Interactive 3D Map via OpenStreetMap

Melchizedek Gray

B.Sc. Computer Science

Lancaster University, England, United Kingdom

Abstract

Write abstract here.

• Aims

• Design

• Implementation

• Findings

[Chapter 1. Introduction 4](#_Toc130253327)

[1.1 Aims of This Project 5](#_Toc130253328)

[Chapter 2. Background 6](#_Toc130253329)

[2.1 Overview of Existing Navigation Applications 6](#_Toc130253330)

[2.2 OpenStreetMap 6](#_Toc130253331)

[2.3 A-Frame 7](#_Toc130253332)

[2.4 Robert Kaiser’s VR Map 9](#_Toc130253333)

[Chapter 3. Technical Implementation 11](#_Toc130253334)

[3.1 Design of the System 11](#_Toc130253335)

[3.2 Requirements 11](#_Toc130253336)

[3.3 Overview of Languages and Software Used 12](#_Toc130253337)

[3.3.1 General Development Setup 12](#_Toc130253338)

[3.4 Discussion of OpenStreetMap Data Integration 12](#_Toc130253339)

[3.4.1 Getting the Data and Displaying It 12](#_Toc130253340)

[3.5 Explanation of 3D Rendering and Interactive Components 14](#_Toc130253341)

[3.6 Height Accurate Map 16](#_Toc130253342)

[Chapter 4. User Study 18](#_Toc130253343)

[4.1 Explanation of Methodology for Data Collection 18](#_Toc130253344)

[4.2 Description of Research Questions 19](#_Toc130253345)

[4.3 Analysis of User Feedback 20](#_Toc130253346)

[4.4 Evaluation of the 3D Map Implementation 22](#_Toc130253347)

[Chapter 5. Conclusion 23](#_Toc130253348)

[5.1 Recap of Key Findings and Results 23](#_Toc130253349)

[5.2 Discussion of How the Project Could Be Extended 23](#_Toc130253350)

[5.3 Final Thoughts on This Project 23](#_Toc130253351)

# Chapter 1. Introduction

For over two thousand years, humans have been practising cartography to create detailed maps [1] and without them, settlements would not have been able to communicate effectively, trade routes would not have been established, and navigation between countries over the sea would have been extremely difficult. Physical maps are helpful for navigating from Point A to Point B given that A and B are on the map with enough detail of the routes between. However, one major problem with a physical map is that as A and B get further away from each other the level of detail on the map diminishes greatly, e.g., villages between would be reduced to text. If you wanted to travel to multiple villages, towns, and cities and then explore those places, you would require multiple maps. This is where mobile phones step in. With their digital screens, they are able to show a map and have that map update as the user moves or interacts with it.

Considered to be the first mobile phone with Global Positioning System (GPS) built-in, the ‘Benefon ESC!’ was released in 2001 [2] and it paved the way for a new era of cartography. As GPS-capable phones became more widespread, some companies took notice of this new emerging market and capitalised on it and now, as of 2023, the total revenue in the navigation segment is over $1 billion [3] and is projected to reach an impressive $1.6 billion in 2027 [3].

Many companies in the mapping industry have strict licenses on who can access and use their data, such as Google Maps, which was made public in 2005 [4]. However, alternatives do exist such as OpenStreetMap (OSM). OSM is a freely accessible, community-driven mapping database that can be edited by anyone under the Open Data Commons Open Database License. By utilising its three main elements, nodes, ways, and relations, OSM allows for the inclusion of practically any geographical structure on the map, including, buildings, rivers, trees, bus routes, etc.

As the real-world changes, so too can a virtual map that uses OSM’s database. Due to its open data nature, anybody can update the map as they please, facilitating near real-time updates. This is extremely important for universities as they tend to constantly expand with new lecture, lab, and residency buildings. As an example, Lancaster University last erected a building in 2021. [5]. The ability to add that building to OSM’s database as soon as the doors open allows any mapping software using OSM to start displaying that building on day one.

In the recent years that we transitioned from paper maps to digital maps, the obvious evolution was to keep the format of a traditional bird’s eye view, especially due to the lack of performance available on mobile devices. This top-down view is excellent for understanding your whereabouts in a large area making it great for driving at speed as you can see upcoming turns easily. Though, with the walking speed of humans being considerably slower than a vehicle, we do not need to see nearly as far allowing for a more zoomed-in map. However, humans are not used to seeing the world from above and popular map applications such as Google Maps and Apple Maps do little with the extra zoom. The lack of visual landmarks for the next turn and reliance on audio prompts creates a disconnected user experience, lacking a human touch.

A three-dimensional environment from the perspective of a human where the user can look around using their phone and easily identify buildings and paths and be able to visibly associate the next turn with a specific geographical feature would solve this issue. One benefit to this approach would be that the navigation feels more organic allowing the user to more intuitively get around. Another benefit would be that integrating the user’s experience in the world around them with the application allows for a more immersive experience. Overall, the evolution of navigation and 3D technology presents exciting opportunities to revolutionise the way we navigate.

## 1.1 Aims of This Project

This project has three overarching aims:

* Explore the use of open-data mapping databases to generate a 3D environment of the user’s current location.
* Investigate the effectiveness of a navigation system in that 3D space.
* Evaluate the implementation through a user study to understand the needs and wants of the users of such a system.

These aims will attempt to be met by using the A-Frame framework and OpenStreetMap’s open-data database to develop the map in a web browser and then conducting a user study on the implementation.

# Chapter 2. Background

## 2.1 Overview of Existing Navigation Applications

In terms of its user base, Google Maps dominates the mobile phone navigation market. A study completed in 2022 shows Google Maps as the fourth most popular application in the UK with an audience reach of 69% [6]. To give this some perspective, this is ahead of Facebook Messenger, Instagram, and Amazon, and with no other navigation applications below it for the next sixteen applications. Google Maps is quite the staple in navigation apps. It has grown from just navigating to being an all-in-one travel guide. It can suggest restaurants, suggest nearby hotels and how much they cost, and take you to sightseeing spots. It even shows customer reviews of restaurants and take-outs. This makes Google Maps quite difficult to compete against.

However, there are some navigation applications that have unique features, which have gotten recognition, such as Waze. In a study done in 2022 by AppMagic, Waze was the second most downloaded navigation app in the United States with 9.12 million downloads [7]. Waze is based around the community bettering the app's functionality for others by reporting certain things. For example, a person driving using Waze could report through the app that there has been an accident or that there is a speed camera. This will alert other drivers and even change their route if necessary.

Navigation applications play a key role in helping us get around, but I believe they have stagnated in their innovation, and they still have the capability to evolve. Overall, with advancements in technology such as virtual reality and 3D mapping techniques, there is potential for navigation applications to create a more immersive and inuitive experience for their users.

## 2.2 OpenStreetMap

Founded in 2004 by Steve Coast, OpenStreetMap was founded to give people, who face various barriers, access to free geographical data. Their database covers the entire world, including various geographical features such as buildings, shops, roads, trees, bins, and more. As mentioned earlier, OSM uses nodes, ways, and relations to label such data. A node is a single point on the map, a way is made up of a string of nodes, and a relation is a group of members that is comprised of an ordered list of one or more nodes, ways, and/or relations. A relation is used to describe a geographical structure between different objects, such as a lake with an island in the middle.

These elements can then be labelled with tags; tags consist of a key and value pair with there being over 90,000 unique keys [8]. For example, a node could have the tag “amenity=cafe” with the “name=…” tag denoting its name. A way could have the tags “highway=pedestrian” and “lit=yes”, this indicates that the way is a pedestrian path, and it is lit with streetlamps. A closed way (a way that forms a complete loop) could have the tags “building=university” and “amenity=library”, to indicate that it is a university library building. Tags can be invented and added by anyone if there is no existing tag scheme for what is wanting to be added. Using these three basic elements with thousands of well-established tags and the ability to add tags, anything can be included in OSM’s database.

The collaborative side of OpenStreetMap can be both a strength and a weakness, relying on the trust of the community to keep the data accurate. If 1% of the users mistakenly or purposefully make errors, the other 99% are relied on to correct those errors. This allows for the map to organically grow into an accurate representation of the real world, which is great for this project. Any changes that are made on campus can be reflected in the 3D environment immediately by simply updating OSM’s database and any inaccurate data will be corrected quickly.

Overall, OpenStreetMap's accessibility, flexibility, and collaborative nature make it a powerful tool for creating accurate and up-to-date maps for various applications, which makes it a great candidate for this project.

## 2.3 A-Frame

A-Frame is a web framework for developing interactable 3D environments. It is based on top of HTML and three.js and uses a powerful Entity-Component System (ECS) that implements the composition over inheritance and hierarchy principle. An entity is comprised of one or more components that describe it. For example, a box entity may have a position, geometry, and material components. A couple of the benefits of an ECS, as listed on A-Frame’s website [9], are greater flexibility when defining objects by mixing and matching reusable parts – this relates to using A-Frame’s mixins where the same material component could be used across multiple entities – and it allows for extending new features that can be shared with other developers as components.

To start developing using A-Frame, it is as simple as including the script in your HTML file and writing a few lines of code. The first step to displaying 3D objects on the webpage is to add an ‘<a-scene>’ entity to the HTML. Every shape that is wanting to be rendered must be a child of this entity. To get started with displaying some shapes, A-Frame provides some very handy primitives, including cubes, spheres, cones, and twenty-four others. Figure 2-1 is an example scene that demonstrates A-Frame’s incredible simplicity. The example scene adds a box, a sphere, and a cylinder onto a plane which acts like a floor. The background is then set to a specific colour using the sky primitive. Figure 2-2 shows the scene running in a webpage.

A-Frame’s simplicity made it an excellent choice for me as this project was my first venture into developing in the browser and developing 3D environments. Additionally, A-Frame’s extensibility meant that there was plenty of online documentation and extra entities and components that people had created and shared, some of which I have talked about in the implementation chapter.



Figure 2-1: A-Frame example scene code. From https://aframe.io/docs/1.4.0/introduction/

## 2.4 Robert Kaiser’s VR Map

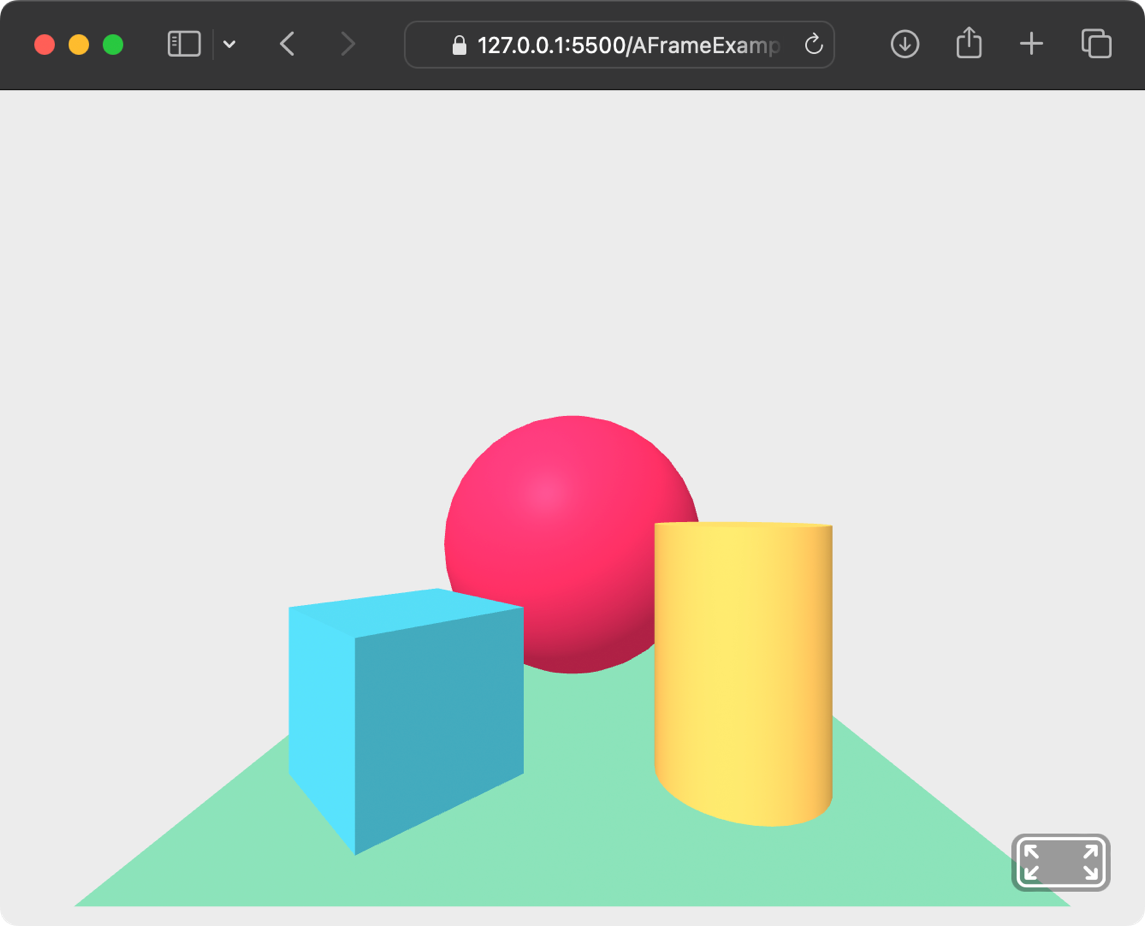


Figure 2-2: Webpage of the A-Frame example scene. Image taken by Melchizedek Gray.

Robert Kaiser developed a WebVR demo using both OpenStreetMap’s data and A-Frame to display a 3D map of a given area in a browser [10]. His implementation renders buildings and trees as 3D objects, which are placed on a floor that is a map of that area. See figure 2-3 for a visualisation. This demo effectively showcases the potential of using 3D technologies to display a navigable landscape.

However, his implementation has a major limitation that highlights its demonstrative nature and that is the map does not update as you move. On the start screen, there are numerous pre-set options for places, such as New York Plaza and Vegas Mirage, and the option to get your current location. There is also an option to manually enter your latitude and longitude for a specific place. Once the map has loaded, you can fly around using WASD, but no more data will be loaded for that area. The ability to update the map as the user moves is crucial for a production-quality map application that is intended to be used for navigation. The implementation discussed in this document fulfils this role, as it allows for a more interactive and user-friendly experience.

Another crucial feature for practically every map application is the ability to search for a specific location or input an address and receive turn-by-turn directions to that place. This is a feature that this demo is lacking. It was necessary for me to include this functionality in my implementation to fulfil its role of being a mapping application.

In summary, while Robert Kaiser’s WebVR demo shows the exciting potential of combining A-Frame and OpenStreetMap to create a 3D map navigation application, it falls short in a number of areas. My implementation addresses these shortcomings and in general improves upon his work.

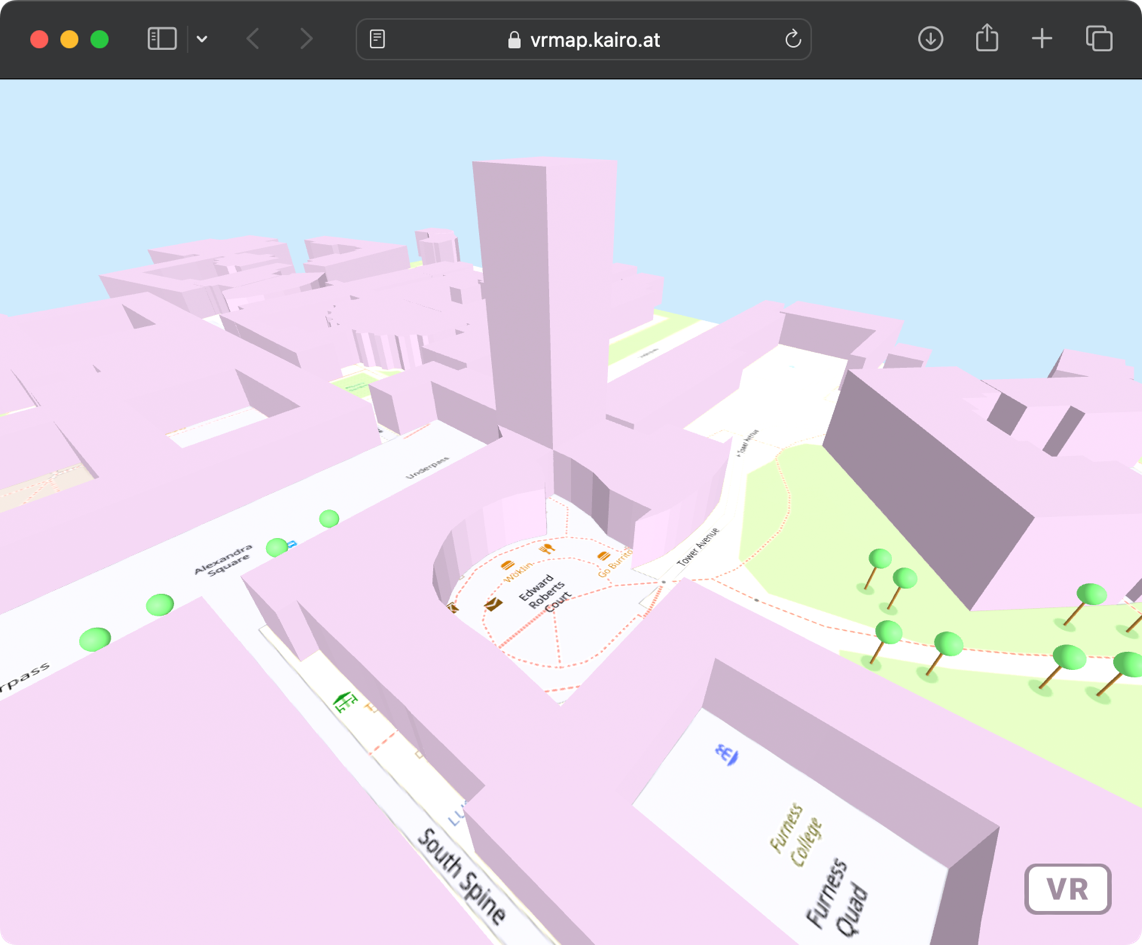


Figure 2-3: Robert Kaiser's 3D map implementation at Lancaster University. Image taken by Melchizedek Gray.

# Chapter 3. Technical Implementation

## 3.1 Design of the System

For the system's user interface (UI) design, I started with several sketches for what the webpage could look like. Due to this being my first foray into mobile development, sketching out some designs and jotting down my thoughts really helped me understand the space I had to work with. However, I made the decision to focus on the implementation side of the project first. Though towards the end of the implementation, attention was paid to ensure the user interface was visually pleasing and functional.

My first goal was to create an extremely basic UI that would allow the user to open the webpage and immediately start using the map. Doing this allowed me to put more effort into developing the main aim of this project first.

I don’t really know what else to write here.

## 3.2 Requirements

For this implementation to meet the goals laid out in chapter 1, functional and non-functional requirements had to be met. Below are the functional requirements of this project.

The implementation shall:

* display a 3D map of the area.
* show various geographical features such as buildings, paths, trees, rivers, and grass areas.
* update the map as the user moves.
* allow the user to enter a place to navigate to and then display directions there.

The non-functional requirements are listed below:

The implementation should:

* display the map within 6 seconds of the user pressing the button to load the map.
* work on Chrome, Safari, and Firefox.

## 3.3 Overview of Languages and Software Used

### 3.3.1 General Development Setup

For the development of this project, the text editor Visual Studio Code was chosen due to my familiarity with it and its flexibility. One extension for Visual Studio Code that was crucial to development was Live Server. This extension made it possible to have the website open while developing it. Any changes made were instantly reflected on the open webpage. Not only did this allow for the website to be run on my laptop but also by any device that is connected to the same local network such as my phone, which was critical for developing a website that is designed primarily for phones.

The main browser used during development was Chrome on MacOS. With its great dev tools and its excellent compatibility with all the latest technologies, it was a smart choice. However, due to its excellent compatibility, extra research had to be taken before using a specific feature in case it was not supported by other popular browsers. One such feature that caused me confusion was using ‘importScripts()’ from within a JS web worker to retrieve a file from a CDN. This feature is supported by Chrome, Firefox, and Safari Technology Preview (a beta version of Safari) but not by Safari despite the MDN Web Docs not mentioning it. It works with scripts that do not need to be fetched from outside the current folder. I am unable to find an answer to whether this is purposeful or a bug.

To evaluate the implementation in real scenarios during development, the use of a website hosting service was needed. GitHub Pages filled my needs perfectly. As new features were added and pushed to my GitHub repository, the webpage would update automatically.

## 3.4 Discussion of OpenStreetMap Data Integration

### 3.4.1 Getting the Data and Displaying It

To access OSM’s rich database, the read-only API, Overpass API, was used. One can query this API to request specific data from a specific area. Another option would have been to pre-download the specific Lancaster area and send that over with the webpage. Choosing the former allows for my implementation to always stay up to date with the current database with little-to-no upkeep and to be lighter, to deliver the website quicker to the user’s device. However, this does incur a time penalty as there are load times that often vary drastically. To combat this, a chunking system was implemented using the ‘cache’ interface in JS. After the map has initially loaded its first chunk around the user, the API is then queried eight more times (buildings and paths are queried separately) for each chunk that is at the edge of the current chunk. The responses are then stored in the local cache of the browser so that when the user travels close enough to the next chunk, the chunk can immediately load in front of them. Then, the next lot of queries are sent out for the next chunks. This keeps the loading times to a minimum as they navigate using the application.

Querying the API is executed in three requests; requesting the buildings is one request, roads and footpaths are another, and trees, rivers, bodies of water, and grassy areas are grouped in the final request. Doing it this way decreases the total time that the user will stare at a blank screen as they wait for the map to load. Additionally, if one of the queries is unsuccessful, the remaining two types of structures can still be loaded. If a query does fail, a new query with the same parameters is sent out. If it fails again, it will retry eight more times with the time between each query doubling starting at 1 second.

Overpass Turbo, the interactive frontend to Overpass API, was used to construct the API queries required. The website allows you to enter and run an Overpass query and the results are then displayed on an interactive map or as raw text. This tool allowed me to build syntactically correct queries and ensure that the requested data was correct.

In an effort to minimise time spent blocking the UI thread, JS workers were implemented to execute the fetching of the OSM data. Blocking the UI thread should be kept to an absolute minimum as it heavily affects the responsiveness of the website. Each of the three queries has a worker that it can use and when a fetch request is needed, a message will be sent to the worker via a ‘postMessage(message)’ method call. The message sent to the worker will contain the query and the name of the cache where the response should be stored. The worker will then check the cache to see whether this request has already received a response. If it has, it sends a message back to the main thread with the response, otherwise, it sends the request to Overpass API. Once the worker has received the response from the API call, it will store the response in the cache and return the response to the main thread.

To parse the data from the API query, osmtogeojson.js was used. osmtogeojson.js is a JS library that does exactly as advertised; it takes the response from an OpenStreetMap API call and converts it to a GeoJSON. The reason for doing this is that it rearranges the data into an easier-to-use format. Figures 3-1, 3-2, and 3-3 illustrate the changes it makes.



Figure 3-1: Raw text from an Overpass query Part 1.

Figure 3-2: Raw text from an Overpass query Part 2.

Figure 3-3: GeoJSON version of the data.

Figures 3-1 and 3-2 show the data received from the response and figure 3-3 shows how the data is rearranged by replacing the node IDs with the nodes. It also figures out the type of geometry it is. In this case, it has been identified as a polygon.

## 3.5 Explanation of 3D Rendering and Interactive Components

In order to add each geographical feature, a new entity is created, given a geometry, material, position, and scale, and added as a child element to the scene entity I mentioned earlier in the Background chapter. However, the existing geometries in A-Frame are too rudimentary to display a building so to combat this, using A-Frame’s excellent extensibility, Robert Kaiser created his own custom geometry. This geometry interacts with the three.js layer directly to create the polygon necessary for his buildings. This geometry was extremely helpful in teaching me how custom geometries and custom components work in A-Frame. After reading and understanding the code, I made a couple of tweaks and used it to add buildings to my implementation. My tweaked version of the geometry also allowed me to add paths, grass areas, and water in the later stages of development. Figure 3-5 shows how a custom geometry is registered with A-Frame. Lines 2 to 7 in figure 3-4 illustrates how the custom geometry is then utilised.During my time developing, I found out about a 3D rendering technique called geometry instancing. An article on “Game Performance: Geometry Instancing” explains it well with “Geometry Instancing is a technique that combines multiple draws of the same mesh into a single draw call, resulting in reduced overhead and potentially increased performance. This works even when different transformations are required” [11]. Three.js has support for this technique via InstancedMesh, which can be used to render a large number of shapes with the same geometry and colour while allowing each copy to have a different position, scale, and rotation. I was able to efficiently draw any number of trees on the map in only 14 draw calls using three.js' InstancedMesh, thanks to a set of A-Frame components created by Diarmid Mackenzie [12], which allowed for easy integration of InstancedMesh into my implementation.

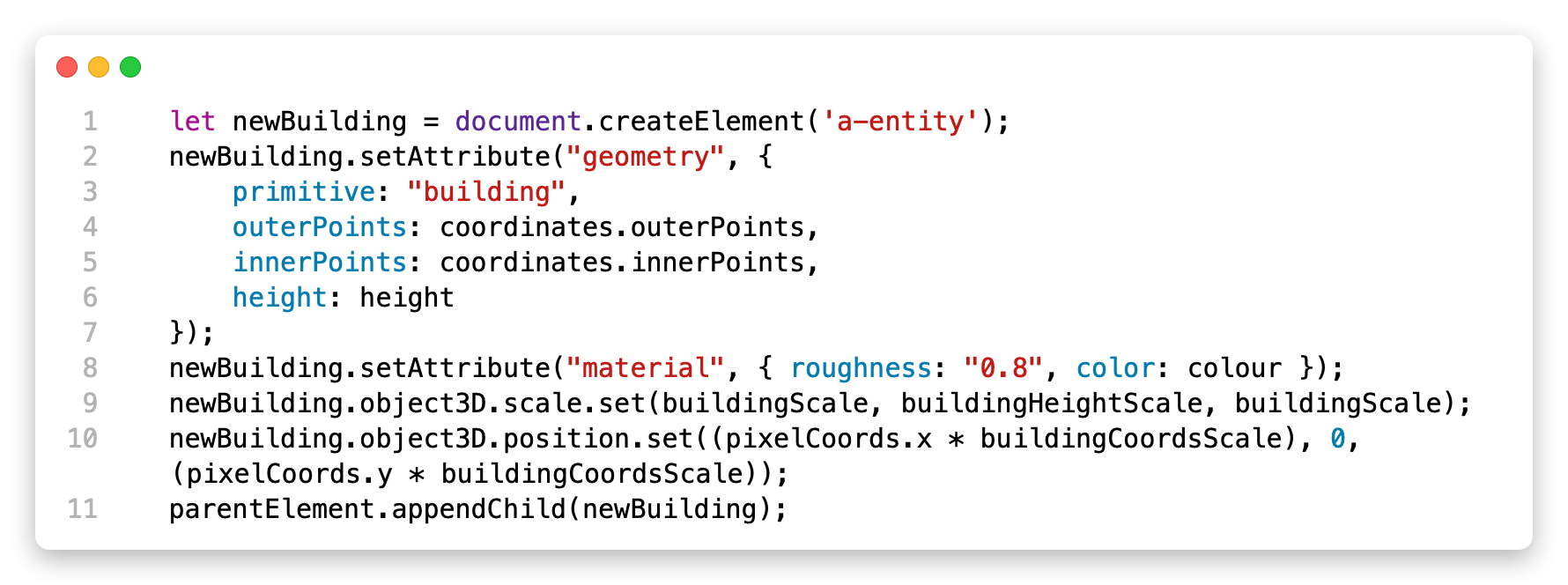


Figure 3-4: Code to add a building entity to the scene. Written by Melchizedek Gray.

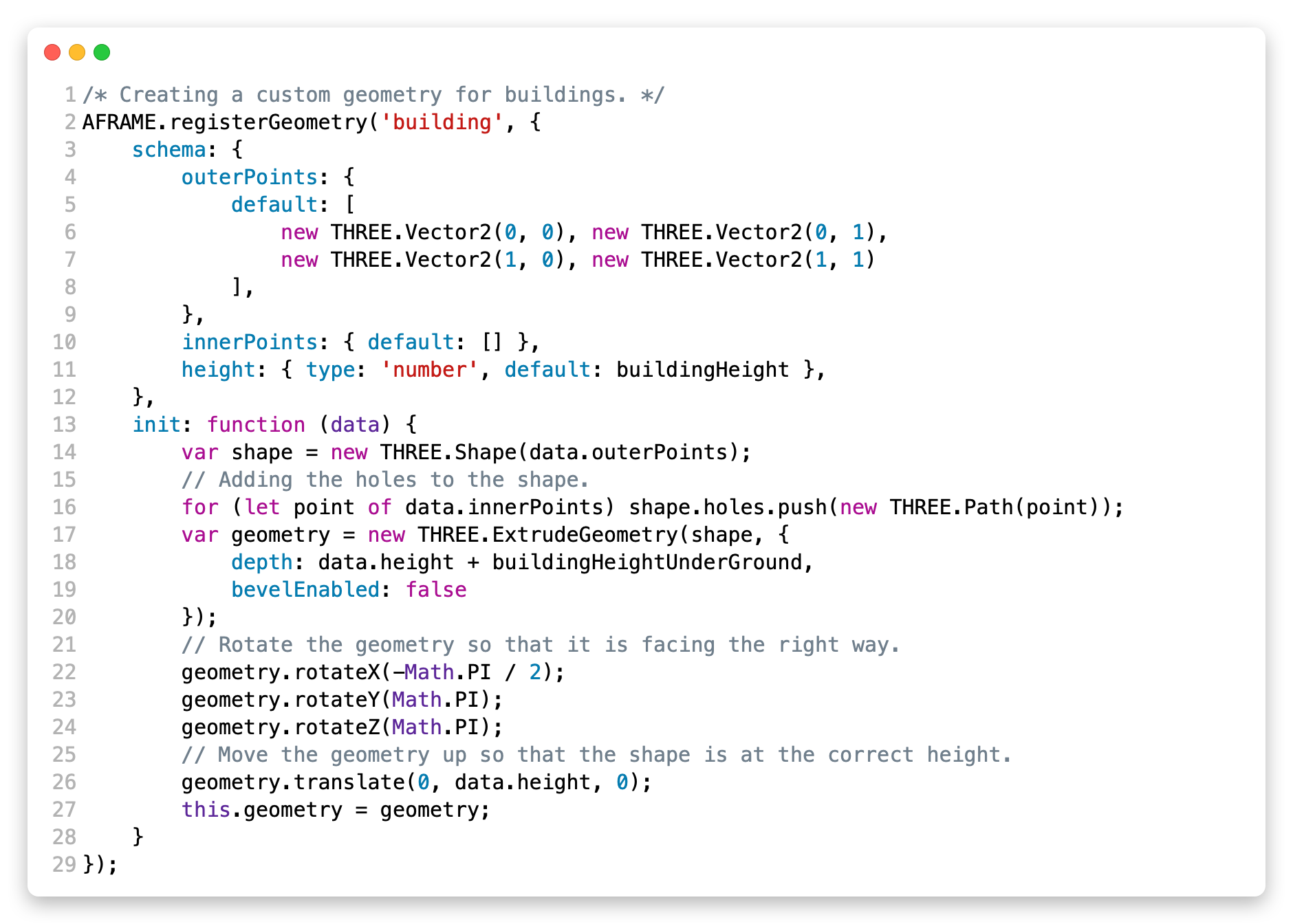


Figure 3-5: Custom building geometry code. Written by Robert Kaiser, adapted by Melchizedek Gray.

To enhance the visual appeal of the 3D world, I incorporated Doug Reeder’s ‘<a-simple-sun-sky>’ entity [13]. Including this entity was extremely simple and immediately improved the look of the sky with the sun and the colour gradient it added. Furthermore, the initial trees I added to the environment were very simply, they consisted of a sphere on the end of a cylinder, but they did not achieve what I was hoping for in terms of their appearance. To remedy this, I integrated two aesthetic low-poly tree models, both created by Render Zing [14] [15]. A-Frame’s native support for the GLTF standard format made the implementation a breeze. To make the world look a bit more complete, I added textures to the roads and to certain paths. I used a dark grey stone pavement texture for the roads and a light grey stone pavement texture for the paths. A-Frame's native support for textures made the implementation process incredibly simple. Overall, these enhancements greatly improved the visual quality of the 3D world and made it feel like a more immersive experience.

3.6 Height Accurate Map

A feature that could be extremely helpful for people navigating using a 3D map is a height-accurate map. I was successful in implementing this feature but, in the end, I decided to remove it as I was unable to make it look good visually and the performance was suffering.

My implementation used a pre-downloaded GeoTIFF file of the Lancaster University campus, which is a format for storing geographic information in a TIFF. To parse the GeoTIFF, I implemented a small library called geotiff.js. This allowed me to read the file and get the heights of every 2m area. The GeoTIFF file was 2500x2500 pixels big, it covered a 5000-metre squared area, and each pixel represented the height of that point in the real world. I then took the grid of heights and connected them using A-Frame triangles. This allowed me to render a height-accurate map of the university campus.

However, as alluded to earlier, there were several issues with my implementation of it. One of the big issues I faced that resulted in the feature being removed was that the ground was clipping through the roads and paths that I had placed. Not finished.

Another issue was the performance hit for drawing so many triangles. Talk about trying to use geometry merging and instanced meshing to solve this problem.

This feature would be an excellent feature to include in a 3D map as it would give users a better understanding of the elevation changes in their path and provide a more accurate representation of the terrain.

# Chapter 4. User Study

With the aim of analysing the effectiveness of a 3D map and navigation system, a user study was conducted on my implementation of such a system. The study involved asking participants to walk to two locations on campus while using the application for one walk and not using it for the other. This approach allowed for a comparison of the routes that the participants took versus the route the application took them. The first place they were instructed to walk to was Barker House Farm located in the Cartmel college area. The second place was Hawkshead which is a residential building in Furness college. For the walk not using the application, if they did not know where either Barker House Farm or Hawkshead is, that walk was skipped and the next walk or the interview was completed.

In total, five participants partook in the study.The**se** participants were selected from the student population at the university via an SCC announcement on the study participation forum and by word of mouth.

## 4.1 Explanation of Methodology for Data Collection

The study was conducted using a mixed-methods approach, incorporating both quantitative and qualitative data collection techniques. In order to collect quantitative data, an Apple Watch was used to record the routes of the walks. This data was helpful to analyse the efficacy of the implemented navigation system. On the other hand, observations of the participants using the application and an interview afterwards were used to collect qualitative data on the user experience. Combining these two methods of data collection allows for a more complete understanding of the system’s effectiveness in the real world.

By asking the participant to think aloud during the walk, it encouraged them to voice their thoughts in real time as they used the application. This allowed for a more human-like interaction as I walked with them, netting the benefit of a more calm, relaxed environment which can elicit more of their opinions. It also helped me to better understand their perspective and thought processes. Any issues or difficulties encountered, or any important statements said were recorded on pen and paper.

After both walks, a semi-structured interview was conducted with each participant to gather more detailed feedback about their experience using the application. The interview was comprised of twelve questions with the option of asking any follow-up questions due to its semi-structured nature. Their answers were transcribed onto paper and then later transcribed onto a password-protected Word Document.

## 4.2 Description of Research Questions

Below are the questions asked and an explanation of why each one was chosen.

1. “What year of study are you in?” – This question was asked to help gauge the participant's familiarity with the campus, as someone who is newer to the campus may have a different experience navigating using the 3D map compared to someone who has been on campus for several years.
2. “Did you find it easy to navigate the webpage?” – This question was asked as an icebreaker conversation and to get initial feedback on the generic functionality of the webpage.
3. “Was the 3D environment helpful in understanding where you were situated in the real world?” – This question was asked to analyse whether the 3D nature of the implementation was effective in increasing the user's situational awareness. It helps assess whether the 3D map added value in terms of the user's understanding of their physical location.
4. “Was the 3D aspect helpful for navigating to the destination?” – This question was asked to gain insight into whether the 3D environment was helpful in navigating. It helps determine whether the 3D map helped users find their way to their destination more easily than a traditional top-down view.
5. “How likely would you be to use a production-quality 3D map application over a traditional top-down view? (Never, rarely, sometimes, often, always)” – This question was asked to understand the participant's overall preference for 3D maps versus traditional top-down views. It helps assess the potential market for a 3D map application.
6. “What was your reasoning for that answer?” – This follow-up question was asked to gain a better understanding of their reasoning for the prior question.
7. “How often did you look at the top-down mini-map in the top-right corner? (Never, rarely, sometimes, often, always)” – This question was asked to gauge the participant's use of the top-down mini-map. It helps to determine whether users still rely on this feature when using a 3D map.
8. “How was the performance on your device? Was it acceptable or too slow?” – This question was asked to gain feedback on the technical performance of the 3D map, including whether it was too slow or caused any technical issues.
9. “Is there anything that comes to mind that could improve the 3D environment?” – This question was asked to obtain suggestions for improving the 3D map, which could be used to inform future development and updates.
10. “Are there any improvements that could be made to the navigation system?” – This question was asked to obtain suggestions for improving the navigation system, which is a crucial part of the application.
11. “Are there any other features or improvements that come to mind to improve the overall design or usability of the website?” – This question was asked to elicit any additional feedback or suggestions for improving the overall user experience of the website.
12. “And finally, is there anything you noticed that you would like to ask about?” – This question was included to give participants the opportunity to bring up any additional feedback or questions that may not have been covered by the previous questions.

## 4.3 Analysis of User Feedback

One of the main benefits of this implementation was highlighted by the fact that, despite all five participants not knowing where Hawkshead is, all three participants that used the application found their way there. The fact that three of the participants involved in the study were in their third year of university, shows that the application can still help find places regardless of one’s familiarity with the campus.

All the users found the UI easy to navigate with no trouble. Two of the participants explained that it was due to there only being a couple of buttons on the screen, and one said it was an intuitive design. Four of the five participants figured out that the mini-map could be made bigger by clicking on it with one participant mentioning that it would be good to “make the bigger mini map the default and then when you tap it, make it full screen.” The person that said this was one of the participants that answered ‘rarely’ to question five and overall, they were not too fond of the 3D aspect of the application, which explains why they would prefer it if they could make the 2D mini-map full screen.

For question three, “Was the 3D environment helpful in understanding where you were situated in the real world?”, two of the participants responded positively, one was on the fence, and the final two responded negatively. One of the positive responses said, “Yes, it’s a lot quicker when you see it in 3D because there are more reference points”. Considering this is one of the main goals of this project this is an excellent response as a 3D environment should help the user understand their situation more effectively than a top-down view. However, one of the participants that negatively responded said “I found the 2D environment easier because it gave me a wider area to look at and be able to orient myself.” These mixed responses suggest that some users may prefer the familiarity and simplicity of a 2D environment, while others appreciate the additional reference points that a 3D environment provides. Giving the user the option to customise their view may be a promising idea for future development.

For question four, “Was the 3D aspect helpful for navigating to the destination?”, four of the five participants responded positively. One participant found that “the shape of the path was helpful in enclosed environments”, while another found that it was “easier to see the direction with more reference points.” However, one participant noted that the 3D aspect could be helpful if the “destination was on a vertical offset, e.g., up or down stairs.” These answers indicate that a 3D environment can be effective at helping a person navigate a route more effectively.

The answers to question five, “How likely would you be to use a production-quality 3D map application over a traditional top-down view? (Never, rarely, sometimes, often, always)” were very split. See figure 4-1. The split in responses suggests that a 3D environment for navigation may not be universally preferred and that the utility of such an application may depend on a person’s preference.

For question six, “What was your reasoning for that answer?” (in reference to question five), the person that responded with “sometimes” to question five said, “In an urban environment, it can be helpful, but in rural places, it is not really. In rural areas, you rarely change paths and 3D navigation is helpful when changing paths a lot.”

## 4.4 Evaluation of the 3D Map Implementation

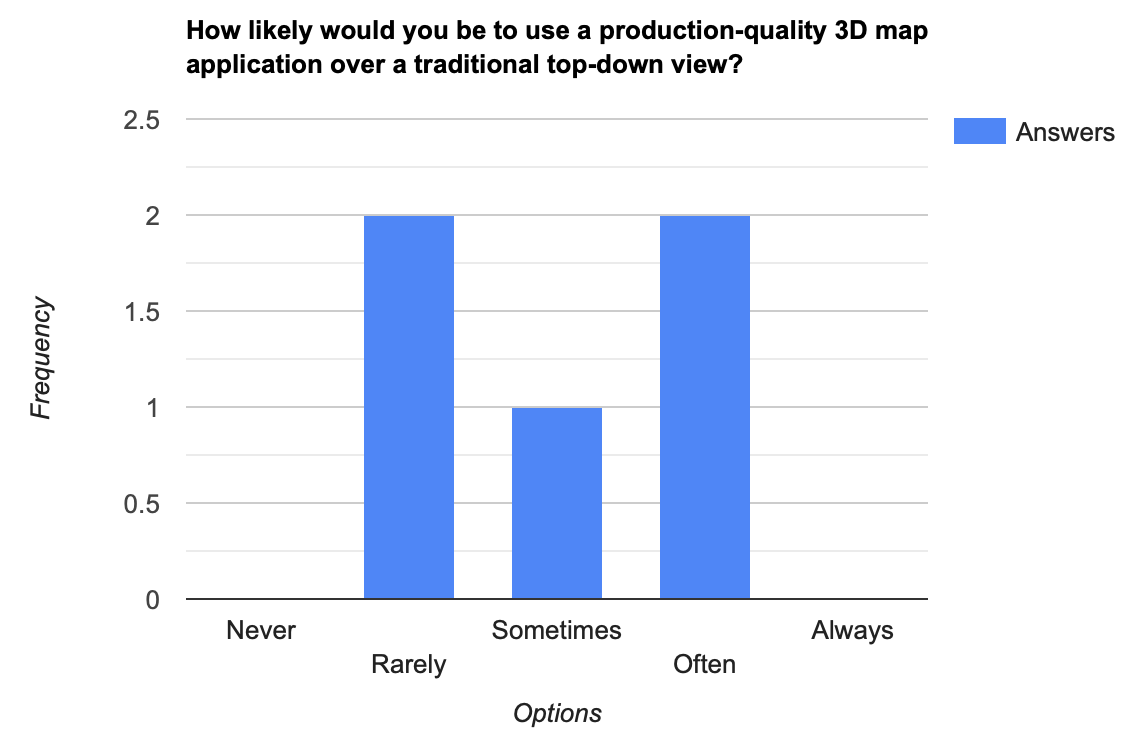


Figure 4-1: Graph of answers to question 5

Overall, the opinions of the participants were a mixed bag. Some preferred the 3D environment and found it genuinely helpful, whereas others less so.

# Chapter 5. Conclusion

## 5.1 Recap of Key Findings and Results

* Here, list the aims from the first chapter and talk about if they have been met.

## 5.2 Discussion of How the Project Could Be Extended

* Talk about the height map.
* Talk about how textures could be added for different structures such as stone vs concrete vs gravel paths.
* Maybe talk about scanned images of the buildings and paths that could be overlayed over the map.

## 5.3 Final Thoughts on This Project

End on a happy note ☺

References

|  |  |
| --- | --- |
| [1] | E. Andrews, “8 Remarkable Early Maps,” 22 August 2018. [Online]. Available: https://www.history.com/news/8-remarkable-early-maps. [Accessed 11 March 2023]. |
| [2] | Mobile Phone Museum, “Benefon ESC!,” [Online]. Available: https://www.mobilephonemuseum.com/phone-detail/esc. [Accessed 10 March 2023]. |
| [3] | “Navigation - Worldwide,” Statista, February 2023. [Online]. Available: https://www-statista-com.ezproxy.lancs.ac.uk/outlook/dmo/app/navigation/worldwide#revenue. [Accessed 11 March 2023]. |
| [4] | E. Reid, “A look back at 15 years of mapping the world,” 06 February 2020. [Online]. Available: https://blog.google/products/maps/look-back-15-years-mapping-world/. [Accessed 11 March 2023]. |
| [5] | D. Nowell, “£24m management school building now open at Lancaster University,” Lancashire Post, 21 May 2021. [Online]. Available: https://www.lep.co.uk/business/ps24m-management-school-building-now-open-at-lancaster-university-3245393. [Accessed 9 March 2023]. |
| [6] | “Most popular mobile apps in the United Kingdom (UK) in September 2022, by reach [Graph],” UKOM, September 2022. [Online]. Available: https://www-statista-com.ezproxy.lancs.ac.uk/statistics/1282098/top-mobile-apps-uk-by-reach/. [Accessed 12 March 2023]. |
| [7] | “Leading mapping apps in the United States in 2022, by downloads [Graph],” AppMagic, 07 February 2023. [Online]. Available: https://www-statista-com.ezproxy.lancs.ac.uk/statistics/865413/most-popular-us-mapping-apps-ranked-by-audience/. [Accessed 12 March 2023]. |
| [8] | Taginfo, “Database statistics,” [Online]. Available: https://taginfo.openstreetmap.org/reports/database\_statistics. [Accessed 27 February 2023]. |
| [9] | A-Frame, “Entity-Component-System,” [Online]. Available: https://aframe.io/docs/1.4.0/introduction/entity-component-system.html. [Accessed 11 March 2023]. |
| [10] | R. Kaiser, “VR Map - OpenStreetMap goes WebVR,” [Online]. Available: https://vrmap.kairo.at. |
| [11] | S. Nishry, “Game Performance: Geometry Instancing,” 25 May 2015. [Online]. Available: https://android-developers.googleblog.com/2015/05/game-performance-geometry-instancing.html. [Accessed 19 March 2023]. |
| [12] | D. Mackenzie, “instanced-mesh,” [Online]. Available: https://github.com/diarmidmackenzie/instanced-mesh. |
| [13] | D. Reeder, “aframe-simple-sun-sky,” [Online]. Available: https://github.com/DougReeder/aframe-simple-sun-sky. |
| [14] | R. Zing, “Low Poly Tree 1,” [Online]. Available: https://sketchfab.com/3d-models/low-poly-tree-1-ad78e0142e0c444391757d6db6101c40. |
| [15] | R. Zing, “Low Poly Tree 2,” [Online]. Available: https://sketchfab.com/3d-models/low-poly-tree-2-73eb8c73bd6c4a8db4e02dbb6bfcffcf. |