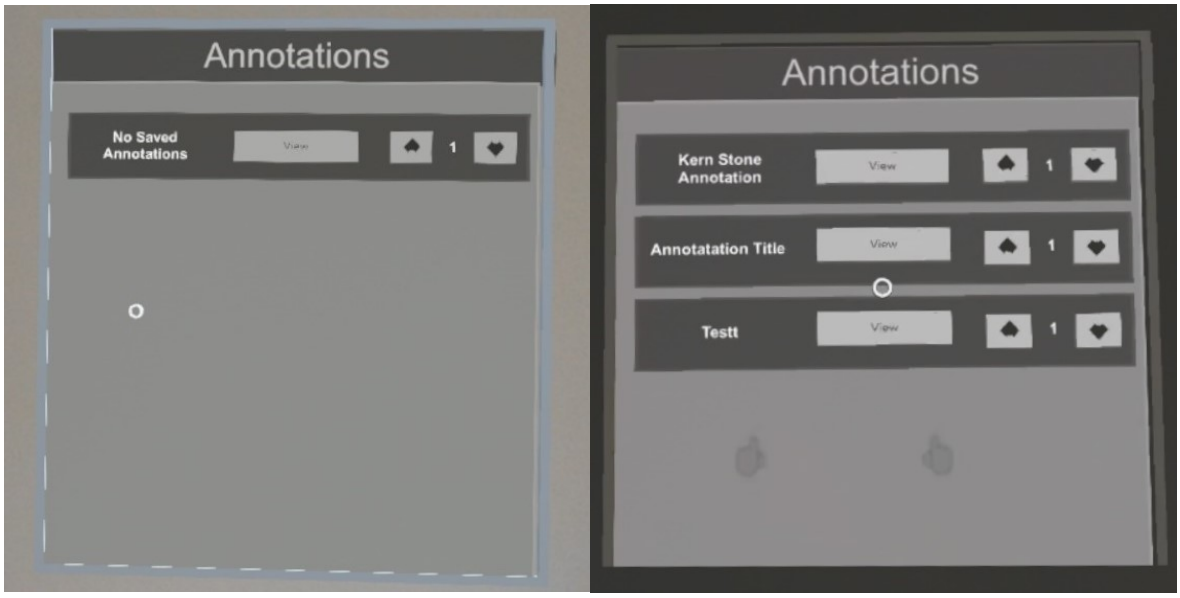


Figure 5.10: Annotation Window, without files (left) and with files (right)

Loading Annotations

When the "View" button is pressed in an annotation panel, Unity changes to the "ViewAnnotation" scene. This scene is used for viewing the contents of the saved file. The text and title are displayed in a text panel and the saved annotation is rendered onto the stone, above the text panel is a button to return the user to the main annotation view.

When changing scene, the ID of the selected file is passed to the new scene. This allows for the "viewAnnoation" scene to retrieve the save file. The annotation title and text are de-serialized from the JSON format and displayed. The positional data is first de-serialized to retrieve a list of strings, then de-serialized to retrieve the vector arrays that contain the positional data. These are then stored back into a list of LineRenderer objects.

As of now, there is only functionality for displaying drawn line annotations, however, this could be expanded in the future to include the rectangle selection tools.

Chapter 6: Conclusion and Future Work

Conclusion

After a challenging start as outlined in the *Outline of Approach* section, the project as a whole reached a successful conclusion. While the goals of the project shifted slightly to account for the unforeseen circumstances, the project was still completed and achieved its outlined goals.

Video Demonstration

Project Github

Initial work

While not intended to take up the amount of time that it did, important findings and bug fixes were made that will hopefully aid in any future development of MAAP Annotate. It was unfortunate that MAAP Annotate was unable to be deployed on the first machine as it had better hardware than the original machine that MAAP was developed on. A newer machine may help with the building and deployment time of MAAP from Unity to the HoloLens which at present, can take over ten minutes. Unfortunately, it could not be tested whether the fault is solely with the first machine or if other machines will have difficulty developing for MAAP as well. For any future work to be done on MAAP it would be important that it could be developed on other machines so testing and adjusting MAAP to achieve this is of vital importance.

Expansion of Tools

While reduced in scope however it was still successful in expanding the functionality of the Lamp tool. If any future work were to occur, one possible suggestion would be to implement a reset button that returns the lamps to their original positions.

UI Improvement

Although the user interface improvement was replaced by the *performance testing* section, the research done in this area was applied to the new UI elements that were added during the other sections. Elements such as buttons and sliders were positioned to ensure usability. Visual feedback was implemented to make sure the users actions were properly highlighted.

Performance Testing

The performance testing provided valuable data for developing applications for the HoloLens and the limitations of MAAP Annotate. It is hoped that this will be useful in future work. One note not mentioned in the *Outline of Approach* is that the HoloLens 2 may solve a lot of the issues presented by the test. With more on-board memory[23] and improved performance, it may be able to handle an increased load in comparison to the HoloLens 1. This opens up many doors for future work like the possibility of the carousel-style stone viewer

Multi-Stone Viewing

Taking on board the results from the performance testing, the multi-stone viewing functionality was successfully implemented. While there are only a limited number of stones and spawn points available to view in this project, the ability to add more in future was left open for when more scans of Megalithic artwork are acquired.

Future development of this feature could include the addition of the annotation tools. As discussed, there is the known issue of larger annotations crashing the application. If this issue is solved, either through code improvements or by a hardware upgrade, then annotations could be added to the multi-stone view. If there is no easy fix for the issue, a workaround solution could be to impose limits on the size of the annotations to prevent the user from creating annotations that are too large.

Collaborative Annotation

While not what had originally been envisioned, the saving and loading functionality that was developed will be useful for any future work in MAAP Annotate. It allows users to save their work and revisit it later due to it being persistent. If future work were to occur on this feature, it would be advised that the save functionality is expanded to include both types of rectangular annotation.

Outside of the save system, if MAAP Annotate is one day successfully updated to a newer version of Unity, revisiting collaborative annotation with spatial anchors would be an interesting project. Research done for this project showed the possibility of using a mobile phone in conjunction with a HoloLens to share objects in AR[24]. This could be an interesting avenue for the MAAP Annotate project to explore.

Discretionary Requirements

While unable to be accomplished during this project, updating to a newer version of Unity and the Mixed Reality ToolKit should be revisited when MAAP Annotate is worked on in the future. The new features added, such as spatial anchors, would potentially, be useful in the further development of MAAP.

Advanced Requirements

Testing of the new features with a test group was unfortunately unable to be done due to the ongoing pandemic. In place of this, a video demonstration of the features was produced. It is

hoped, if more work is done on MAAP, that these new features will have the opportunity to be tested to devise their functionality and usefulness.

For this project, development focused solely on the HoloLens but, if possible, a virtual reality port of MAAP Annotate should be considered. Virtual Reality headsets such as the Oculus Quest are more available in comparison to the HoloLens. Having a VR application of MAAP Annotate would increase the number of people that the application is available to.

Chapter 7: **Acknowledgements**

I would like to thank Dr Anthony Ventresque for his help and guidance with this project, Johanna Barbier for her assistance in debugging MAAP, Patricia Kennedy for guidance in the area of Megalithic artwork, and Robert Young for his help in sourcing the hardware used for this project.

Chapter 8: Appendix

This section was originally part of the interim report. It has been placed here as it is not relevant to the final report, but still shows the initial intentions and goals of the project.

Project Workplan

8.1 Workplan

Figure 8.1: Gantt chart of my proposed project timeline

The project is set to take place over the course of the 2021 UCD Spring Trimester. Each section of the project has been broken down and shown on the chart in relation to when in the semester it will hopefully take place. While the intention is to stick to this plan, it is acknowledged that there will be other responsibilities during the trimester and that development projects by nature have unforeseen aspects that can result in changes to the development time.

Inside each development block (blocks 2, 3, 4, and 5) a standard development cycle will occur:

1. Planning and further research
2. Analysis and definition of feature
3. Design of feature
4. Implementation
5. Testing and debugging
6. Revisions

It is planned this way so that each feature gets its own dedicated research, development, and testing time. This is opposed to a more global development cycle where each feature would be

going through research, development, and testing at the same time. This approach could easily result in the project becoming disjointed and unfocused.

8.1.1 Setup

The projects setup phase will take place over weeks one and two of the semester. It will entail the setup the required software, such as Unity and HoloLens emulator, and hardware in the form of the HoloLens. Two weeks have been allocated for this section as during this phase further analysis will be done on the existing project code to decide if it will be possible to upgrade it to the most recent version of Unity. If it can be done without much difficulty this will occur on the second week of the project setup.

8.1.2 Expansion of Tools

Improvements to the interactive tools will be made during weeks 2-4. This period overlaps with the setup section as if the project is not being updated, then this section will begin in week 2. The tools that are set to be improved and expanded upon are:

- Drawing Selection Tool
- Lamp Tool
- Rectangle Selection Tool

8.1.3 Multi-Stone View

The multi-stone view feature will begin development during week 4. It is starting here as the planning and analysis for this development will begin as the expansion of the tool section is concluding. The limiting factor of this feature is the limited memory on the HoloLens which the solution will have to account for.

8.1.4 Collaborative Annotation Feature

The largest block has been allocated to the collaborative annotation feature. This block also falls during the mid-term which should allow for more time for it to be worked on. This feature is predicted to be the most complex in nature. It will require further research into collaboration in AR and a decision will need to be made on whether to adopt full AR collaboration or to allow collaboration from other devices.

8.1.5 UI Improvements

The UI improvements have been left until last as with the addition of proposed features the UI will be changing to accommodate them. It would be an inefficient use of time to update the UI at the beginning of the project only to have to update it again at the end to add in the new features. These changes should primarily be aesthetic in nature and thus will not require as much

development compared to other features. This section overlaps with the Collaborative Annotation in case of early completion of that section. It overlaps with the code clean up and test group sections, the reasoning for which is explained in their descriptions.

8.1.6 Code Clean-Up

In weeks 11 and 12 the code that has been developed over the course of the project will be cleaned up. This entails revising the code and making sure it is displayed in a clean and well-commented manner to assist any future developers. This section overlaps with the UI section as they are not as programming intensive as early sections and can be done in unison.

8.1.7 Test Group

This section will not require three weeks. However, three weeks have been allocated to allow for some leniency in the timeframe due to the pandemic and the uncertainty with if and how the testing can be accomplished. If the tests can occur, ideally the features will be trialled by a group of Archaeologists and if possible by some members of the same group that tested MAAP Annotate last time. These tests will be done on UCD campus over the period of a day or two and will be similar in nature to the previous MAAP Annotate test. Time is also allocated in this section for the aggregation and compiling of the results.

8.1.8 Report Write Up

While notes will be taken over the course of the project, they will be compiled and organised into a report in the final two weeks.

8.2 Project Completion

Under normal circumstance, the project goal would have been to develop and expand the MAAP Annotate project and have had these features tested by a test group. However, given the restrictions that are in place during the conception of the project and that might still be in place during its development and completion, an alternate goal of delivering a recorded video demonstration of the working features that were developed would be accepted as successful completion of the project.

Bibliography

1. Barbier, J. *et al. MAAP Annotate: When archaeology meets augmented reality for annotation of megalithic art* 2018.
2. Turner, A. & El Hamalawi, M. *Install the tools - Mixed Reality* Sept. 2020. https://docs.microsoft.com/en-us/windows/mixed-reality/develop/install-the-tools?tabs=unity.
3. Ventresque, A. *aventresque/MAAPAnnotate* Oct. 2017. https://github.com/aventresque/MAAPAnnotate.
4. Jordan, P. W., Thomas, B., McClelland, I. L., Weerdmeester, B. & Brooke, J. in *Usability Evaluation in Industry* 189–204 (Chapman and Hall/CRC, 2014).
5. Guzman, A. *Photoshop Brush Tool: A Basic Guide* Oct. 2020. https://design.tutsplus.com/tutorials/photoshop-brush-tool-a-basic-guide--psd-5200.
6. Patterson, S. *The Polygonal Lasso Tool - Photoshop Selections* Apr. 2019. https://www.photoshopessentials.com/basics/selections/polygonal-lasso-tool/.
7. Keighrey, C., Flynn, R., Murray, S. & Murray, N. *A QoE Evaluation of Immersive Augmented and Virtual Reality Speech Language Assessment Applications* in (June 2017).
8. Hammady, R. & Ma, M. *Designing Spatial UI as a Solution of the Narrow FOV of Microsoft HoloLens: Prototype of Virtual Museum Guide: 6th European Conference, ECIL 2018, Oulu, Finland, September 24–27, 2018, Revised Selected Papers* 217–231. ISBN: 978-3-030-13471-6 (Jan. 2019).
9. Evans, G., Miller, J., Pena, M. I., MacAllister, A. & Winer, E. *Evaluating the Microsoft HoloLens through an augmented reality assembly application* in *Defense + Security* (2017).
10. Hall, R. & Hanna, P. The impact of web page text-background colour combinations on readability, retention, aesthetics and behavioural intention. *Behaviour IT* **23,** 183–195 (May 2004).
11. Mavitazk, M. *Color light and materials - Mixed Reality* Mar. 2018. https://docs.microsoft.com/en-gb/windows/mixed-reality/design/color-light-and-materials.
12. Park, Y. *Typography - Mixed Reality* Mar. 2019. https://docs.microsoft.com/en-us/windows/mixed-reality/design/typography.
13. Barbier, J. *MAAP Annotate* Oct. 2017. https://www.youtube.com/watch?v=XUPRaAGhMAw.
14. Godin, K. *Use spatial sound in mixed-reality applications - Mixed Reality* Feb. 2019. https://docs.microsoft.com/en-us/windows/mixed-reality/design/spatial-sound-design.
15. Kaufmann, H. Collaborative Augmented Reality in Education (Mar. 2003).
16. Chusetthagarn, D., Visoottiviseth, V. & Haga, J. *A Prototype of Collaborative Augment Reality Environment for HoloLens* in *2018 22nd International Computer Science and Engineering Conference (ICSEC)* (2018), 1–5.
17. Turner, A. *Spatial anchors - Mixed Reality* Feb. 2019. https://docs.microsoft.com/en-us/windows/mixed-reality/design/spatial-anchors.

18. Chen, H., Lee, A. S., Swift, M. & Tang, J. C. *3D Collaboration Method over HoloLens™ and Skype™ End Points* in *Proceedings of the 3rd International Workshop on Immersive Media Experiences* (Association for Computing Machinery, 2015), 27–30. ISBN: 9781450337458. https://doi.org/10.1145/2814347.2814350.

19. Ferrone, H., Coulter, D. & Valverde, L. *Understanding Performance for Mixed Reality - Mixed Reality* Mar. 2019. https://docs.microsoft.com/en-us/windows/mixed-reality/develop/platform-capabilities-and-apis/understanding-performance-for-mixed-reality.

20. Bridge, K., Coulter, D., Jacobs, M. & White, S. *Windows Device Portal overview - UWP applications* Jan. 2021. https://docs.microsoft.com/en-us/windows/uwp/debug-test-perf/device-portal.

21. Eggers, M., Kerawala, S. & Paniagua, S. *Getting around HoloLens (1st gen)* Sept. 2019. https://docs.microsoft.com/en-us/hololens/hololens1-basic-usage.

22. Parker, R., Coulter, D. & Wojciakowski, M. *Azure Spatial Anchors Unity overview - Azure Spatial Anchors* Feb. 2021. https://docs.microsoft.com/en-us/azure/spatial-anchors/unity-overview.

23. Microsoft, N. D. *HoloLens 2-Overview, Features, and Specs: Microsoft HoloLens* 2021. https://www.microsoft.com/en-us/hololens/hardware.

24. Developer, M. *Developing Mobile Augmented Reality (AR) Applications with Azure Spatial Anchors - BRK2034* May 2019. https://www.youtube.com/watch?v=CVmfP8TaqNU.