FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Accountability in Immersive Content Creation Platforms

Luís Guilherme da Costa Castro Neves

Mestrado em Engenharia Informática e Computação

Supervisor: Daniel Filipe Martins Tavares Mendes

Second Supervisor: Rui Pedro Amaral Rodrigues

March 26, 2023

Accountability in Immersive Content Creation Platforms

Luís Guilherme da Costa Castro Neves

Mestrado em Engenharia Informática e Computação

March 26, 2023

Abstract

Nowadays, 3D content has a wide range of applications, from entertainment such as animation movies and video games to more industrial areas like robotic simulations in factories, architectural representations, and flight training. Accurate and faithful 3D representations of the real world are essential for most use cases. However, the creation of 3D content heavily relies on complex 3D modeling software with a steep learning curve, making it a time-consuming and not easily accessible process for many people. Collaboration among multi-disciplinary teams is crucial in such contexts, requiring access to the proper tools to combine their expertise efficiently.
Virtual Reality (VR) may provide a solution to this problem by facilitating the process of 3D content creation, offering an accurate and immersive representation of models. Although VR has been around for over 40 years, it has only recently become widespread and proven to be an extremely useful tool in many areas such as education, health, and engineering, allowing for more immersive 3D representations and training exercises. With the appearance of better and more affordable devices, we can now not only create 3D representations but also interact with them through an immersive experience that improves our perception of such creations.
However, most collaborative 3D content creation platforms do not consider the concept of accountability, which is a crucial aspect of collaborative tasks, providing a feeling of ownership and responsibility for the outcome of the creation. In this project, we propose the development of an integrated versioning tool for a 3D content creation platform to assist with the collaborative development of 3D content, enabling users to see real-time changes made in the environment. It aims to address the lack of study in terms of accountability in immersive and collaborative content creation platforms, demonstrating how a versioning tool increases users' accountability in the environment.

Keywords: Virtual Reality, Accountability, Collaboration, Content Creation

Acknowledgements

I would first like to thank my supervisors, Professor Daniel Filipe Martins Tavares Mendes and Professor Rui Pedro Amaral Rodrigues from the Faculty of Engineering of the University of Porto, for the tireless guidance and support given to me. Without their help, I would have given up and wouldn't have striven so much in trying to finish a thesis and project of which I could be proud.

I would also like to thank the Faculty of Engineering of the University of Porto, not only for giving me the tools needed to complete my thesis but also for giving me the tools to complete this stage of my life. Acting as a second home for me in what seems to have been my whole life, it helped me become who I am today.

I would also like to thank Tuna de Engenharia da Universidade do Porto for being a second family to me. I have made so many close friends there who have been with me through so much and lived many adventures that I won't forget. It gave me a lot while asking for so little.

I would like to thank Sky for never giving up, and for his Stoic attitude toward everything. Truly an inspiration that helped me through these trialing times.

I would like to thank most dearly my friends, namely João Lemos, Sérgio Pinto, Ricardo Figueiredo, João Manarte, and João Trigueiros. Without you, I wouldn't be half the person I am and would have given up a long time ago. How any of you stuck with me for so long is a true mystery. Thank you.

To my parents, Luís Castro Neves and Ana Rosa Costa, and my sister, Ana Mafalda Neves, no words are enough to describe my gratitude. Through highs and lows, throw thick and thin, you have stuck with me, always helping me be better. Even when the going got really tough. I know I can never repay you, but just want to say, "see you in Japan".

Finally, and not least, I would like to thank the Venus of Willendhorf. She started all this content creation trend, and if it weren't for her, I wouldn't have a thesis.

Luís Guilherme Neves

"It is wrong and immoral to seek to escape
the consequences of one's acts."

Mahatma Gandhi

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Social interactions in face-to-face settings tend to follow established norms due to a sense of accountability and awareness of potential consequences. However, when communicating through digital platforms, individuals may behave differently and not always adhere to the same social rules as in-person interactions. A variety of factors such as the lack of visibility, when the system doesn't show every user and its actions, and asynchronous editing allow for users to make actions without having direct consequences. Erickson et al. [16] present the concept of accountability as the consequences of users' actions. When designing a platform that supports collaboration, it is extremely important to have accountability in mind, so that the experience is beneficial for all who are involved.

1.1 Context and Motivation

3D content, although in high demand, has a arduous creation process that most people avoid. In many cases this happens because most of the solutions in the market for 3D content creation are extremely complex, with a steep learning curve, such as Autodesk MAYA and Cinema 4D.

Not only are the tools hard to understand, having concepts that are foreign to most people, but it is also usually hard to get a grasp on how various aspects of the content are turning out. An example of this is the perception of the ratios, many times because moving in the creation environment is not as straightforward as walking in the physical world. The fact that the creation is perceived through a screen hinders the perceived immersion and might create a detachment to the creation process, which might jeopardize the creative process.

Virtual Reality(VR) has the power to mitigate these negative aspects, by providing an immersive and more life-like visualization of the environment. This allows creators to be more in touch with their creations, allowing them to move around more freely than with the typical mouse and keyboard approach. This freedom allows to perceive different perspectives easily, which is useful to get the overall feel and look of the creation.

When creating content, it is many times the case that more than one person is working on the same project. As such, when designing a platform for content creation, one should consider how to facilitate the collaboration of a team. When looking at collaboration in the real world, most interactions are based on visual and audio cues. When a person needs to convey some information about a certain aspect to one of the collaborators, he usually approaches the subject in question, uses a visual cue to bring his attention to the aspect he is referring to, and then communicates his idea by using a form of communication known to all interested parties. In order to make collaboration a seamless experience, we need to translate these techniques to the platform. The users should have a means of communication that can be understood by anyone involved, they should be able to see a representation of each other and, preferably, their posture, and they should all have visibility to the same content.

When people collaborate, there is always the possibility of people interfering with each others' jobs, either inadvertently such as due to technical issues, or by design. It is many times the awareness that the user will be held accountable for his actions that makes him refrain from ruining others' work, either by accident or design. Not only does it keep people from doing actions, but it also impels them to do the ones they should, as people will be held responsible if they do not deliver on something they are supposed to. As such, accountability is a very important aspect of collaboration, one that goes many times ignored.

1.2 Research Questions

As mentioned previously, accountability is an important aspect of collaborative environments. This dissertation focuses on just that, increasing accountability, more specifically on immersive environments. For this purpose, the following research questions were defined to help accomplish this goal.

Research Question 1 (RQ1) - How can we increase the accountability of users in an immersive and collaborative environment?

Research Question 2 (RQ2) - How can we preserve the immersion of the environment while increasing accountability?

As these research questions will be referenced multiple times throughout this dissertation, they will from here on out referred to as RQ1 and RQ2 respectively.

1.3 Objectives

As previously mentioned, when creating 3D content, it is normal to have many people working together to achieve a common goal. This ensues in people making changes to the same environment, leading to the need for accountability. For this purpose, the effects of a versioning tool will be studied in regard to the increase in the accountability of users. Some important features that a versioning tool should have are:

- The ability to see past changes, which relates to the term of awareness;

- The ability to see who has made the changes, which pertains to the term of accountability;
- The ability to see pertinent information about the changes, such as date, users involved (as in the case of the changes being subject to acceptance), and state before the change;

Another thing that must be kept in mind is that every user must have knowledge of the full set of actions it has available, and also access to pertinent information.

1.4 Document Structure

The remaining of this document is divided into: State of The Art, the Immersive 3D Versioning, the Prototype, the Evaluation and the Conclusion and Future Work.

In the State of The Art is presented some previous work on the area that will be used as a basis. It will focus mostly on other platforms for the creation of 3D content and give special emphasis on the ones that allow collaboration and/or use Extended Reality(XR) technologies. It also presents some of what was found in relation to the concept of accountability.

Chapter Immersive 3D Versioning presents the proposed solution that will be studied. It starts by defining the approach that was taken to achieve the solution, then presents the requirements, and ends with the solution architecture.

Prototype starts by detailing the tools and technologies that are used in the development of the prototype. Afterward, a detailed description of each implemented feature is provided alongside the challenges faced during development. The chapter closes with a description of the development environment, the hardware used, and the input solution used to interact with the virtual environment.

In the Evaluation the testing phase is detailed. It starts with a description of the tests and how they were executed, followed by an analysis of the results to validate the implemented solution.

In the Conclusion and Future Work is presented a summary of what is in this document, while also describing the results of implementing the versioning tool.

Chapter 2

State of The Art

In regards to 3D content creation, especially in immersive environments, there is a lack of focus on accountability, which is a very important aspect when talking about collaboration. In this chapter are presented some key concepts important in the scope of the research and some papers and projects on the topics.

The first concept to be analyzed is Content Creation. This will be tied in with 3D Content and its advantages.

Still in the context of content creation, a subsection will be dedicated to Virtual Reality and Augmented Reality, and how it can help in the visualization and creation of 3D content.

After this, the concept of Collaboration will be presented, and why it is important for content creation.

The last part of this chapter will tie all the concepts together and present a taxonomy table for the main related papers and projects.

2.1 Key Concepts

Before analyzing some of the research related to this dissertation, it is important to provide the reader with some key concepts on the subject. Below is a brief description of them.

2.1.1 Content Creation

Content creation is the process of concretizing an idea. The author of [36] identifies five steps in the process of producing an idea, as shown in image 2.1:

1. Gather raw materials;
2. Digest the material;
3. Incubation stage;

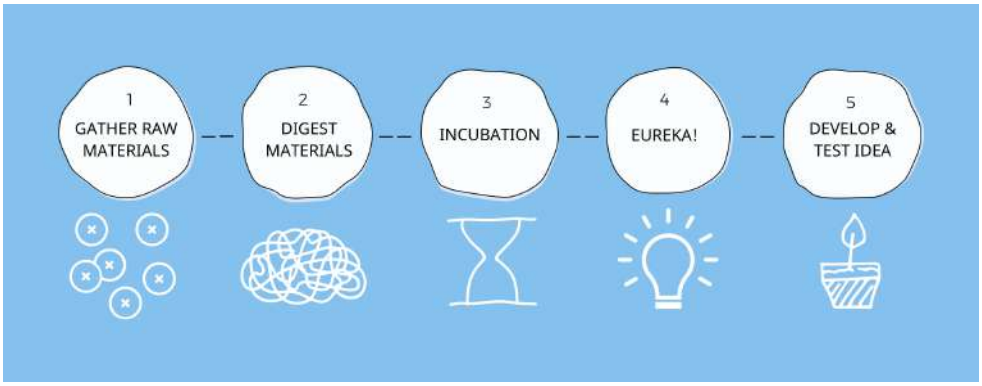

Figure 2.1: Five steps of producing an idea

4. The appearance of the idea and finally;

5. The development of the idea;

Although the whole process is important, the final stage, the development of the idea, is where the content is created.

Nowadays, most of the content consumed is on digital screens, such as computers, mobiles, and TVs. As such, most of the content created is directed to these mediums and also made on most of them.

2.1.1.1 3D Content

3D content can provide a representation closer to real life, allowing the user to have a better perception of it. This type of representation can greatly help in the process of content creation.

3D content can be represented in many ways, being the following methods the most used [18]:

- Polygons, which are 3D points that define faces of objects
- Curves, comprised of weighted points, making up splines, NURBS, and Bézier surfaces
- Implicit Surfaces, which are the result of mathematical formulas
- Constructive Solid Geometry, or CSG, are objects created out of other primitive objects such as spheres and cubes
- Voxels which is similar to a pixel, but in 3D
- Displacement Surfaces which uses 2D textures as height maps

In [18] 3D modeling is defined as the process of digitally representing an object with three dimensions.

Common 3D modeling techniques are:

- Polygonal Modeling where users edit the points of the polygons directly

Figure 2.2: Google's Starline Project

- Digital Sculpting where the users model with brushes (Ontlus [14], TiltBrush...)
- 3D Scanning and Reconstruction where objects are replicated using photogrammetry (Project Starline [9], as seen in image 2.2)
- Procedural Modeling where users use rules and mathematical formulas to generate content
- Kitbash Modeling where users mesh different models together
- Parametric Modeling where the model is defined by a set of operations that can be changed at any given time

2.1.1.2 Virtual Reality and Augmented Reality

In [15] VR is defined as a computer-generated digital environment that can be experienced and interacted with as if that environment were real, and [29] defined Augmented Reality as a technology that overlays computer graphics with the real world, which is a part of Mixed Reality(MR), defined in [21] as a blend of the physical and real world, where items of the virtual environment overlay the real environment or items of the real environment overlay the virtual world. These technologies comprise what is called Extended Reality, or XR. The term XR was coined by Nick Whiting, the X being a placeholder to symbolize AR, MR, and VR, as mentioned in [10] and not symbolizing extended.

These technologies present obvious opportunities in 3D content creation, providing a more immersive and realistic visualization of the content. A number of platforms already exist for this purpose, such as LeMo [24], CEPVR [7], or VR Designer [18].

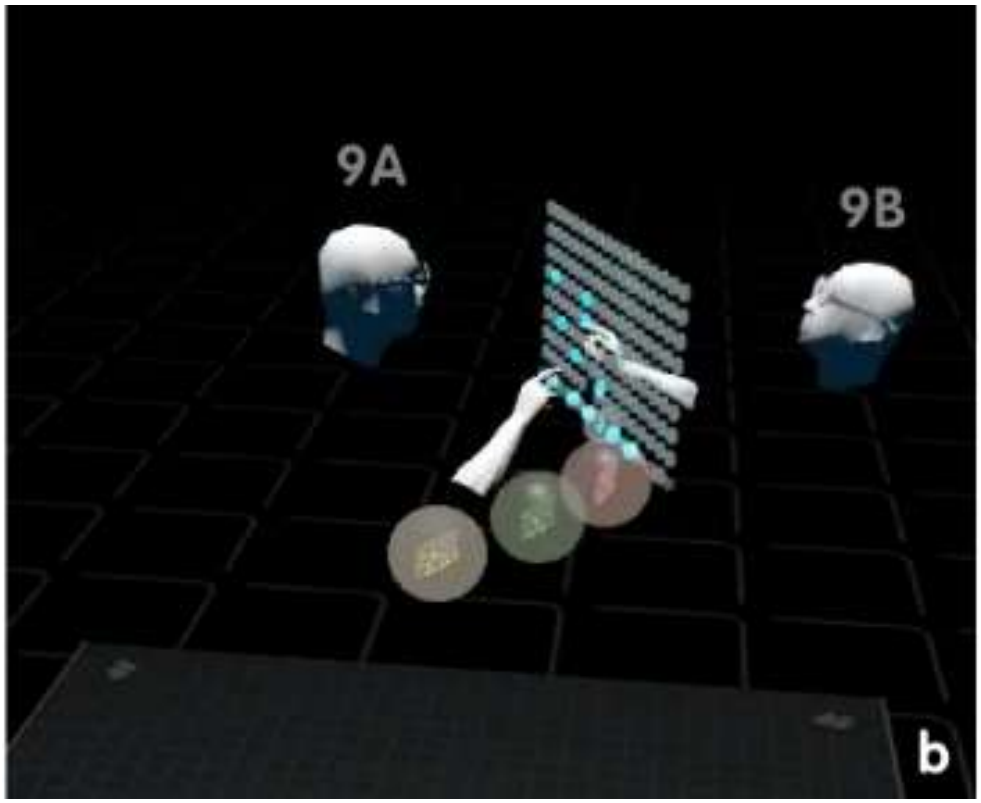

Figure 2.3: Musicians creating music together inside a Virtual Environment

2.1.2 Collaboration

Collaboration can be defined as the process of two or more entities working together in order to achieve a common goal.

As mentioned previously, teams of people collaborate in order to achieve a common goal or even to achieve better results. [25] shows that students have better results when performing group work than when working alone. Musicians feel it's easier to find inspiration when collaborating with other users, as seen in [24]. Image 2.3 shows two musicians writing a song together inside the platform LeMo.

Collaborative mechanisms can either be passive or explicit forms of communication, where the passive form pertains more to the posture of the user rather than specific actions, and the explicit is the opposite, the users having to perform explicit actions in order to communicate, such as talking. [27]

When designing collaborative techniques, there are some important concepts to keep in mind.

2.1.2.1 Awareness

Erickson et. al [16] describes awareness as the knowledge of the shared space and its state, bringing social rules which govern our actions into play. In other words, it is the ability to know how our surroundings are, even if we don't necessarily see it

2.1.2.2 Visibility

Visibility, according to [16], pertains to the ability to see socially significant information. This means that a user must be able to see other users and what actions they take, as well as be able to communicate with them.

2.1.2.3 Accountability

Accountability, according to [16], is described as the fact that we will be held responsible for our actions. As such, we will refrain from doing something that is deemed wrong.

2.1.2.4 Ownership

Ownership pertains to the belonging of an environment or model, i.e., described to whom the model belongs. This is a characteristic of the environment/model and can be used as a way to keep users without permission to edit things they shouldn't.

2.1.2.5 Affordances

Affordances, according to [20], are properties of an object that give strong clues of how to use it, and, as such, when taken advantage of, provide the user with the knowledge of how to use it just by looking.

2.1.2.6 Proxemics

Proxemics is the usage of the space around the subject as a means of communication.In [31] the concept of remote proxemics is presented as the usage of virtual space as a form of communication.

2.1.2.7 Concurrency

Concurrency is the act of multiple actions happening at the same time. This can lead to race conditions, which happen when multiple users try to access/edit the same object at the same time. It is crucial to think of how to handle these problems when creating collaborative techniques, as many users will be working simultaneously in the same environment.

2.2 Related Work

Although there is a multitude of platforms that allow for immersive content creation, such as the Ontlus [14] and the CEPVR [7], little focus on the concepts of accountability and ownership, much less have a versioning feature, most focusing on the content creation process and object representation.

Version control is the practice of tracking and managing changes to files [5] [12]. One very widely adopted version control tool is Git. It was initially created in 2005 by Linux creator Linus Torvalds to help the developers who were working on the kernel share and control different

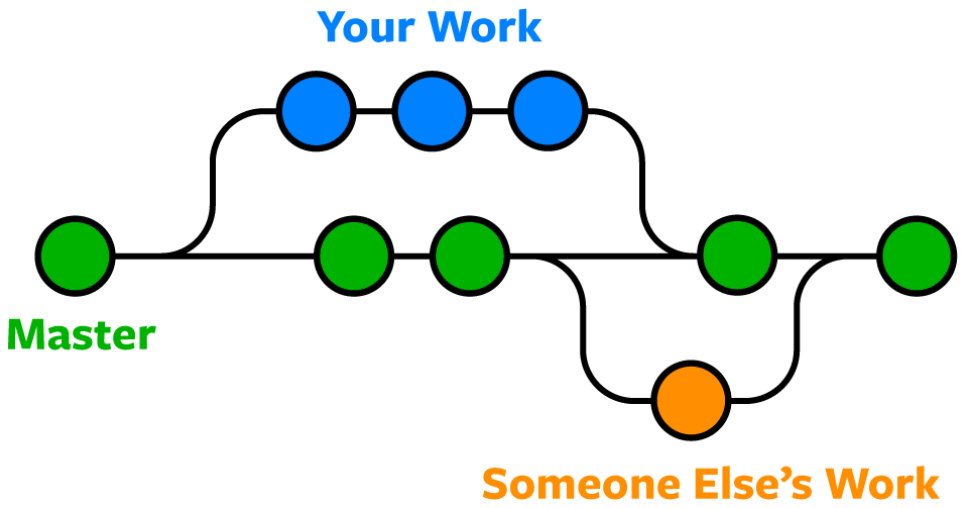

Figure 2.4: Illustration of multiple users working simultaneously on Git

versions while working simultaneously on the same project [6] [12], as seen in figure 2.4, but has since evolved to handle not only code but also different files such as 3D file types [30]. Among other features, Git allows users to see when a change was made, who made the change, and also see what was changed, showing the differences from the previous version. It also allows reverting changes to a previous state.

Another famous tool that allows for version control is the Google Docs suite [4] [2], which, amongst other features, allows collaborating users to make changes or suggestions to the same document simultaneously, as well as see version history and activity for those documents, as illustrated in figure 2.5, being these features limited to text documents, spreadsheets, slide presentations and code documents.

Figure 2.5: Google Docs Usage Example

Unlike Git and Google Docs, 3DRepo is a solution created specifically for the versioning of 3D models. It allows for multiple users to collaborate and communicate in the same 3D project, see differences to the models, as shown in figure 2.6, and track issues, among other features [1]. It follows a similar approach to version control as Git and is widely adopted amongst 3D designers.

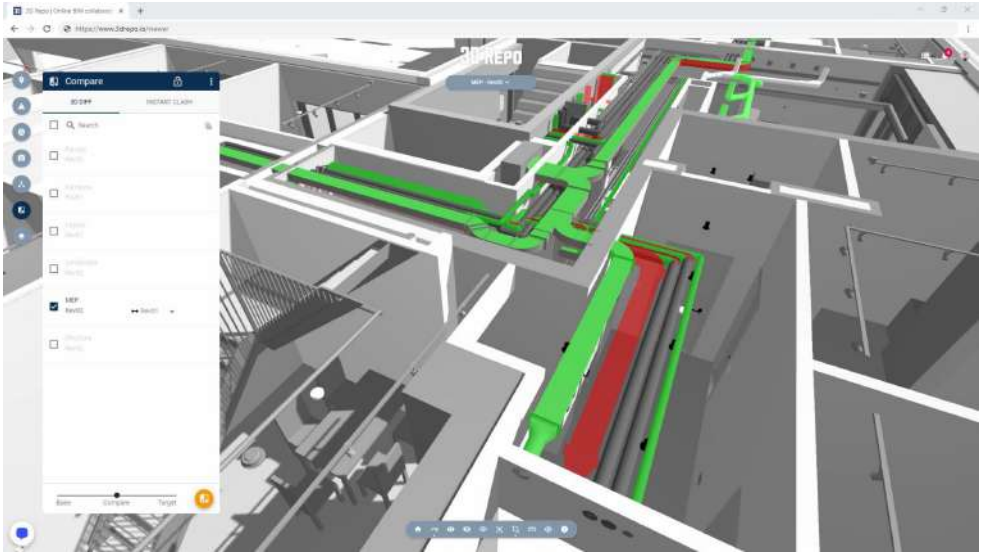

Figure 2.6: 3D Repo differences view

One of the struggles to overcome when designing a version control system for graphics applications is the memory constraint, as storing many versions of the file can lead to large memory occupancy [11]. Calefato et al. [11] present a Directed Acyclic Graph (DAG) approach for collaborative image manipulation software. In this approach, it is proposed that on binary files, such as images, videos, and meshes, instead of storing the whole binary file for each action taken, which is the most predominant approach and uses more space, store the binary file only in certain milestones dictated by the user of the system, and instead store the actions made to the file. In the end, a working prototype was developed with basic image editing actions in order to be used for testing the proposed solution, having overall positive results. The users who tested the solution found the system easy to use and intuitive and considered this a viable solution for the problem presented, decreasing the storage size needed.

Wang et al. [34] also used a DAG approach for a version control system, but for a 3D environment. They identified two types of differences, structure, which is caused by scene editing operations, and state which is caused by property modifications like transform, name, etc. A scene can be defined by a graph, where each element is a node in the graph. They propose storing the changes made to each node, state-wise, and structure-wise, which allows for the comparison of the differences only on the changed nodes, instead of the whole scene. In the developed system, which sits atop the Three.js Editor, four different actions are available to the user, commit the changes made, retrieve a version of the scene, tag, and branch, and finally diff and merge. Two algorithms were developed to aid with the merging and differentiating of versions. The first algorithm was for automated differencing, allowing to automatically find the difference between two versions, and the second was for interactive merging, which allows a user to merge two versions while handling conflicts. Although a couple of further improvements were presented by the authors, like the reparenting of nodes and the SHA encryption of the objects to decrease the required storage size, the

prototype was successful in solving the presented problem by presenting a way to store changes made to a 3D scene.

Ke et al. [22] also present a Graph-based open database for 3D content with versioning, which allows for the accumulation of models from a large number of contributors, making the memory scaling lower than storing the whole model many times. It tackles this issue by having a model database and a scene database. The first one stores the information that makes up a model, like the points of the model, its normals, mesh, etc. The second database stores the reference to the primitive objects in the first database.

Özkil [26] focuses his paper on how concurrent design evolves, and the needs of the users when creating content concurrently. One of the main takeaways of his case study was that versioning is a critical need in collaborative content creation, namely that of 3D content creation for 3D printing, helping improve the organization and quality of the designs. In his paper, Özkil used data from a platform called Thingiverse, which is a platform that allows for the sharing of designs, which hosts one of the largest design repositories, used by a very big and active community. It also implements the concept of derivatives, which is the process of creating a design that is derived from another one previously uploaded to the platform. The author also mentions that the need for versioning is so important in this field that the users would improvise a versioning tool by creating derivatives, i.e. new designs derived from old ones, of their own designs just to keep a history of the process, so that they could, at a later stage, pick up the same design from a previous point in time.

Eriksson [17] presents the need, obstacles, and recommendations for the creation of a versioning tool for 3D city models, being the main obstacles related to data gathering, which is not very relevant to this dissertation, and the database to store the information, which needs to be accessible to all interested parties and must be accessible between layers. A table with versioning information present in different city model platforms is also presented.

VR has brought numerous opportunities in regard to content creation. One such example is LèMō [24]. LèMō, or *Let's Move*, is a VR platform that allows multiple users to collaborate while creating music in a virtual environment, as seen in figure 2.3. In it, users have both personal and shared spaces where they can create music. In the paper the authors studied the impact of having those spaces, arriving at the conclusion that having personal spaces were beneficial for the creators.

The Nvidia Holodeck [3] is a VR collaborative 3D content creation platform created by the Nvidia corporation, with the aim of allowing creators to edit their creations from inside the Virtual Environment while collaborating with users from the other side of the world, as seen in figure 2.7.

The Ontlus [14] is a collaborative VR content creation platform that was created to facilitate the job of creating 3D models, putting the designer inside the environment. In it users can draw on the 3D space with the controller acting as a brush, changing the values of voxels in the environment.

Ferreira et al. [18] developed a tool called VR Designer on top of the PAINTER project, of which this dissertation is a part. VR Designer is a 3D scene creation tool from inside VR, which besides the normal editing tools also offers a comprehensive procedural tool and speech control

Figure 2.7: Users collaborating in the Holodeck platform

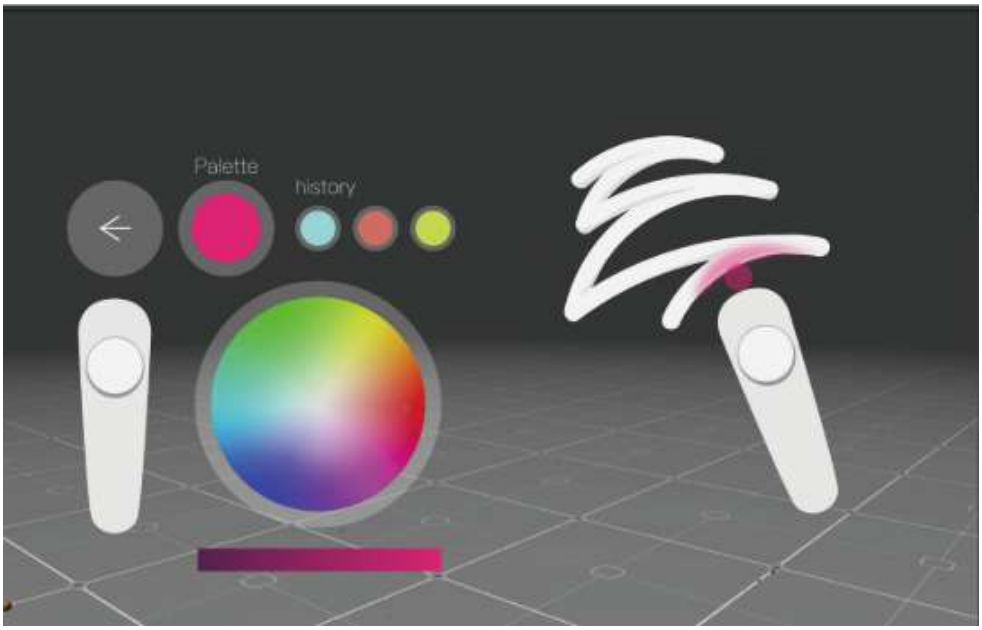

Figure 2.8: User interface on the Ontlus

in order to expedite the process of creating the scene, removing the need to make repetitive tasks such as instantiating many of the same objects, allowing for the user to add one through a search on an online repository, using the user's voice as input, and then applying procedural modifiers to make several copies of the object, as seen in figure 2.9.

Another such platform for content creation is Sur.faced.io [13], which allows users to create face filters with pen and paper and see them in real-time through Augmented Reality, as seen in figure 2.10.

Figure 2.9: Using speech to search for objects online in VR Designer

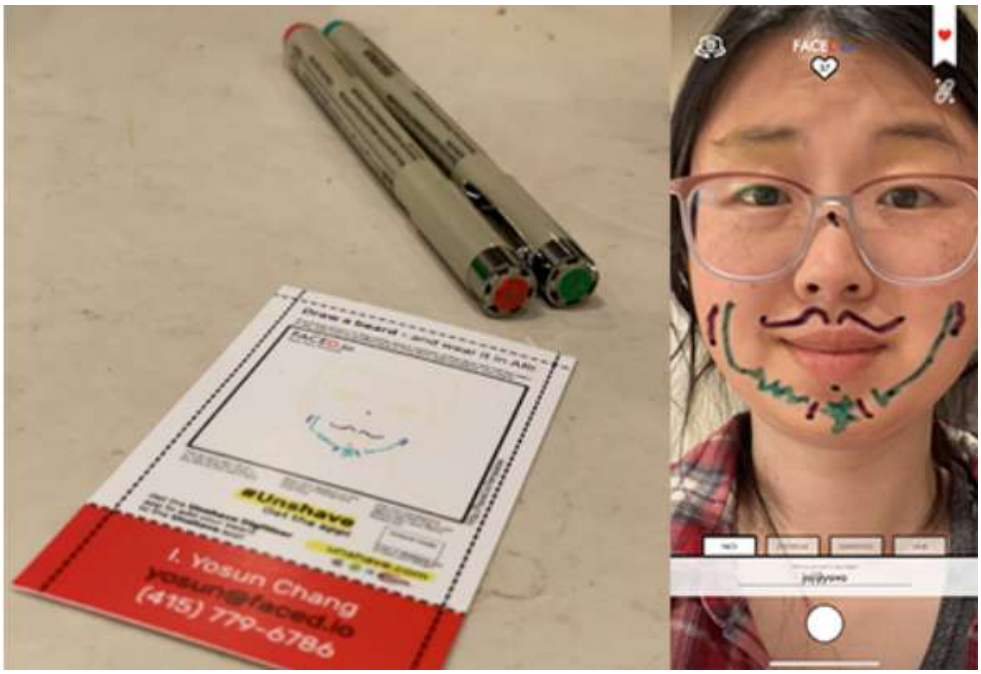

Figure 2.10: Sur.faced.io example usage

Sang-Gyun An et al. [7] presented a platform that allows users to draw car components with a tablet, and other users could view them appearing in real-time in VR. In this platform, one user would be designing on a drawing tablet the components of the car to be tested, while another would be trying them on the VR headset. Another user would simultaneously be using VR and AR to provide a more realistic feel of the environment, such as vibrations and movement, and could also create basic shapes with hand gestures to complement the design made on the tablet. In figure 2.11 can be seen users testing the platform

In the 360theater [32] paper, the authors presented the possibility of using the wizard of Oz for an immersive VR experience. Just like in the Wizard of Oz, a user with the role of wizard implements the actions the user requested in real-time, giving the feeling of a high-fidelity prototype. Although ever-present and omniscient, the wizard is never seen by the VR or AR user.

Figure 2.11: CEPVR demo session

Tang et al. [33] conducted a study that shows how MR can be useful for teaching in higher education on more practical courses such as engineering, allowing for better visualization from the students. One example is in aeronautical engineering where students can see the turbofan in 3D (figure 2.12).

Figure 2.12: Student viewing turbofan with the hololens

Schott et al. [28] also presented evidence that collaborative VR and AR can be beneficial in the education panorama, specifically on medical students, showing many samples of 3D modeled liv-

ers for the users to interact with, while learning the introduction to kidney surgery. The developed platform can be seen in figure 2.13

Figure 2.13: Overview of the platform developed by Schott et al. 2021

Baron [8] created a platform called ColCo (CollaborativeConstraints) focusing on manipulation techniques in collaborative environments. In the solution developed, users would define an axis to constrain the manipulations of the object and then apply modifications to it. This would result in all operations happening on a single axis, making the software easier to use.

Girard et al. [19] conducted a study to attain the advantages of using a collaborative approach for area selection on a VR 3D modeling software. It was concluded that although the times between a collaborative and individual approach were very close, the collaborative approach winning by a small margin, the collaborative approach reduces the distance each user needs to travel and improves the workload distribution.

Miyata et al. [25] developed an educational framework focusing on the idea that group interaction improves collaboration skills and overall results. The results showed that group work in VR is an efficient training framework for graduate students.

Makris et al. [23] studied the possibility of using a semantic approach for designing inside VR, i.e., using an XML file. The user can make changes to the environment from inside, as illustrated in figure 2.14.

Weissker et al. [35] conducted a study to find ways to improve group navigation in a Virtual Environment. These navigation techniques should be comprehensible for all users involved, both the guide and the attendees, should not result in collisions with obstacles, and allow for the creation of spatial dispositions that highlight the objects of interest. A preview of the work done can be seen in figure 2.15.

2.3 Discussion

The previous sections presented key concepts to take into account when developing collaborative systems and introduced some of the projects that closely align with the topic of this dissertation. This section will expand on the previous one, making a more in-depth analysis of the related work, relating them to the concepts previously presented. These concepts are collaboration, content creation, visualization method (whether immersive such as VR and AR or not), visibility, awareness accountability, and lastly ownership.

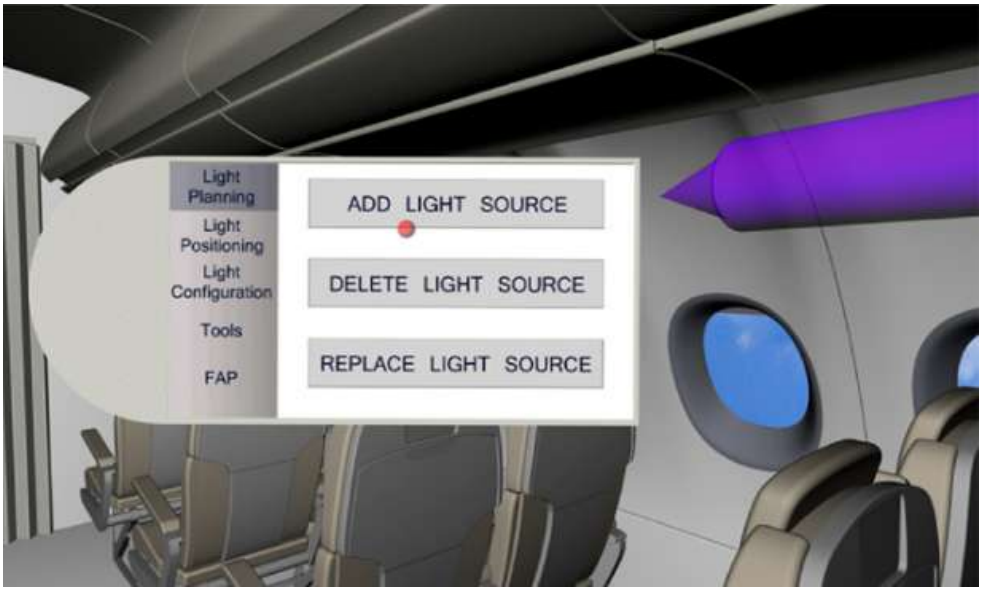

Figure 2.14: Adding a light source through VR

Figure 2.15: Group navigation on VR tours, with the preview of the positions after moving and resulting movement

The table 2.1 at the end of this chapter presents a taxonomy for each of the projects, giving a perspective on the most approached subjects and where there is a bigger lack of research.

In the LeMo [24] platform, users have access to two types of spaces in the Virtual Environment, personal or shared, where they can create music individually on the first case and collaboratively on the second. All users can see each other and all actions taken, even in private spaces, having a constant awareness of the overall state of the environment, although the awareness isn't directly referenced in the paper. It also has a voice communication feature, allowing for better collaboration among users. Since only the owner of the personal space can make changes there, the ownership comes into direct play. Although users will be held accountable for ruining the music in the shared space, their actions are not stored anywhere, and as such users might act without accountability.

In CEPVR [7] users collaborate to design car interiors. One user is tasked with designing components in real-time on a drawing tablet, another user would be testing on the VR side, with another one using AR and VR to give the feel of the environment, such as movement and vibrations, and create basic shapes with hand gestures. Although there are multiple users interacting simultaneously with the same environment, the editors are invisible to the user testing the environment, and the user on the tablet isn't seen by any other user. This lack of visibility incurs a lack of accountability, as the users testing will not know who was designing and as such, won't be able to hold them accountable for their actions.

The 360theater [32] is a VR content creation platform that follows a wizard of Oz metaphor, where a user with the role of wizard implements the actions the user requested in real-time, giving the feeling of a high-fidelity prototype. The wizard, as in the wizard of Oz, although ever-present, is not seen by the user throughout his experience. As such, the wizard has no accountability.

The paper from Tang et al. [33] focuses on the advantages of using VR or AR in higher education, to help students better visualize the contents. This solution doesn't allow users to collaborate from within it. As there is no collaboration or content creation, the other concepts aren't relevant in the context.

Collaborative Constraints, or ColCo [8], a VR platform that allows for content creation, allows two users to collaborate in content creation. However, it doesn't address any of the other concepts.

Miyata et al. [25] developed a framework where multiple users can create VR applications from inside of it. Users can see each other at all times, but the other topics aren't mentioned in the paper.

The Ontlus [14] platform is a 3D modeling tool in VR that allows users to collaborate on the same project. All users can see one another and are aware of the current state of the environment, but like the other VR content creation platforms, it doesn't address accountability directly.

VR Designer [18] is a VR and AR tool that allows multiple users to create content inside the Virtual Environment. Although the users are aware of the state of the environment and can see all other users that are also using it, there is no reference to the concepts on the paper.

The paper from Schott et al. [28] allows users in both VR and AR to study different liver models simultaneously, with representations of each user in the environment. This system however

does not support content creation and doesn't directly address any of the other concepts.

The paper from Weissker et al. [35] focuses on improving the awareness of guided tours inside Virtual Reality, allowing for multiple users simultaneously on the system.

In the paper from Girard et al. [19], users can create content inside of VR. The authors focus especially on the awareness of the users but don't mention the other concepts.

The Nvidia Holodeck [3] allows users to create models inside of VR. In it, users can collaborate and communicate with one another, as well as see who is involved, adding to the awareness and visibility. It also doesn't have any accountability measures in place.

Git [12] is a versioning tool that allows multiple users to work on the same project at the same time, but it does not allow the users to see the changes that other users make in real-time. It does, however, store the information about who made the changes, and when they were made, allows to revert the changes, and only allows certain users to make changes, amongst other features. It does not however use immersive technologies like VR.

The Google Docs Suite [2] allows multiple users to edit the same file at the same time, showing where each user is making changes in real-time, storing who made changes and when and also allowing to revert the state. It also does not use immersive technologies such as VR.

Calefato et al. [11] presented an image editing tool that allows versioning, being able to share with others, and seeing the changes made. However, it does not allow for the simultaneous collaboration of users, just as Git does, and doesn't use VR.

Wang et al. [34] and Ke et al. [22] created a versioning tool for 3D content, which does not support VR.

3D Repo [1] is another versioning tool for 3D content that allows tracking changes made to the project, while at the same time, allowing for the simultaneous edition of it. Just like the other versioning tools, it does not use VR.

The main takeaway from this analysis is the lack of research on accountability in immersive technologies. Although there are numerous solutions for this problem on normal media, such as image and 3D editing on screens, as is the case of 3DRepo [1] and the solution presented by Calefato et al. [11], the impact of such solutions aren't studied in the Virtual or Augmented environments. As such, this dissertation will focus on this topic, by developing a versioning tool that works inside the virtual environment, cutting the need for external versioning tools.

Table 2.1: Taxonomy Table

Project	Collaboration	Content Creation	Visualization Method		Visibility	Awareness	Accountability	Ownership
			VR	AR				
LeMo [24]	○	○	○		○			○
CEPVR [7]	○	○	○	○	○			
360theater [32]	○	○	○	○	○			
MR in higher education [33]			○	○	○	○		
ColCo [8]	○	○	○					
Educational framework [25]	○	○	○					
Semantic-based taxonomy [23]		○	○					
Ontlus [14]	○	○	○		○	○		
VR Designer [18]	○	○	○	○				
Liver Anatomy Education [28]	○		○	○	○	○		
Guided Tours in VR [35]	○		○		○	○		
Collaborative Polygonal Modelling [19]	○	○	○			○		
Holodeck [3]	○	○	○					
Git [12]	○	○			○	○	○	○
Google Docs [2]	○	○			○	○	○	○
3D Repo [1]	○	○			○	○	○	○
Revision Control System [11]	○	○			○	○	○	○
Non-linear Revision Control [34]	○	○			○	○	○	○
Graph-based Open Database [22]	○	○			○	○	○	○

Chapter 3

Immersive 3D Versioning

The previous chapter presented the current state of the art regarding 3D content creation in immersive environments, as well as some research on versioning tools. The main conclusion that arose from this analysis is that there is a lack of research on accountability for immersive environments.

This dissertation centers on enhancing user accountability within an immersive and collaborative environment. Specifically, it explores the impact of a versioning tool on accountability and lays the groundwork for further research and development in this area. Ultimately, this work aims to enhance the content creation experience within the VR environment.

It is expected that with a versioning tool, not only will the productivity improve as every action is monitored, showing the owner of the actions, but also the freedom and creativity of the process, as previous states aren't completely lost, being stored in some form in the versioning tool.

3.1 Approach

To tackle the issues that arise from the lack of accountability, it is necessary to find the effects of some features of a versioning tool.

The most important features in a versioning tool are the ability to see past changes, the owner of said changes, as well as some pertinent information about them such as date and time. By keeping the information of all the changes made to an object or environment and the ownership of those changes, the users will be held accountable for their actions or the lack of them.

As such, in order to achieve a feature-rich and useful versioning tool, it is advisable to divide the implementation feature-wise, i.e., the implementation should approach each relevant feature at a time, building the final solution in increments, and seeing how each of them impacts the accountability of the platform.

3.2 Requirements

The requirements serve as a development guideline, encompassing a collection of components and features that must be incorporated into the project. Their purpose is to steer the project toward its ultimate goal.

To categorize the requirements based on their expected impact on applications, the subsequent sections have been partitioned. First, a section detailing some key concepts and terminology used. Then a section presenting the requirements for the input.

3.2.1 Key Concepts and Terminology

To understand how a user can interact with the system, a few concepts must first be introduced. In this section, we will explain the terminology used.

3.2.1.1 Event

An event refers to any change or action applied to an object. An event can be either an import, a translation, a scaling, or a rotation. Each different type of event is represented differently on the events panel, except for the translation and import, which are both represented by a normal square. The others are represented by a rotated square when it is a rotation and a scaled-up square when it is a scaling. When selected, the event square will glow on the events panel *(concept presented below)*. Each event will have associated a user who performed the action, the type of action, and the date.

In Figure 3.1 are the representations of the events on the events panel.

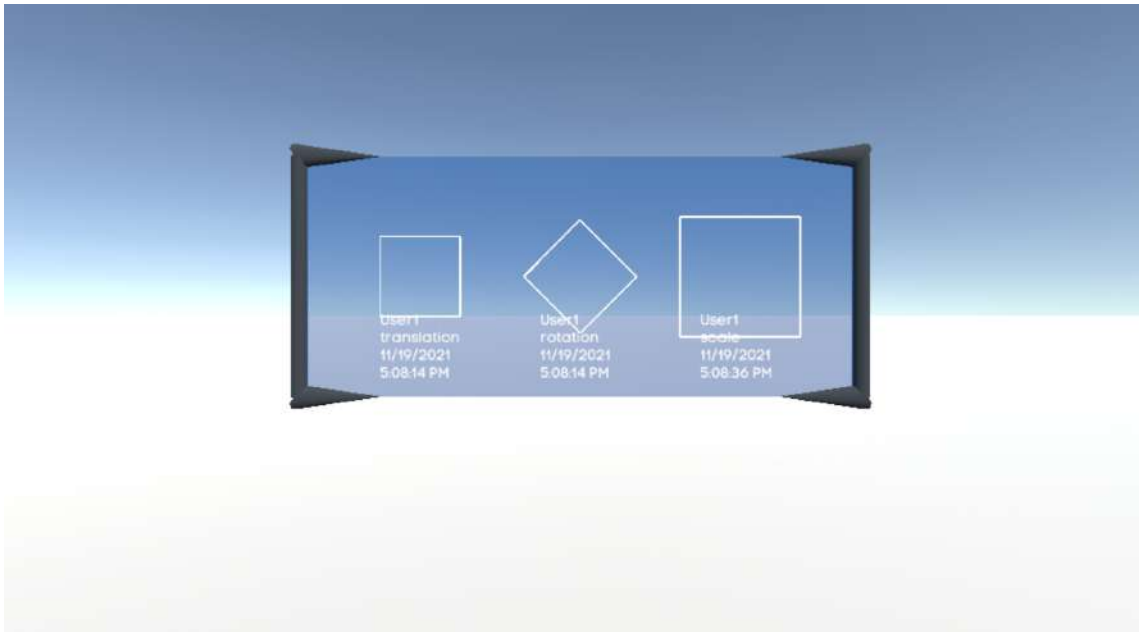

Figure 3.1: Examples of Events on the event panel

3.2.1.2 Node

In this context, a node represents a physical location in the virtual world where the object it is associated with has been. It is represented by a blue cube, except when it has an import event associated, in which case it is green.

Each node has at least one event associated but can have any number of events. To help the user understand what events happened in that position, the cube will have corresponding transformations, i.e., if a scale event happened in that position, the node cube will be scaled up, if a rotation happened the cube will be rotated and if that position coincides with where the cube was imported, it will appear green.

When a node is selected, it will emit a soft glow in order for the user to easily identify which node is selected, and, therefore, where the associated events happened.

An example of a node can be seen in Figure 4.7

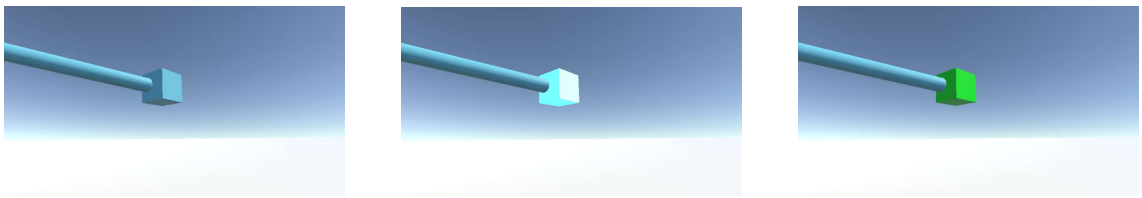

Figure 3.2: Example of a normal node, a selected node, and an import node

3.2.1.3 Path

A path refers to the line between positions where an object has been. It is non-interactable and does not depict the actual path the object took, as the system only stores where the user drops the object and not the actual path it took. Nevertheless, it is important to give it a name to make it easier to reference.

An example of a path can be seen in Figure 3.3

3.2.1.4 Timeline

The timeline is the representation of the path of physical positions the object took on the scene at different times, along with all the nodes and events associated.

Each object in the scene has its own timeline, and each timeline has a beginning and end position (although these might coincide and the object might have crossed the same node multiple times). The ending position can be easily discovered as it will be placed where the object it is associated with currently is. To identify the beginning position, the user must trace back the path the object took until it finds the import event. The import node can also be distinguished by its green color, which is not found in any other node.

An example of a timeline can be seen in Figure 3.4

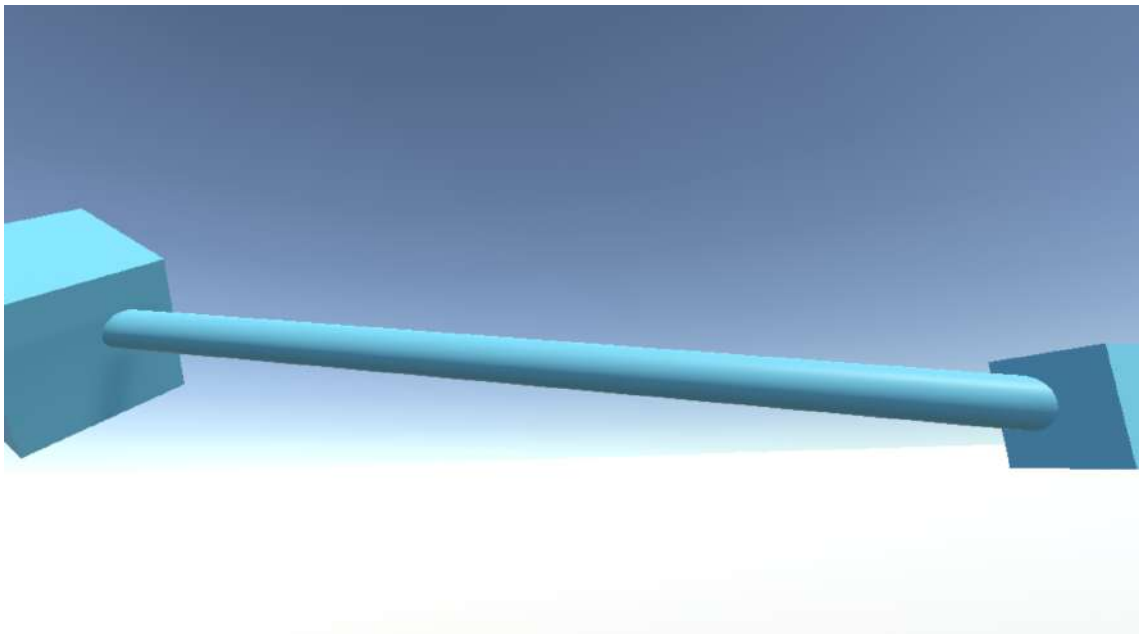

Figure 3.3: Example of a Path

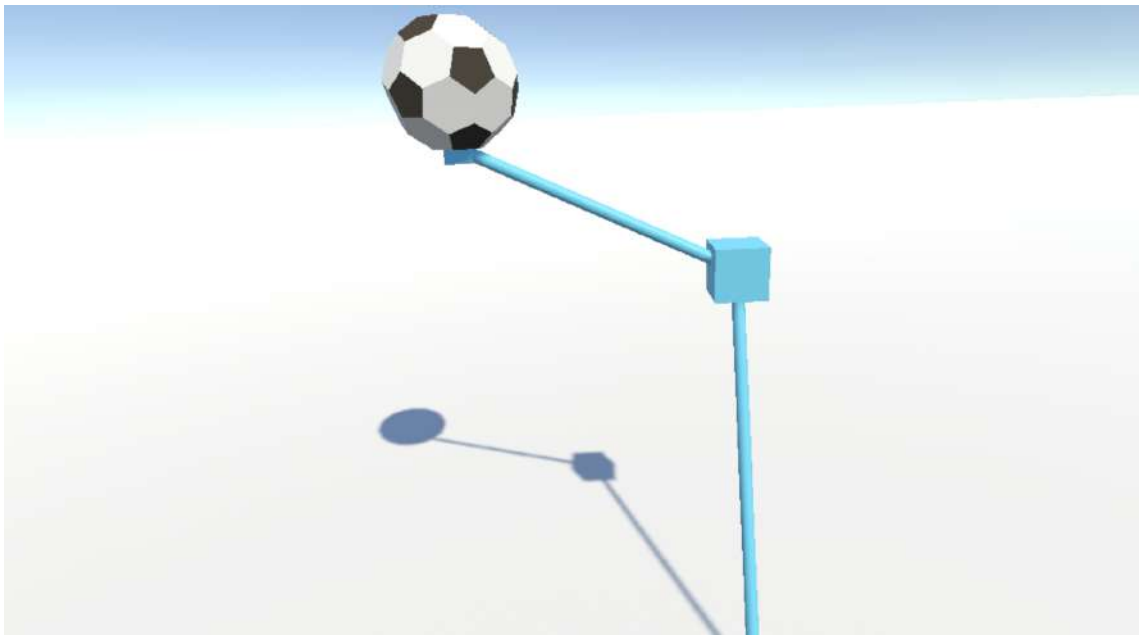

Figure 3.4: Example of a Timeline

3.2.1.5 Scene Object

A scene object is any object in the scene that has a timeline, excluding the timeline itself and its associated nodes, panels, events, and snapshots. It is the object in which the users have made changes and want to see the changes made in the past.

An example of a scene object can be seen in figure 3.5

Figure 3.5: Example of a Scene Object

3.2.1.6 Scene

A scene is a group of objects that make up the virtual environment put together by the users. A scene will have multiple scene objects, and each scene object has its own timeline. Although the scene has its own timeline, it is not accessible as building a UI for it would be too intrusive and break immersion. As such, to navigate the scene timeline, the user must do it through the scene objects.

An example of a scene can be seen in Figure 3.6

3.2.1.7 Events Panel

The events' panel is a panel that appears above the hand opposite the one which selected a node, showing all the events that occurred in the position. The panel is where the user can see all information about each of the events.

An illustration of the events panel can be seen in Figure 3.7.

3.2.1.8 Snapshot

A snapshot is the *"ghostly"* representation of the state of an object at a specific time or event. It can assume three different colors: red representing the state of the object before the event, green representing the state of the object after the event, or white representing an object of the scene not affected by the event. This last representation only appears if the scene view is activated. An explanation of what is a scene view is presented below.

In Figure 3.8 are represented the different types of snapshots.

Figure 3.6: Example of a Scene

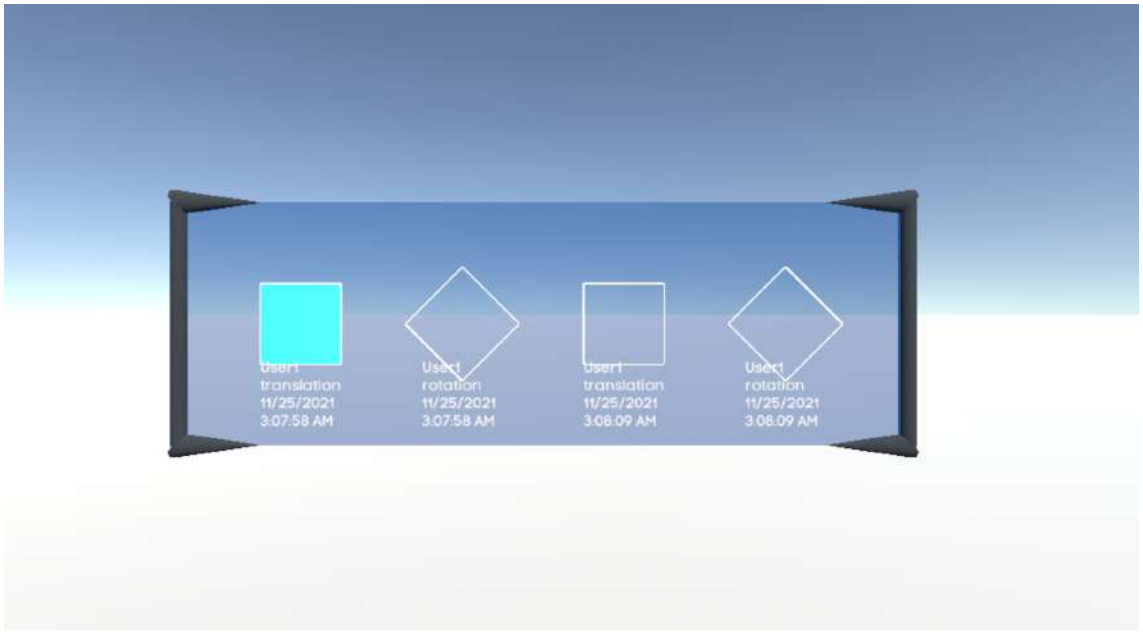

Figure 3.7: Events Panel

3.2.1.9 Scene View

The scene view is the group of snapshots of the deselected objects at a specific time or event. It can be toggled on or off by the user at any time by flicking a switch below the user's left hand, hiding or showing how the scene was at a given time.

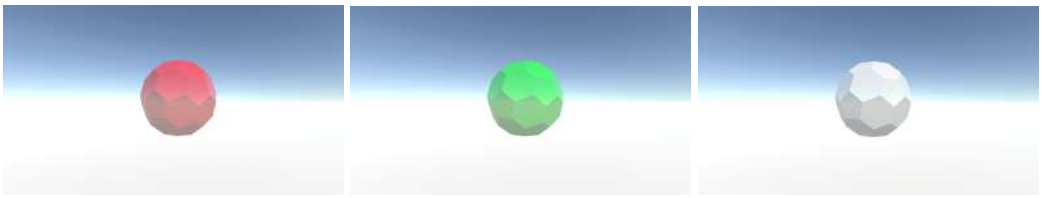

Figure 3.8: Snapshots of an object

In Figure 3.9 is the switch to toggle the scene view, and in figure 3.10 is an example of a Scene View.

Figure 3.9: Scene View Switch

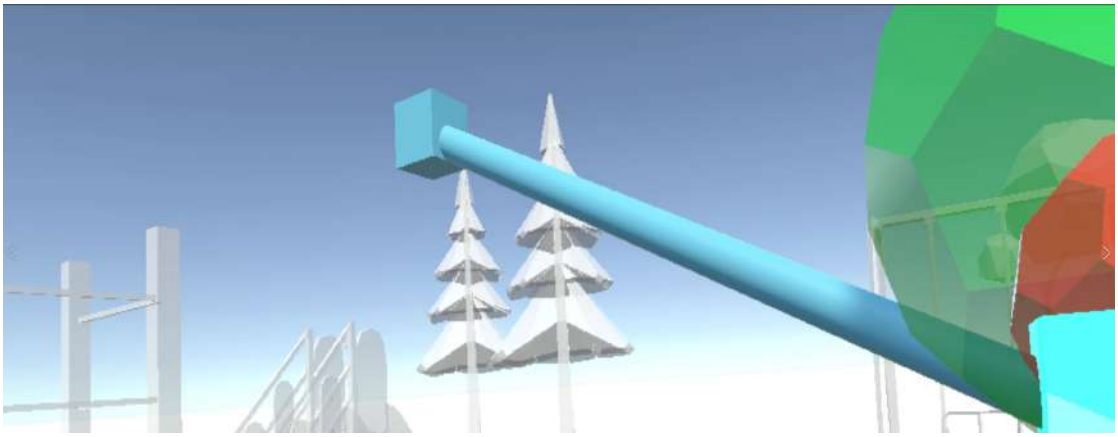

Figure 3.10: Example of a scene view

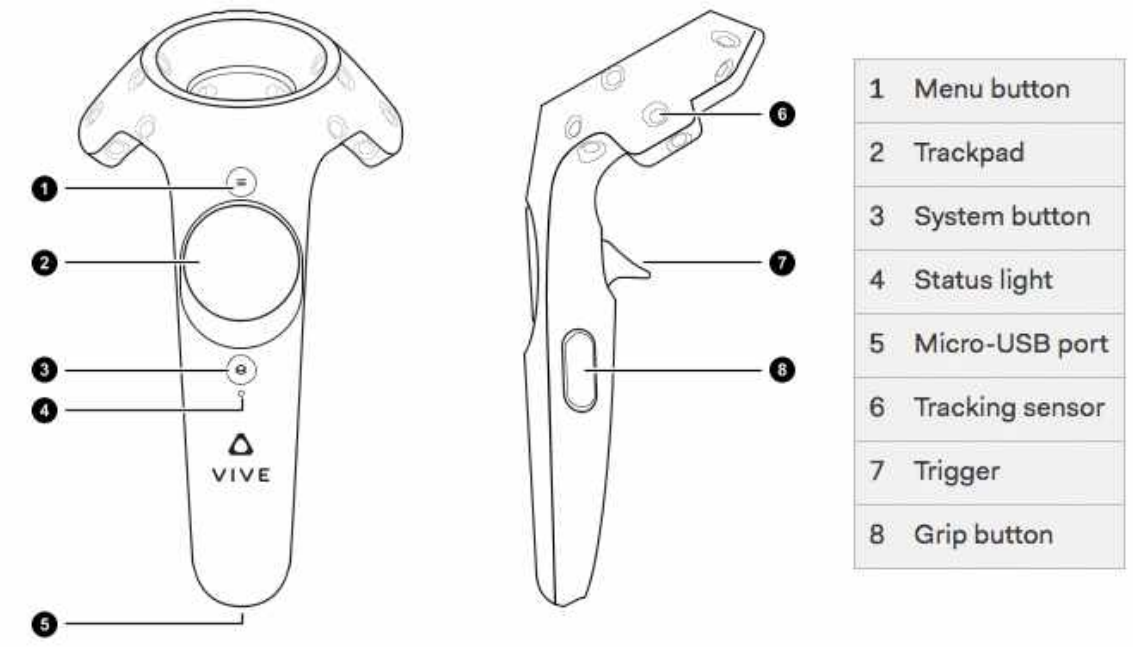

Figure 3.11: HTC Vive Controller

3.2.2 Input Hardware

There are multiple different VR Rigs in the market nowadays, offering different features, but all of them track the orientation of the user's head. Some also have external sensors that track the position of the user in the room and convert it into positions in the virtual environment. The orientation and position of the headset are used to change the virtual camera's transform.

Besides the orientation the user is facing and its position, it is also necessary to track the user's hands and gestures. Hence, the rig must have controllers, although it need not have many buttons, as only one gesture is used in the current system.

The system should be designed in a way that is compatible with most VR rigs to make it more accessible. For this reason, it should be dependent on the least amount of features offered, preferring the ones that are most common.

For these reasons the HTC Vive was chosen for the development of the prototype, as it provides with the basic features needed. However, the system should be compatible with any other rig that meets these requirements.

3.2.3 Features

3.2.3.1 History

The most basic and essential feature of any versioning tool is the ability to see previous versions of a model or the environment. This would be the history of the model. Not only should it be able

to see the information of the history such as the who, what, and when, but also to see the change itself, showing the state of the object at that time.

In order to see the changes made to an object, the user needs to first select the object in question. This could be achieved in different ways, for example having a list of objects and selecting from that list or by touching the object with the index finger. The second approach seems more intuitive and as such was chosen as the solution.

After the object is selected, the changes must be displayed to the user. Again there were multiple solutions possible. One was having a panel with all the changes made. However, if there were a lot of changes made to the object, this panel would have to be too big and might not be very immersive or practical. Another solution would be to present the changes as nodes in the virtual environment. This also is not optimal, as there might be many changes to the object in a single position. In the end, a mixture of the two solutions was chosen, grouping changes by positions, represented by nodes in a 3D timeline. Each of these nodes would have in turn events associated, and once the user selected the node in the same way he selected the object, i.e., by touching with the index finger, a panel would pop up with the events associated with that location. An idealization of this can be found in Figure 3.12.

In order to achieve this, a sort of database will have to be put in place to hold the past versions of each model, which will be updated after every change made to the environment.

3.2.3.2 Ownership and Other Information

After having the database and history preview feature in place, it will be possible to store some extra information such as who made the change and when on said database. This will allow a reviewer to see who made the change, increasing accountability in the system.

This information should be presented with each change made to an object. Again various solutions were thought of, such as the possibility of when a change was being seen, a pop-up would appear with the information. This solution dropped in favor of having the information in the events panel, as this way the users could search the changes without having to actually select each event to see if it was the one they are looking for and reducing the number of interfaces that appear on the screen.

3.2.3.3 Scene View

The last feature planned is the Scene View. The scene view allows a user to see how the environment was at the time of a change. This last feature uses the information of the two other features, not requiring additional changes to the architecture, as it just has to see how each object was at the timestamp of the selected action.

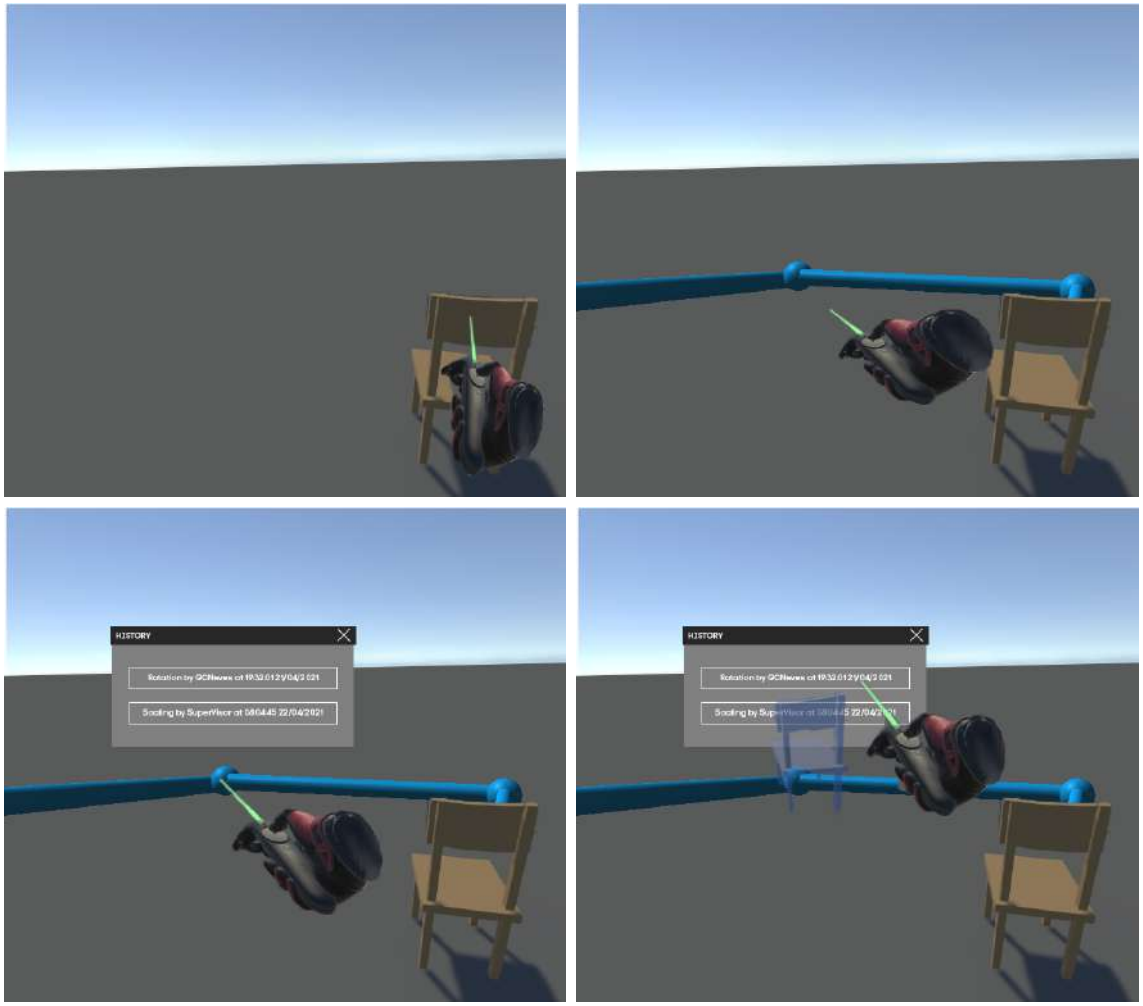

Figure 3.12: Usage example

Chapter 4

Prototype

The present chapter centers around the prototype and its implementation. It commences with the system interaction design in section Using the System, along with the presentation of a set of features. Section Tools and Technologies then offers an overview of the system architecture and interaction diagrams. Additionally, section Solution Architecture describes the tools and technologies utilized in the prototype's implementation. Finally, section Implementation Specification delves into the specifics of certain implementation decisions and offers a justification for them.

4.1 Using the System

In this section, a detailed explanation of how to use the system will be presented. To help understand the usage of the system, there will be figures of an actual scenario accompanying the tutorial.

4.1.1 Modeling

Regarding the modeling of a scene in the prototype, four actions were supported, the Import function, the Translate function, the Scale function, and the Rotate function. These were chosen to be a part of the prototype as they are the simplest and more crucial to 3D modeling.

A detailed description of how they work in the prototype is presented below.

4.1.1.1 Import

The import function serves as a way to bring objects into the virtual scene, to be used in the process of building the scene.

To import an object, a user must open the import panel. This is achieved by clicking on the trackpad of the controller previously presented. The import panel will give the user a list of objects which can be imported. Afterward, the user would select the object he wished to import

by pointing and touching the corresponding label. The object would then be placed at the position of the controller without any other transformations, i.e. scaling or rotation, and the import action would then be stored on the timeline and database.

These objects were manually added to the prototype, as we are more interested in storing the actions instead of conducting the actions. However, this functionality could be improved by connecting it to one of the many existing online assets libraries and importing the models from there.

The import panel automatically closes when an object is selected to import or if the user clicks the trackpad again.

The object is put into the scene by checking which label was touched and then matching the text in the label with the names of the prefabs. When a match is found, the corresponding prefab is instantiated in the scene without any parent object, only changing the position of the object, keeping all the other properties the same as the prefab.

4.1.1.2 Translate

The translation is the most basic and fundamental action. It is the action of moving a model from one place to another.

To achieve this, the user must reach for the object, touch it, and while touching it, grab it by squeezing the trigger on the controller. As long as the user is holding the trigger, the object will follow the position of the controller. When the user lets go of the trigger, the object will assume its' new location as the new position and store the action with the coordinates and timestamp in the timeline and the database.

The object follows the controller by setting the first as a child of the second on the scene hierarchy. This way, any transformation made to the controller will also be applied to the object. When the user lets go of the trigger, the object ceases to be a child of the controller, and the coordinates it currently has are the ones used to populate the database.

4.1.1.3 Scale

The scale action allows the user to change the size of an object in the scene.

This action requires the user to use both controllers at the same time. The user must grab the object with either controller and then with the other controller do the same. Once both controllers are simultaneously grabbing the object, the scaling action begins. As the user increases or decreases the distance of the controllers, the size of the object will change in the same ratio. Once the user stops grabbing with either one of the controllers, the scale action will stop, and the translation action restarts. Only when the user stops grabbing with both controllers will both actions end, and be stored where the object is last placed. The translation action will be the first one to be stored and then the scale action, storing the date of the change, the user who made it, and a vector of three numbers with the new size in each coordinate to the database.

Once the user starts the scale action, i.e. grabs the object with both controllers, the object ceases to be a child of the controller that first grabbed it, so as not to follow its position. The scale to be applied is calculated by comparing the difference of the initial distance against the current distance of the controllers and multiplying the difference by the initial scale of the object, thus following the ratio. Once one of the controllers is let go, the object is again set as the child of the controller that is still holding the object, continuing the translation action previously started.

4.1.1.4 Rotate

The rotate action allows the user to change the orientation of the object.

This action works similarly to the translate action, as it starts when a user grabs the object with either controller. While the user is grabbing the object, it will follow both the controller's position and orientation, and when the user releases the trigger, the object will assume the new position and orientation.

When the user grabs the object with one of the controllers, the object is set as the child of that controller, which will apply any transformation made to the controller to the object. When the user lets go of the object, it ceases to be a child of the controller in the unity hierarchy, and stores first the change in position to the database, and then the quaternion representing the new orientation of the object.

4.1.2 Viewing the changes

As explained previously, each scene is made up of multiple scene objects. Each of these scene objects has its own timeline associated, which the user can navigate to see the changes made to it. In order to see the timeline, the user must first select the object, as shown in figure 4.1.

After selecting an object, the timeline and the nodes associated with it will become visible. The user can then choose one of the nodes to see the events associated with it. As mentioned before, each node has at least one event associated with it, either an import or a translation, but can have any number of events of any kind. The process of selecting a node can be found in figure 4.2.

After selecting a node, the events panel will become available. Here the user can see the different events associated with that node, as well as the information related to each of those events. The user can then select the event in which he is interested from the events panel, in order to see how the object or scene was at a determined point in time. This can be achieved as demonstrated in figure 4.3.

As shown in the previous image, after selecting an event from the events panel, two snapshots will appear, one in red and one in green. These correspond to how the object was before and after the event, respectively. This already gives us information about the who, what, when, and how of a certain change, but sometimes the context, i.e. the why, of an event is relevant in order to explain the actions of a user. In other words, one must have awareness of the state of the system, in order to judge the accountability of a user. For this reason, it is crucial to be able to see how the

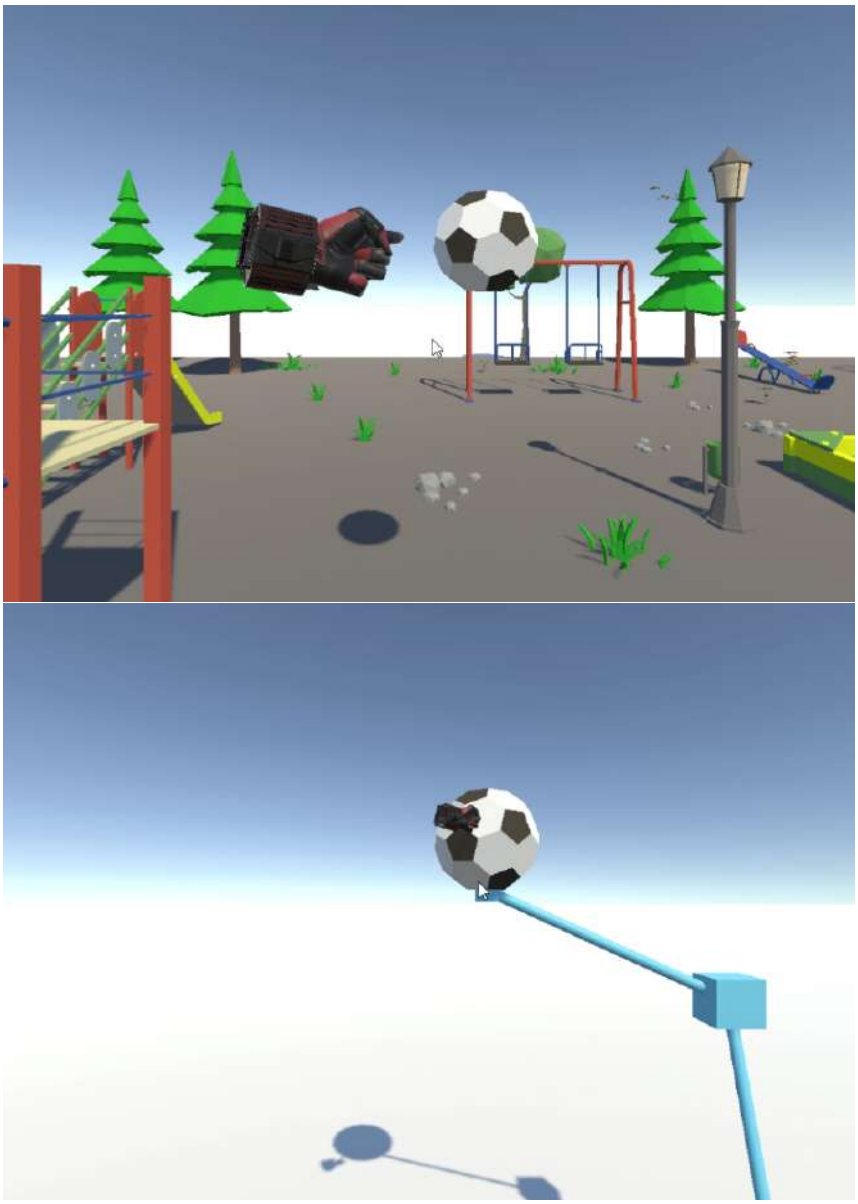

Figure 4.1: Selecting a scene object

scene was at the moment of the change. This can be done by activating the scene view, as shown in figure 4.4. A user can activate the scene view at any time, but it will only show the snapshots of the objects in the scene when an event is selected.

Lastly, in order for the user to go back to the current view of the scene he has to follow these same steps but backward, i.e. deselect the event, then deselect the node, and finally deselect the object. This process is depicted in figure 4.5.

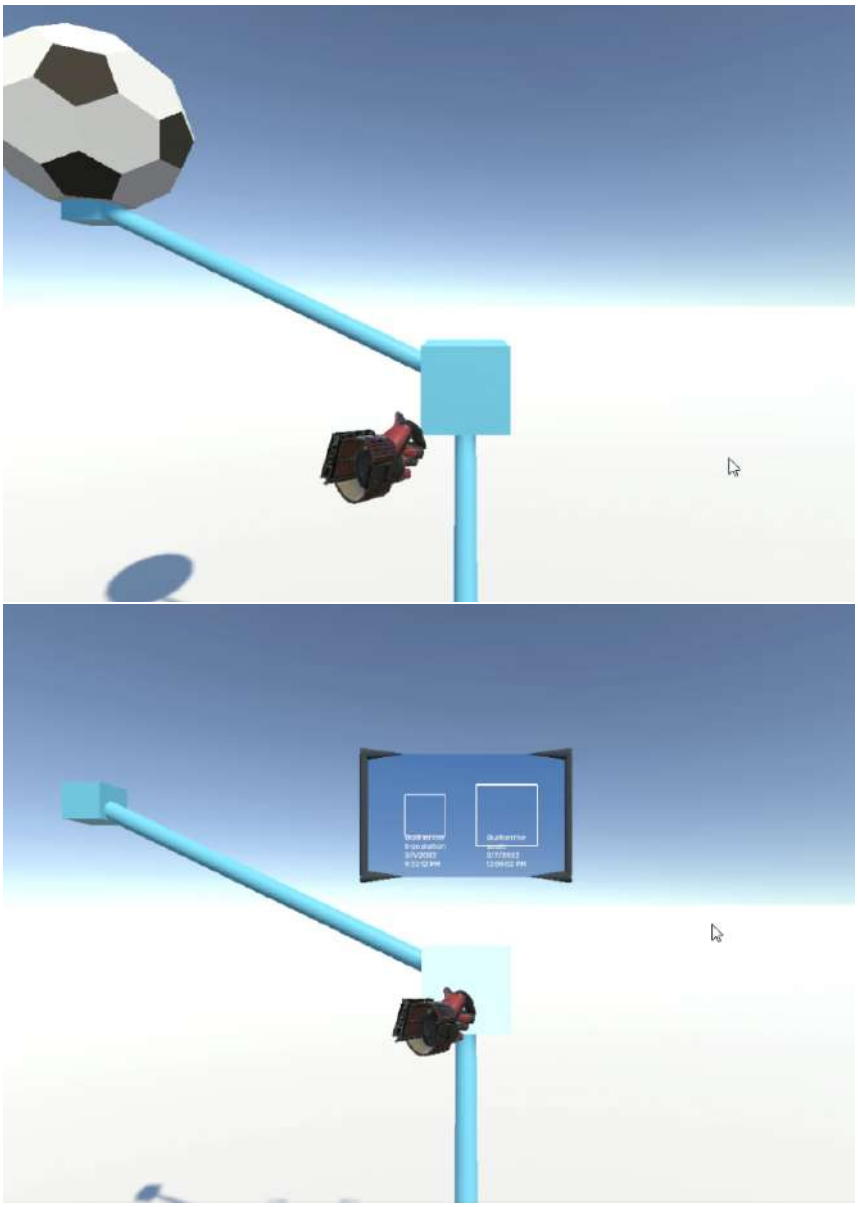

Figure 4.2: Selecting a scene object

4.2 Solution Architecture

This solution was built with the PAINTER (Procedurally Assisted INTErchangeable Reality) [27] platform in mind, in order to see the impact on a collaborative content creation platform inside the Virtual Environment.

It was built on the Unity platform. Unity is an extremely popular solution in the market, having a lot of community support in case help is needed, it has extensive documentation on how to use it, it is extremely user friendly, with a short learning curve, and is very performance efficient.

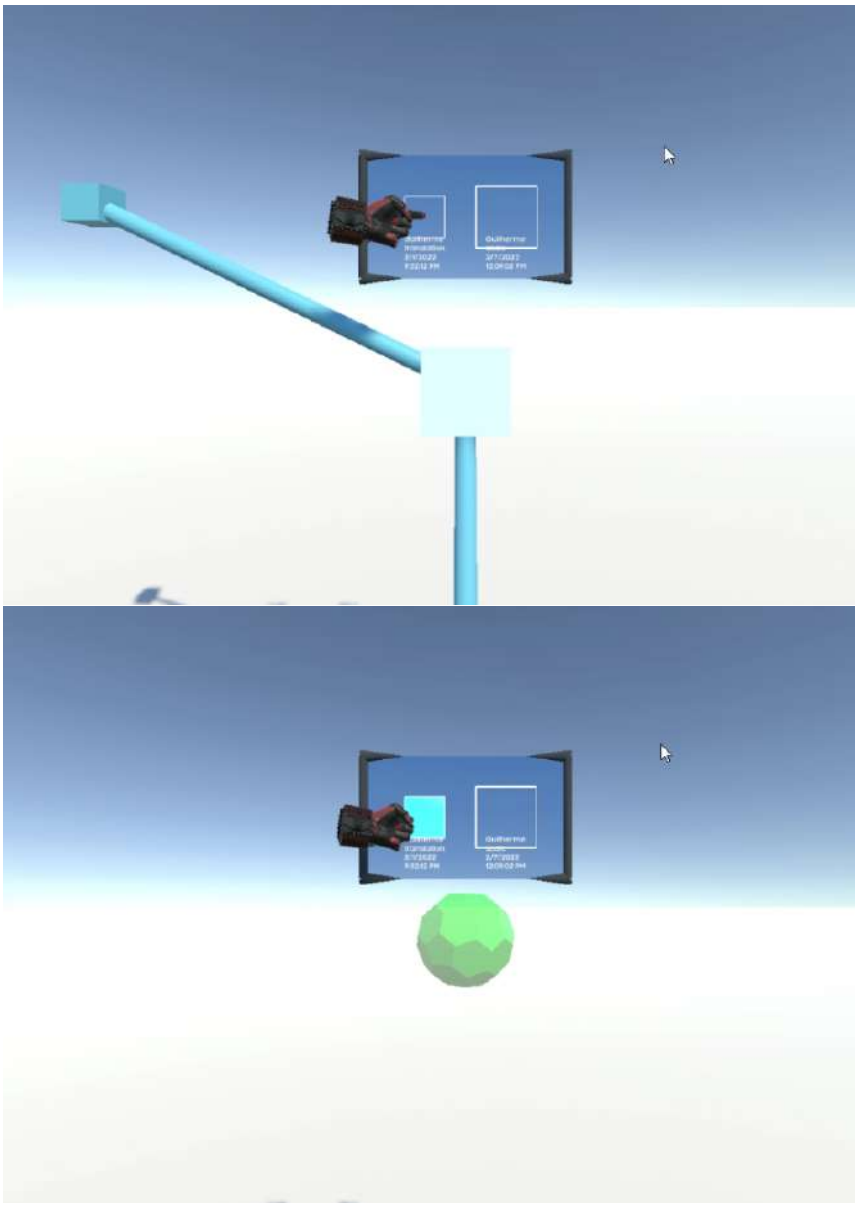

Figure 4.3: Selecting a scene object

4.2.1 System Architecture

Figure 4.9 depicts several VR clients linked to a common server. The dotted lines indicate the communication that preserves a consistent state of the Virtual World across all clients. The components inside each box provide a broad overview of the framework's various elements and their interconnections.

The system predicts multiple users working in the same environment. As such, the environment, or Scene, must be stored in a centralized server that stores every object and its state, in order for every user to have the same information. Every action made on that scene must be stored in a database on that server.

Figure 4.4: Activating the scene view

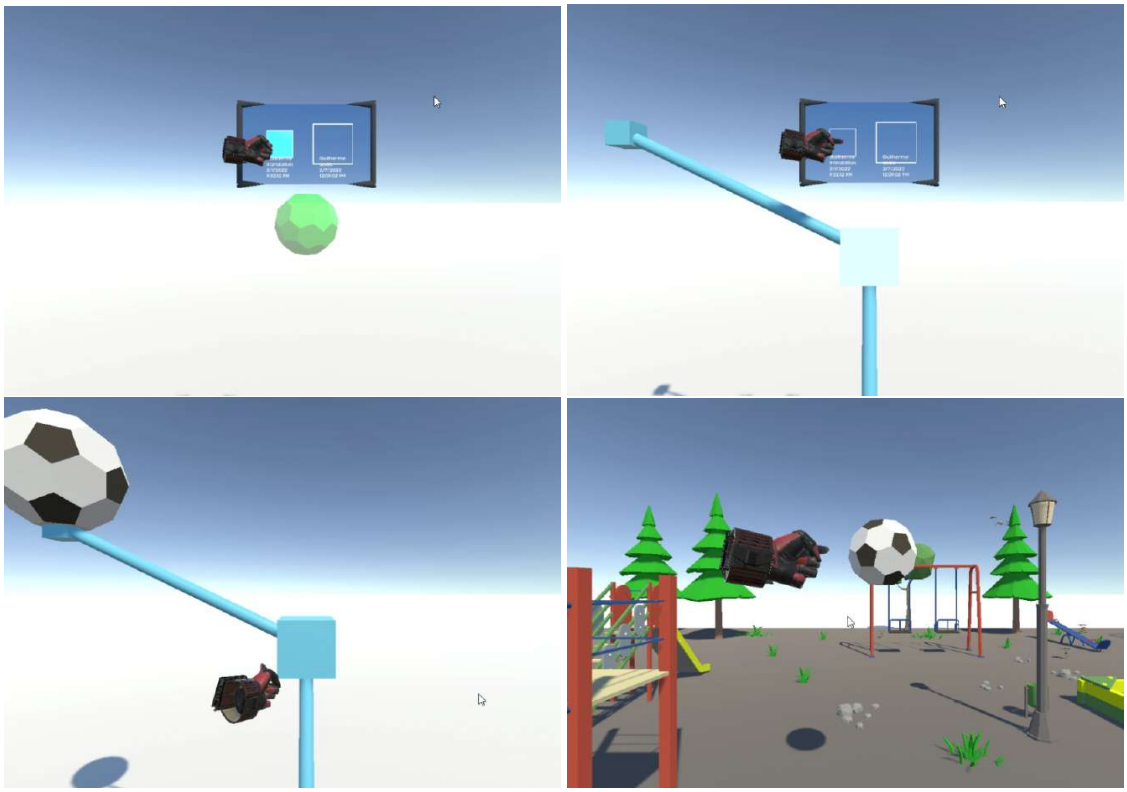

Figure 4.5: Returning to the current state of the scene

4.2.2 Interaction Diagram

The VR client's interaction with the virtual world is not illustrated in the system diagram. However, figure 4.10 presents an interaction diagram that breaks down the dissertation's planned requirements into distinct components. The 3D blocks represent components of the system, whereas the squares represent interactions. The three stacked cylinders represent the database that stores the objects and actions made to them.

The user can interact with any of the other components through actions. These actions can either be a transform action, which will change the position, rotation, or scale of an object, or a select action. The select action has different outcomes depending on which component the User is

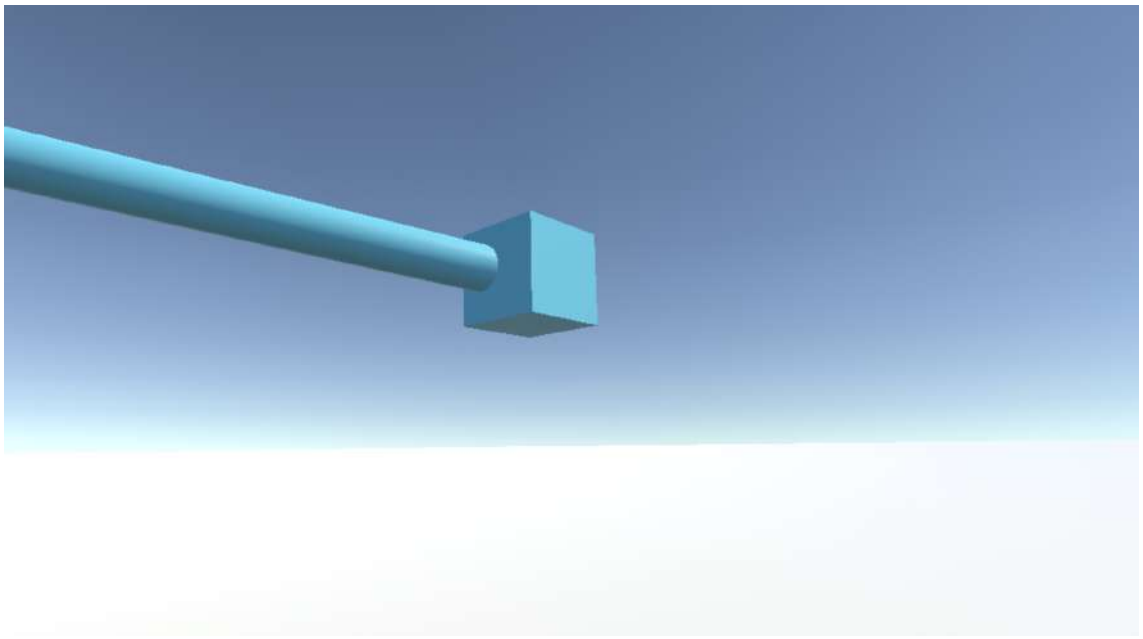

Figure 4.6: Example of a deselected Node

interacting with.

The transform actions can only be applied to objects. When the action is applied, it is stored in the database.

When the User selects an Object from the scene when no Object is selected, that Object will become selected. It will fetch the history of that Object from the database and open a Timeline of the changes made to the Object. However, if the Object target to the Select Action was already selected, this will result in the deselection of the Object, and the Timeline will disappear.

When the User navigates the Timeline by selecting it, more specifically a node of the Timeline, a Panel will open with the events associated with that place in the Timeline. Again, if the User uses the Select Action on a selected node, the Panel will close.

Once the Panel is open, the User can now select any of the events displayed in it. By selecting an event, a Preview of the corresponding Object will appear at the time of the event. This Preview is not interactive, and the User can not transform it or select it, serving only as a visual representation of how the Object was at a specific time. The User can hide this Preview by using the Select action on the selected event on the Panel.

4.3 Tools and Technologies

As this system is to be used by multiple users in a system, the technology choices and implementations must take into account that each user might have different setups, and as such the

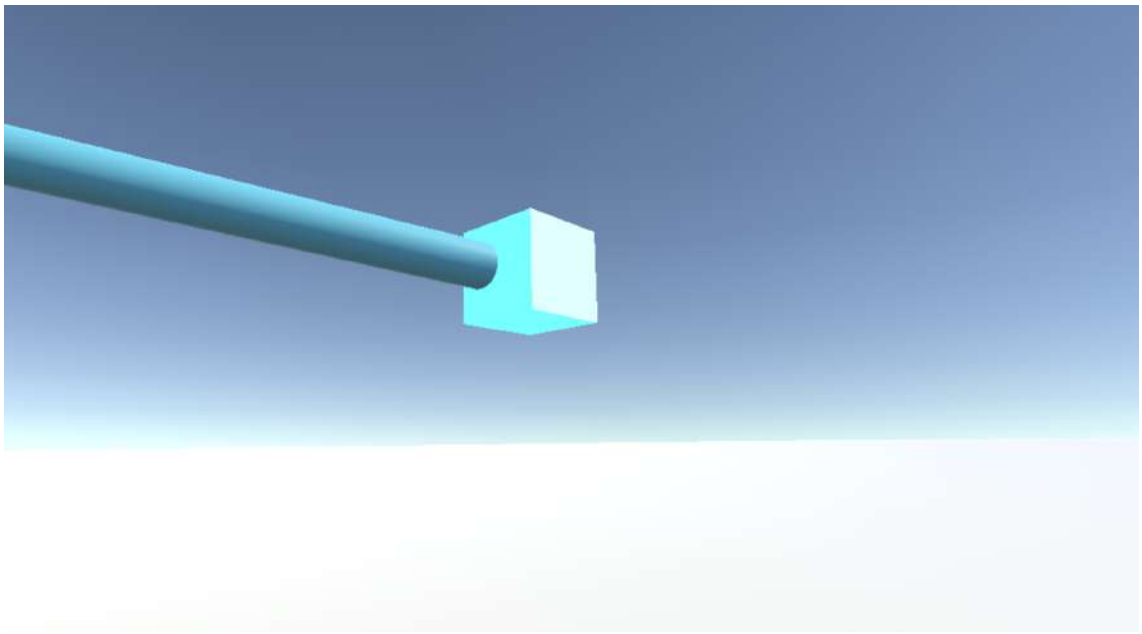

Figure 4.7: Example of a selected Node

development must be as inclusive as possible. For this reason the Unity [1] game engine was used, as it can be built for different systems easily, without having to change anything in the project.

Unity provides a lot of abstraction which allows the developers to focus on their projects, not having to worry about the graphics engine, physics engine, and collisions, amongst other features. It also has a large community of developers already using it, providing more support than other solutions. This game engine was chosen for these reasons and also due to having previous experience with it.

Unity 3D uses C# as a scripting language to create behaviors for each of its components. By adding and removing scripts from the objects we can dynamically change their behavior.

In order to support multiple VR setups and equipment, the SteamVR plugin for unity [2] was used. SteamVR is an open-source library developed and maintained by Valve that supports multiple VR devices, which maps actions to different controllers. This way developers only have to use actions on their system, and users can bind buttons from their controller to those actions. This gives the system compatibility to more VR devices, not having to manually develop the interactions to each of those devices.

4.4 Implementation Specification

To make the experience as immersive as possible, the system interactions and the way information was presented had to be thought in accordance with the definitions of proxemics and affordances,

[1] Unity: https://www.unity.com/
[2] SteamVR Unity Plugin: https://valvesoftware.github.io/steamvr_unity_plugin/

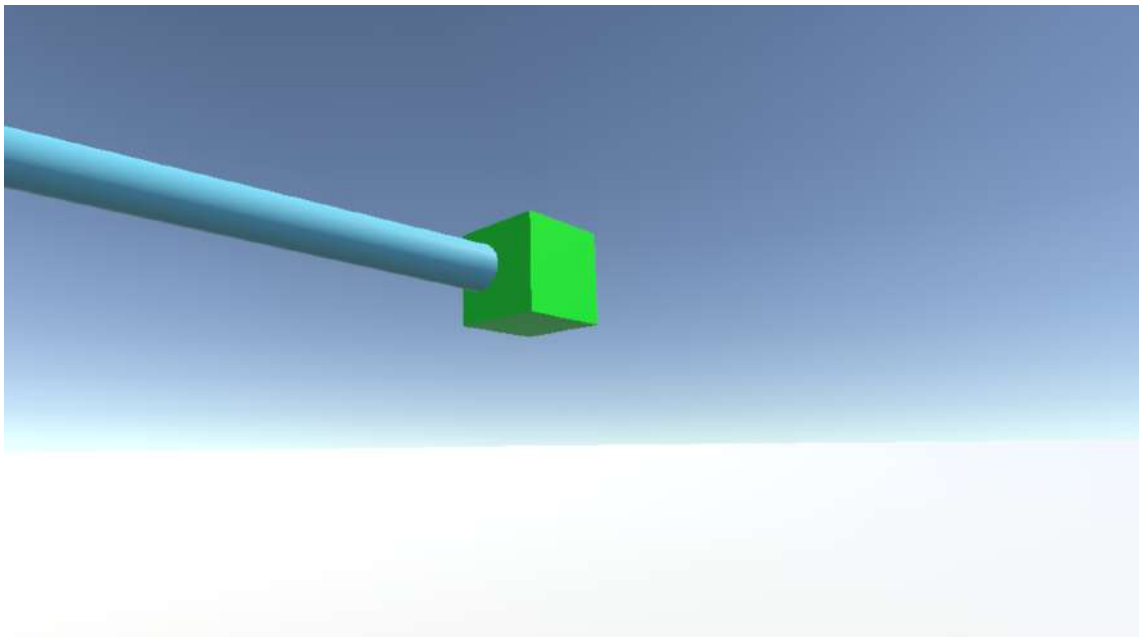

Figure 4.8: Example of an import Node

presented in chapter 2.

By hiding the scene when an object is selected, emphasis is given to the timeline. This way we can communicate to the user the locations where the object has been more clearly, through the nodes represented by the cubes. These cubes also communicate other actions present in that location by applying a transform to it, i.e. if the object was scaled in that position, the cube will be scaled, and if the object was rotated in the position, the cube will be rotated.

It is important to note that if some actions were performed close to where other actions were performed, more specifically four times the size of a regular cube, the actions will be grouped in the same node of the timeline. This was chosen so that cubes didn't overlap, making it easier to navigate the timeline, and it was chosen four times the size of a regular cube because a scaled cube is two times bigger than a regular cube, and this way if we have two scaled cubes next to each other, there will still be space between them.

The node location relative to the object is its origin. This was chosen as most 3D models are built upwards from the origin, and thus the node can always be visible below the object when selected, except in a few cases where the origin of the object is inside the object. In this case, the user must first select another node in order to hide the object and then select the desired node.

As mentioned previously, when an object is selected, the other objects in the scene are hidden. This approach was also used when selecting a node, which in turn hides the selected object, highlights the selected node, and opens the events panel. This way the user only has the timeline showing and doesn't have any object which allows him to more easily navigate the timeline.

When an event is chosen in the events panel, the timeline is hidden and a preview of the object as it was at the time of the event is shown. This way users can see how the object was without

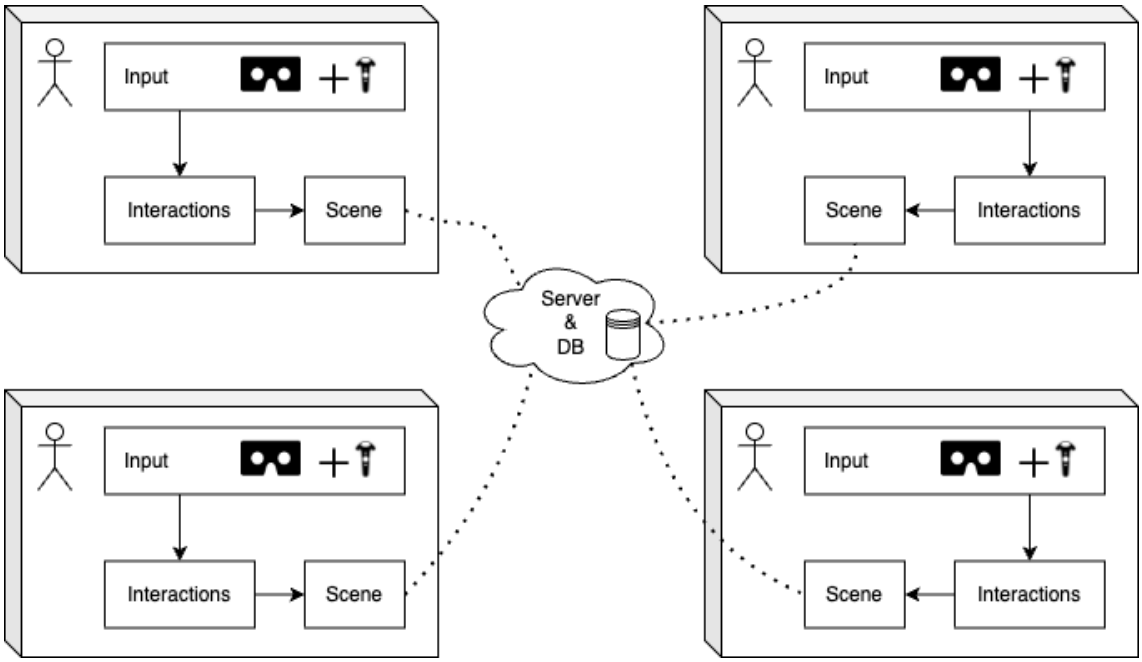

Figure 4.9: Architecture Diagram

having the timeline obstructing their view, which improves the immersion of the scene.

In order to mimic the real world and thus maintain as much immersion as possible, to select an object, node, or event or activate/deactivate the scene view, the user has to touch it while pointing. In order to maintain the events panel and the scene view switch always visible, they are placed respectively above and below the hand opposite the one which was used to select the object. This way they follow the user and make it easier for him to select the changes even far from the position, allowing him to have a better view of the change.

In order to make it easier for users to navigate the timeline and more easily identify where the object was first placed, as the timelines can very quickly get big and confusing, having some nodes overlapping each other, the import node was colored green, making it distinguishable from other nodes.

In an early version of the prototype, the path between nodes was made with Bézier curves as it was thought by the author that it would make the timeline better looking and more immersive. However, testing it with larger timelines, it became obvious that the curves made the whole system very confusing and extremely over-saturated, making it harder to navigate the timeline and cutting on the immersion factor of the system. For this reason, straight lines were chosen to represent the path between nodes.

When viewing an action, the ghost of the previous state to the one selected and the selected state was displayed. This initially was a choice, but the advantage of seeing how the object was before the change without having to select the previous node led to the decision of leaving the feature always active. In order to distinguish both ghosts, the colors red and green were chosen for the previous and current states respectively. These colors were chosen as many versioning systems

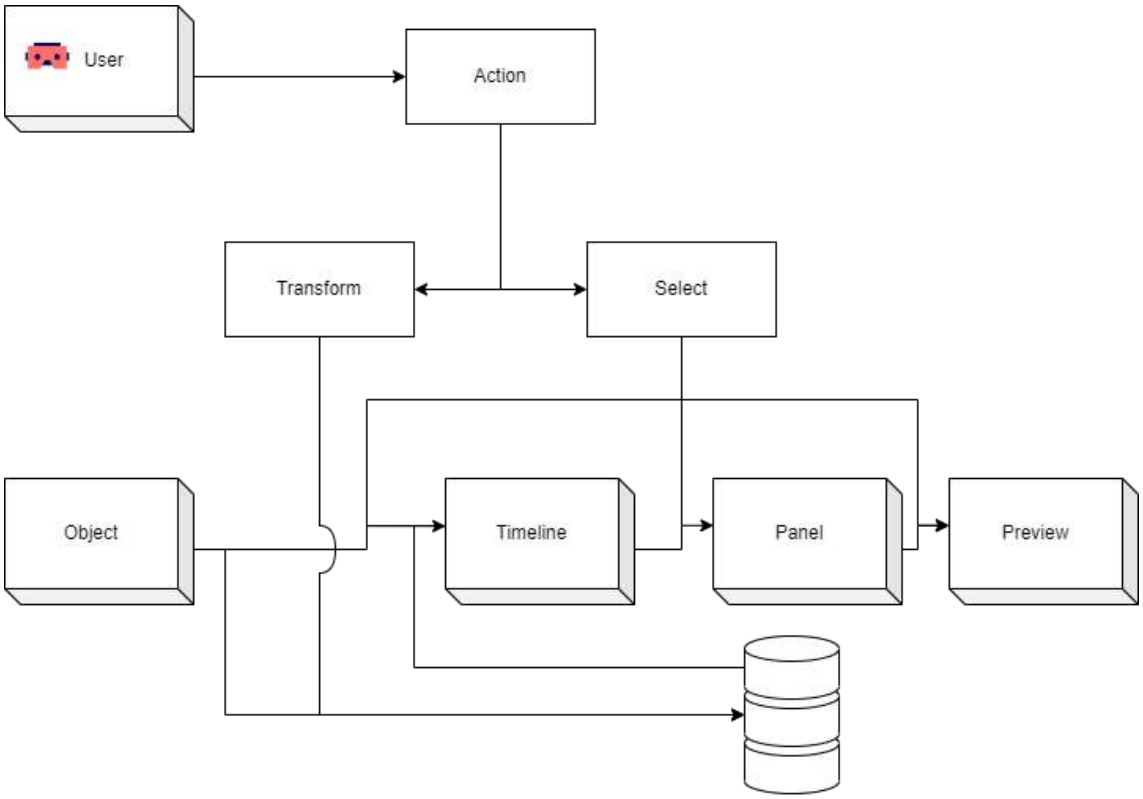

Figure 4.10: Interaction Diagram

use the same coloring system, for example, GitHub uses red for lines of code removed and green for lines of code added and 3D Repo uses a similar color scheme with some additional colors for other actions, and thus would be more familiar to a user that has had experience with these types of software in the past.

Regarding the persistence of the data, the current approach stores the changes locally in JSON files. It divides each scene into different folders, and inside each scene folder are stored JSON files with the changes made to each object. Each JSON file stores the timeline for one object in the scene, and is named with a the name of the object and the UNIX timestamp of when it was imported. This will be used as the ID of the object on the scene timeline. An example of this structure can be seen in figure 4.11.

This approach was chosen as it presented a simple way of storing the information without requiring too much effort, allowing for the focus on the research questions and not on development specifications that other options might have presented, such as using a database, which would require the setup of said database, and using some integration of the database technologies with the current system.

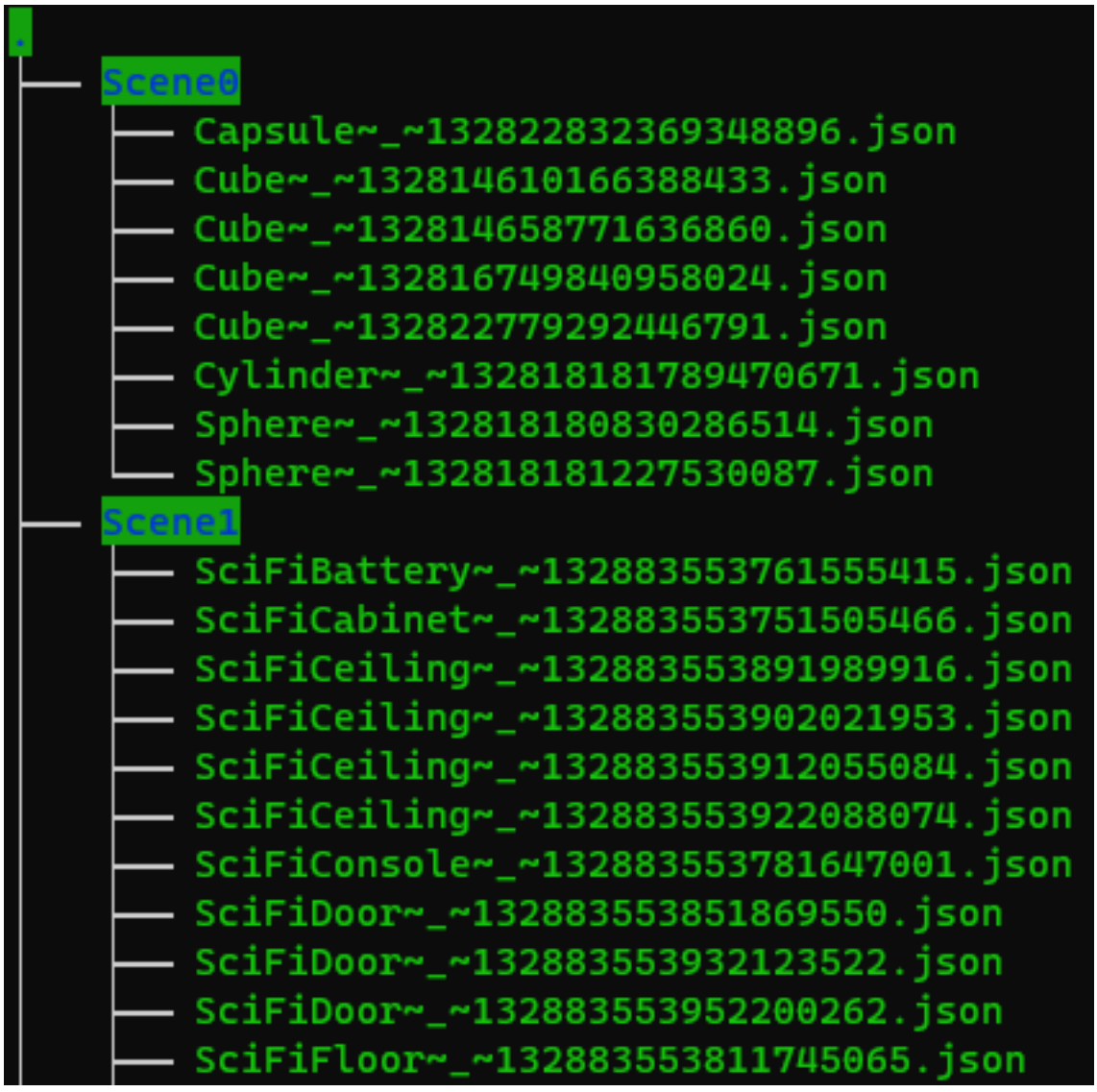

Figure 4.11: Example of local folders with the JSON files for each object

Chapter 5

Evaluation

This chapter will describe in detail the evaluation phase presenting the results of the tests made on the tool made at this stage, and with them ascertain the answers to the research questions. This chapter will be divided into five sections. In section 5.1 a description of the tests that were conducted will be given, and how they help validate the solution. Section 5.2 introduces the prototype tested in this section. Section 5.3 describes the methodology used for the tests and describes the tasks for each test. Section 5.4 details the apparatus and the participants that were part of the user tests. Section 5.5 details the type of data gathered from the tests, including the metrics. In section 5.6, the results of the tests are presented and analyzed. Finally, section 5.7 uses the results from the analysis in order to infer answers to the research questions and validate whether or not the solution solved the problem presented.

In order to assess the viability of the solution to increase accountability, both objective and subjective approaches to testing were used. The subjective approach had as a main goal the assertion of whether or not the solution is practical for it to be used in real projects. This was achieved through user questionnaires related to their experience while experimenting with the solution. The objective approach focused on the time it took the participants to accomplish determined tasks and if the information collected was correct. Although there isn't any baseline data to compare the data to, it's important to acquire this information so that future research on the subject can use these values as a baseline of their own.

5.1 Validation Protocol

Although the prototype allowed for the modeling of a scene and saved the actions made in it, this feature was left out for the user tests described in the following sections. This was chosen due to the fact that the aim of this dissertation is to assert whether or not a versioning system increases accountability, and also the participants testing the prototype not having to necessarily know how to 3D model. Keeping the modeling features would only clutter the system and make it more confusing for the participants, without adding to the goal of this dissertation.

As mentioned before, the system developed in Chapter 4 was built with the Research Questions in mind, trying to solve the problem regarding the lack of accountability. As such, the data collected from the user tests needs to be useful to validate the solution.

Research question 1 focuses on whether or not a versioning system increases the accountability of users on the system. As such, in order to answer this question, data about whether or not users are able to hold and be held accountable for the actions taken needs to be gathered.

Research question 2 focuses on the user experience while using the system. To answer this question, data related to the immersion of the system needs to be gathered, and whether it breaks it or not.

Although this dissertation focuses on the accountability of users, different types of data were gathered during the tests not directly related to accountability. This data was used to validate the usability of the solution and might be used as a baseline for future studies on the subject.

5.2 Prototype

In order to test the presented solution, a prototype that stores the most common content creation actions was developed. This prototype was developed atop a content creation platform that allows for translations, rotations, and scaling. However, this solution focuses on the who, what, and when of those actions, and not on the actions themselves. As such, the users were asked to comment on these components of the prototype and did not have to make any of the actions themselves.

5.3 Methodology

The user tests were divided into two phases. The first phase where the participant could explore the solution freely to get acquainted with the functionalities, in which no data was gathered, and the second stage where the participants were posed with realistic scenarios and had to complete some predefined tasks. The first stage, on the other hand, as it was not meant as a task, did not pose a realistic scenario, having just a simple scene with some primitives, i.e. cubes, spheres, and cylinders. While the participant explored this scene, a brief explanation of the different functionalities and how to use them was given.

When the participants felt comfortable enough using the system, then they would start the second phase. In this phase, the participant had to complete a set of tasks on different scenarios using the features they tried during the first stage. These tasks aimed at being as close to realistic scenarios as possible and where the solution would be useful so that the results gathered might be more meaningful.

Each session took around 30 minutes to complete. The tasks were presented to the participants shifting the tasks, following the Latin square design, to avoid bias. Regarding each session, they started with a short presentation and explanation of the basic usage of the solution, with some spare time for user experimentation before actually starting the tests. Combined, these two stages did not take longer than 10 minutes. Afterward, each participant was given 5 tasks to be performed in the

following 20 minutes, giving 4 minutes for each task. Along each task, the participant was asked to fill out a questionnaire made with Google Forms regarding their experience while performing the task.

5.3.1 Tasks

As mentioned above the participants were asked to perform 5 simple tasks. Each task tried to mimic real-life scenarios, in an attempt to acquire more trustworthy results. All of the tasks consisted of finding information regarding certain changes in each scene. A maximum of 4 minutes was stipulated for each task, although no user exceeded it.

For the first task, participants were asked to find out who imported the chair into the scene, with the timeline being located in a hard place to get to. The chair was located in the center of a spaceship where the participants were. A screenshot of this first scene can be found in figure 5.1.

Figure 5.1: Spaceship Scene

For the second task, participants were asked to find who scaled the ball. The ball could be found floating next to the participant's head in the middle of a playground. The playground can be seen in figure 5.2.

For the third task, participants were asked to find who put the book in its current position. The book is located on top of a table in a dungeon with multiple other objects, such as candles, crates, a chair, and a column. In most cases, participants had to go around the column in the center of the dungeon to get to the book. The book can be seen hovering above the table in figure 5.3.

For the fourth task, participants were asked to see how the scene was at the time the garbage bin was imported. The garbage bin was placed right beside where the participants were in the street. They would have to activate the scene view and select the import action of the garbage bin, in order to see how the scene was at that time. The trash can be seen in figure 5.4.

Figure 5.2: Playground Scene

Figure 5.3: Dungeon Scene

For the fifth and final task, participants were asked to check at what time a plant was imported to the scene. This plant was misplaced in the ceiling of an office, and as such participants would have to look up in order to find the plant. Part of the office can be seen in figure 5.5.

Although only in one of the tasks was the scene view required, the participants could choose to activate it whenever they wanted, i.e. on any task.

Figure 5.4: Simple Town Scene

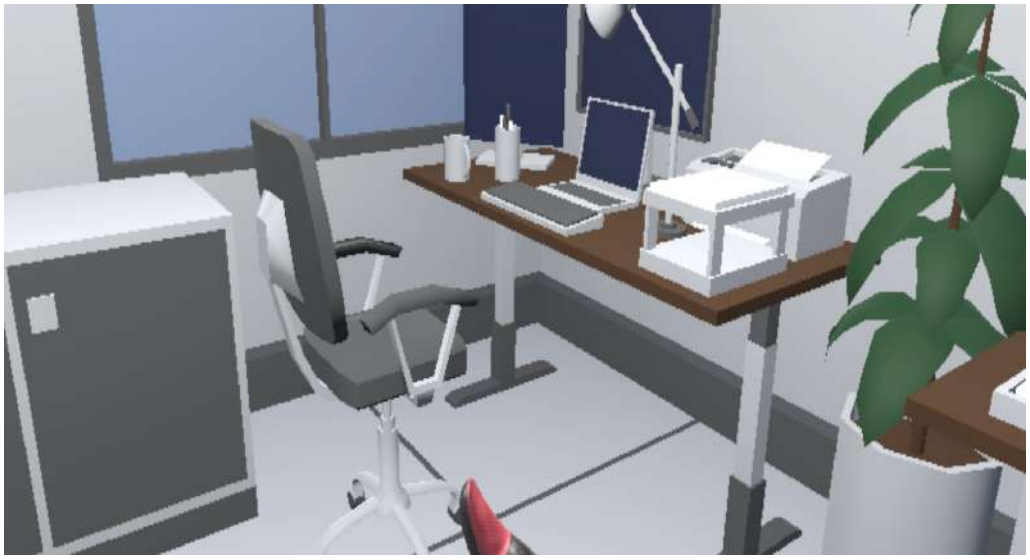

Figure 5.5: Office Scene

5.4 Apparatus and Participants

Evaluation sessions were conducted in the GIG lab in FEUP, room I220, which has restricted access, thus providing a calm and controlled environment. This room is also already equipped with all the required material for the tests, as such participants don't need to bring anything.

Due to the current pandemic situation, each session was conducted with only one participant and a moderator due to the capacity of the room, who both had to disinfect their hands and use a mask or face covering for the whole duration of the session, and the headsets and controllers used were disinfected between sessions due to the high risk associated with sharing headsets. All

Table 5.1: Gender Distribution

Gender	Participants
Male	14
Female	15

Table 5.2: Age Distribution

Age	Participants
<18	0
18 - 23	12
24 - 30	11
31 - 35	1
36 - 40	0
>40	5

Table 5.3: Experience using VR

VR Experience	Participants
More than once per day	0
1 - 7 times per week	0
1 - 5 times per month	0
less than once per month	11
no experience	18

Table 5.4: Experience using versioning software

Versioning Experience	Participants
More than once per day	6
1 - 7 times per week	0
1 - 5 times per month	2
less than once per month	1
no experience	20

participants also had to present either a valid Covid-19 vaccination certificate or a negative PCR or an antigen test.

5.4.1 Participants

A total of 29 tests were conducted throughout the testing phase. The participants were mostly contacted personally. The demographic data are presented in tables 5.1 through 5.4.

An analysis of the demographics shows that around 48.3% of participants identified as male and the rest 51.7% identified as female. Most of the participants were between 18 and 23 years of age, representing 41.4% of the responses. 41.4% of the participants were either FEUP students or alumni. 62.1% of the participants answered they had no previous experience with VR, and 69% never used versioning software. Of the participants that have had experience with versioning software, 66.7% use versioning systems more than once per day.

Participants of either informatics or users in any area of graphics design were preferred, as they are the most likely to use content creation platforms and versioning tools.

5.4.2 Apparatus

The prototype was developed using the unity game engine with the SteamVR plugin and is designed to work with any rig consisting of at least a headset and two controllers. The htc Vive headset and controller were used during the user tests as it offers hand motion tracking and an immersive experience, having external sensors which detect your position in the real world and translate them into the virtual world. 5.6

Figure 5.6: Test session

5.5 Gathered Data

5.5.1 Metrics

As explained previously, both objective and subjective metrics were used. Regarding the subjective metrics, due to the lack of other solutions to use as a baseline comparison, a System Usability Scale (SUS) test was conducted, which is considered to be an industry standard. It consists of ten sentences that the testers of the system are required to answer with the Likert Scale, i.e. from one to five, where one means "strongly disagree" and five means "strongly agree". These sentences are the following:

- **Sentence 1.1:** I think that I would like to use this system frequently.
- **Sentence 1.2:** I found the system unnecessarily complex.
- **Sentence 1.3:** I thought the system was easy to use.
- **Sentence 1.4:** I think that I would need the support of a technical person to be able to use this system.
- **Sentence 1.5:** I found the various functions in this system were well integrated.
- **Sentence 1.6:** I thought there was too much inconsistency in this system.
- **Sentence 1.7:** I would imagine that most people would learn to use this system very quickly.
- **Sentence 1.8:** I found the system very cumbersome to use.
- **Sentence 1.9:** I felt very confident using the system.
- **Sentence 1.10:** I needed to learn a lot of things before I could get going with this system.

Besides the SUS test, the users were also asked to grade on the same scale whether they felt motion sickness, whether they were able to find the actions information easily, whether the scene view was relevant for the perception of accountability, and whether they thought that this solution increased the users' accountability. The corresponding sentences are:

- **Sentence 2.1:** I felt motion sickness or nauseated while using the system.
- **Sentence 2.2:** I was able to see who made the changes easily.
- **Sentence 2.3:** I was able to see what were the changes easily.
- **Sentence 2.4:** I was able to see when the changes were made easily.
- **Sentence 2.5:** The scene view increased my awareness of the state of the system at the time of the change.
- **Sentence 2.6:** The current solution increases the Accountability of the system's users.

As for the objective metrics, the moderator gathered whether or not the participants found the correct information while the system automatically logged the time it takes the user to complete each of the tasks.

5.6 Analysis

In this section, the gathered data, during the user tests, is analyzed to validate the research questions. Each result is explained and reviewed to provide more information. Section 5.4.1 explores the general demographics of the user base and sections 5.6.1 and 5.6.2 analyze the data.

5.6.1 Objective Metrics

Before starting the tasks, the users were given a brief explanation of how the system worked. Afterward, they were allowed to explore a test scene without being timed, in order to get acquainted with the system. These two stages were planned to take about 10 minutes, as most of them didn't have any experience with VR, and the ones that did, had very little experience. However, the users learned to use the system quickly and asked to move to the tasks, stating that the controls were intuitive and easy to use.

Before starting the tasks, the users were told to take their time completing the tasks, so as to mimic how the system would be used in a normal scenario. Afterward, before starting each task, the users would be instructed as to what information they were supposed to search for.

One of the metrics gathered was the amount of correct information the participants were able to acquire. The moderator would ask directly for the information that the participants were supposed to gather and see if they found it or not. Each participant was able to find the correct information in each task and reported the correct information to the moderator. The information given by the moderator prior to the task was enough for the participants to navigate the system and find all of the information without any additional aid.

Bellow on figure 5.7 are the average times it took the users to complete each of the tasks given.

All of the task completion times were considerably below the maximum allocated time. The first task took on average 35 seconds for the participants to finish, the second task around 24 seconds, the third 31 seconds, the fourth 27 seconds, and the last one 22 seconds.

Each of the tasks took about 30 seconds to complete. The first task was the one that presented the highest average completion time, taking an average of 35 seconds, suggesting that having the timeline in a hard place to reach makes navigation a bit harder. The fifth task was the one which took on average the least amount of time, only taking 22 seconds to complete. This suggests that the import node was easy to find among the other nodes in the timeline.

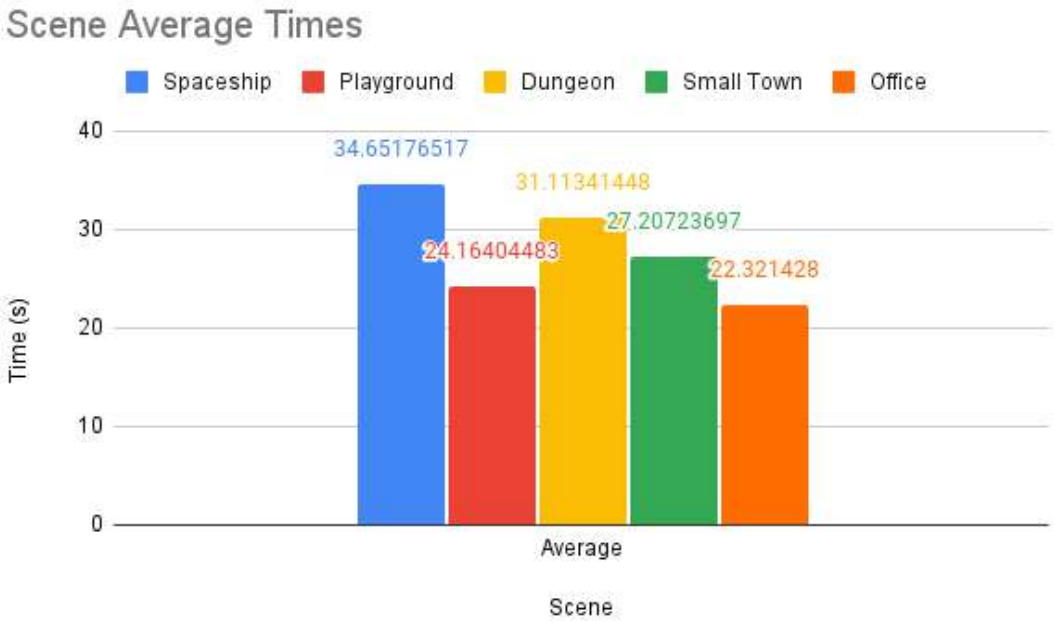

Figure 5.7: Scene Average Times

5.6.2 Subjective Metrics

Through a questionnaire, users were asked to grade how much they agreed with the sentences presented in 5.5.1. Participants were given a Likert Scale from 1 to 5 to answer our questions, with 5 being the favorable value. Answers are depicted in Table 5.5.

Table 5.5: User responses to Sentences, Median(Inter Quartile Range)

Sentence	**Median(IQR)**
S1.1 I think that I would like to use this system frequently.	5(1)
S1.2 I found the system unnecessarily complex.	1(1)
S1.3 I thought the system was easy to use.	5(1)
S1.4 I think that I would need the support of a technical person to be able to use this system.	2(2)
S1.5 I found the various functions in this system were well integrated.	5(1)
S1.6 I thought there was too much inconsistency in this system.	1(1)
S1.7 I would imagine that most people would learn to use this system very quickly.	5(1)
S1.8 I found the system very cumbersome to use.	1(2)
S1.9 I felt very confident using the system.	5(1)
S1.10 I needed to learn a lot of things before I could get going with this system.	1(1)
S2.1 I felt motion sickness or nauseated while using the system.	1(0)
S2.2 I was able to see who made the changes easily.	5(1)
S2.3 I was able to see what were the changes easily.	5(1)
S2.4 I was able to see when the changes were made easily.	5(1)
S2.5 The scene view increased my awareness of the state of the system at the time of the change.	5(1)
S2.6 The current solution increases the Accountability of the system's users.	5(1)

One important aspect of the system is its ease of use. The testers evaluated this on sentences **S1.2**, **S1.3**, **S1.4**, **S1.7** and **S1.10**. Sentences 1.3 and 1.7 affirm positively that the system is easy to use, i.e. agreeing with these sentences means that the user felt that the system was easy to use, whereas 1.2, 1.4, and 1.10 affirm that the system is not easy to use, and as such agreeing to these sentences means that the user felt the system was difficult to use. A histogram of the answers can be found in image 5.8.

Overall, the testers found the system intuitive and easy to use, which is confirmed by the values of the answers. A brief explanation of how the system works prior to beginning using the system was enough for people to get started, regardless of their prior knowledge of versioning systems or VR.

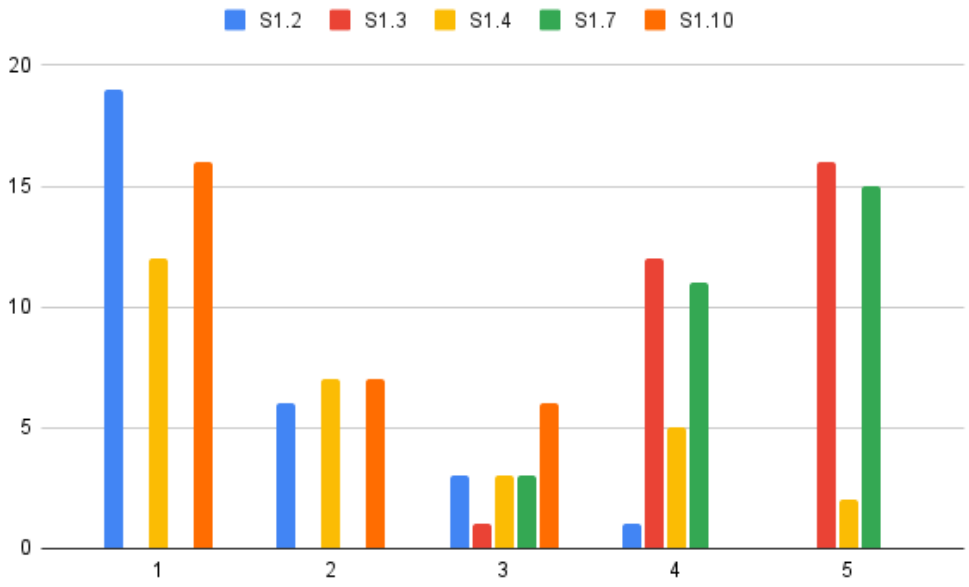

Figure 5.8: Ease of use histogram

Another important aspect is the immersion of the system. When developing an application for VR, it is important that this does not break the immersion of the user. This can be achieved by using the least amount of interfaces and using mechanisms that are natural and used outside VR. The testers evaluated this on sentences **S1.5**, **S1.6**, **S1.8** and **S1.9**, where 1.6 and 1.8 state that the system breaks the immersion, and 1.5 and 1.9 state that the system was intuitive and immersive. The results can be found in figure 5.9.

The prototype was designed to be as immersive as possible, trying to minimize the interfaces present and using intuitive gestures for the different actions supported. Analyzing the results, the testers found the different functionalities in the prototype well integrated, and that it didn't break the immersion of the system.

One common problem felt by VR users is the feeling of motion sickness, which might happen especially when the image shown in the headset stutters or lags behind the movement of the user.

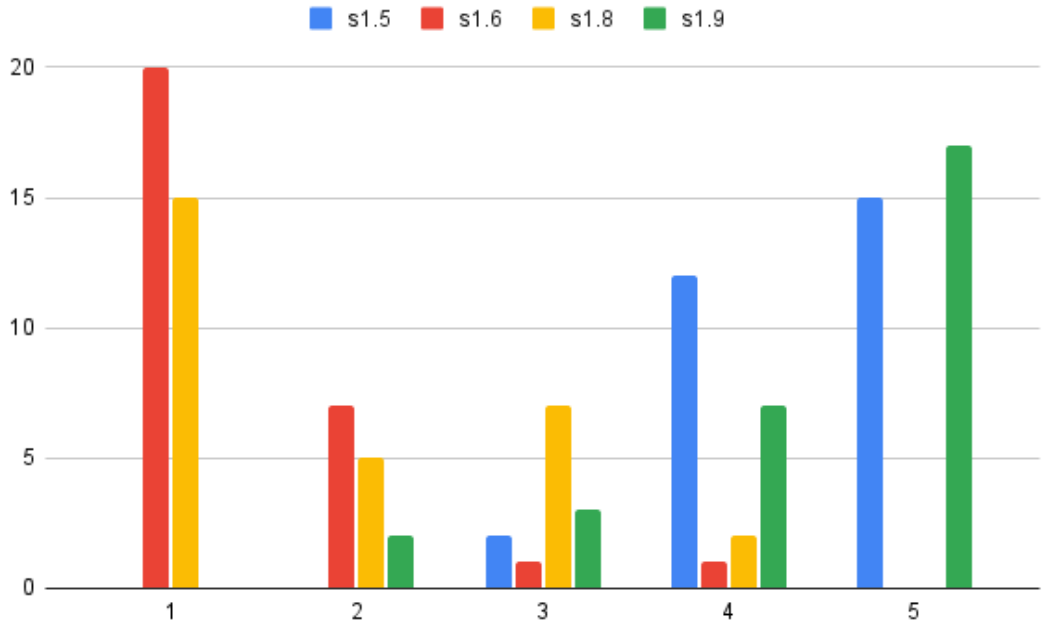

Figure 5.9: Immersion histogram

This can happen when the system is too heavy for the machine it is running on. In the case of the presented solution, the tests were run on a laptop and ran without any issues. The users were asked to state whether they felt motion sickness or not in sentence **S2.1**, where 5 means that they felt motion sickness and 1 means they didn't. The results can be found in figure 5.10.

Although no user stated they felt motion sickness, a small group of testers felt a bit of a headache and said that the image from the headset was a bit out of focus.

In sentences **S2.2**, **S2.3**, and **S2.4** the users were asked if they could see the information regarding the changes made to the scene. A histogram with the answers can be found in figure 5.11.

Prior to each task, the testers were asked to find specific information. When found, they had to say it aloud, to verify they were able to see the information correctly. Every tester was able to correctly identify the who, what, and when asked in each task.

Sentences **S2.5** and **S2.6** stated that the prototype increased the awareness and accountability of the users, respectively. The histogram of the answers to these sentences can be found in figure 5.12.

In general, the testers found that the prototype did increase both the awareness and accountability of the users in the system.

On image 5.13 is a box plot of all the sentences. Answers medians are on the extremes, meaning testers either strongly agreed or disagreed with the sentences, and inter-quartile ranges were low, meaning that the opinions were consensual.

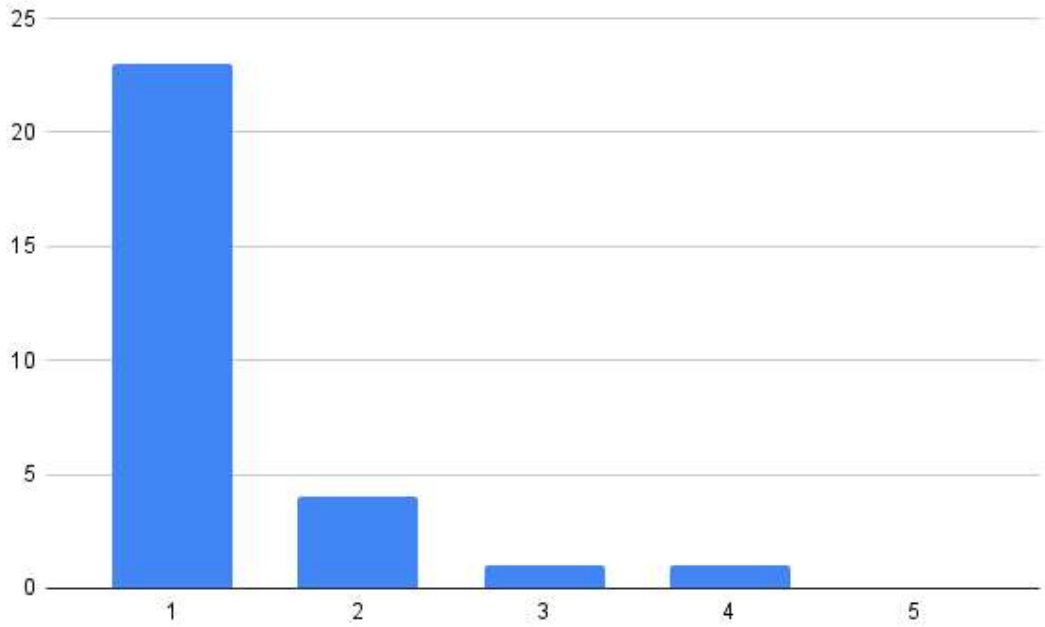

Figure 5.10: Motion sickness histogram

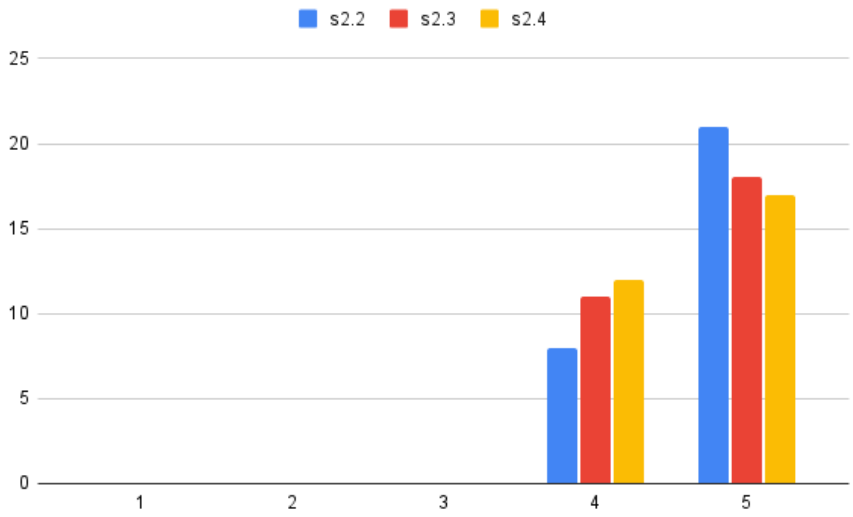

Figure 5.11: Scene Information histogram

5.6.3 System Usability Scale

To measure the usability of the system, a System Usability Scale (SUS) questionnaire was used. This is a tool that allows the evaluation of a wide variety of products and services, consisting of 10 sentences to which the respondents must answer using the Likert Scale, i.e. from 1 to 5, 1 meaning "Strongly disagree" and 5 meaning "Strongly agree". Table 5.6 shows the ranks of each

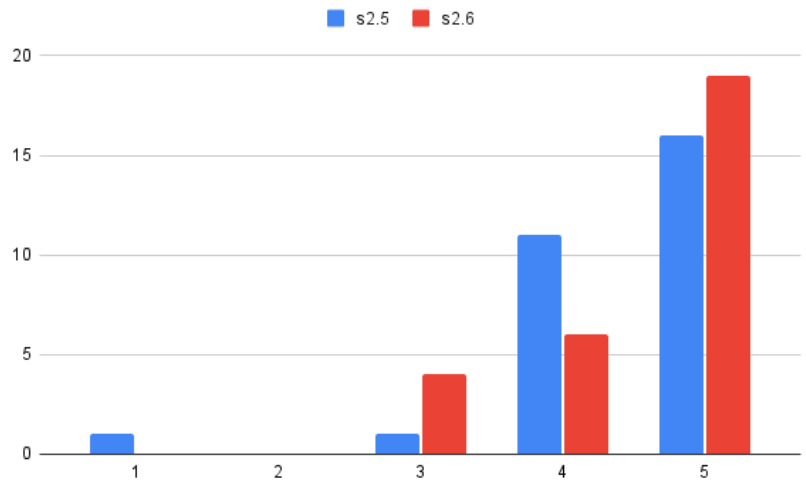

Figure 5.12: Accountability and awareness histogram

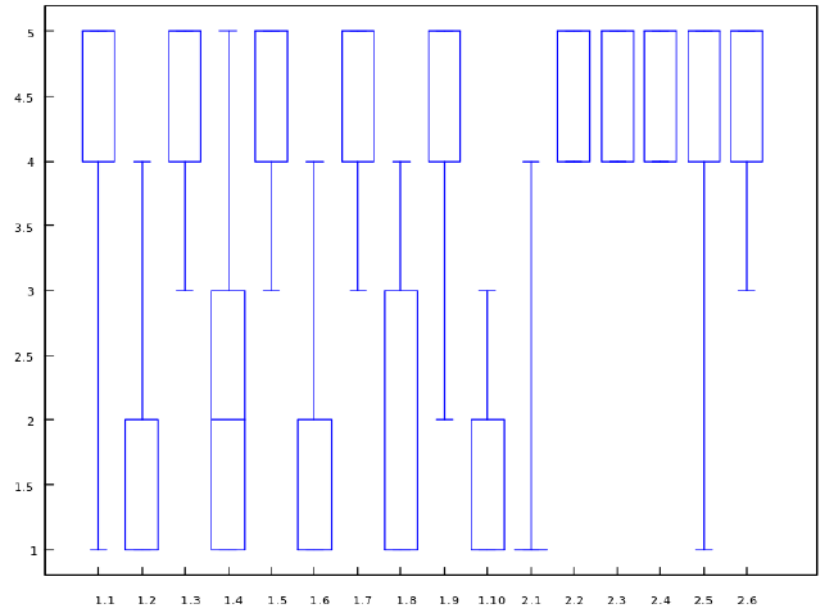

Figure 5.13: Sentences answers box plot

score interval.

The score is calculated by adding the odd rows and subtracting 5, subtracting the sum of the even rows from 25, adding these two values, and multiplying by 2.5. The formula to calculate the score is as follows $\left(\left(\sum_{i=1}^{5} S_{(2i-1)} - 1\right) + \left(\sum_{i=1}^{5} 5 - S_{(2i)}\right)\right) * 2.5$.

Applying the formula to the answers given by the participants who tested the prototype, an average score of 78.83 was achieved, which translates into a B+.

Table 5.6: System Usability Scale Score Ranking

RANK	SUS Score
A+	84.1 - 100
A	80.8 - 84
A-	78.9 - 80.7
B+	77.2 - 78.8
B	74.1 - 77.1
B-	72.6 - 74
C+	71.1 - 72.5
C	65 - 71
C-	62.7 - 64.9
D	51.8 - 62.6
F	0 - 51.7

5.6.4 Participant Feedback

At the end of the test sessions, participants were asked if they wanted to leave some feedback. Most thought the system was very intuitive and easy to use, even without the technical know-how of either versioning systems or VR.

Some participants complained that the headset was a bit heavy and impractical, considering the number of cables it had. This issue can be solved by using a lighter headset with fewer wires or, preferably, with wireless connectivity.

5.7 Discussion

The user tests described previously were used to evaluate whether or not the prototype developed answers the research question presented in chapter 1, having analyzed the results in the previous section. The overall feedback given by the participants was positive, stating that the prototype did increase the accountability of the system.

Regarding the **RQ1**, the participants stated that it was easy to find the information regarding the who, what, and when of the actions made on a given scene. Participants were able to easily identify where a specific action was in the timeline, as the cubes representing the positions of the different actions had the different transformations applied to them, representing the actions applied in that position. The fact that the panel with the information on the changes followed the participant's hand guaranteed that the information was always visible to the user. Thus, the system successfully increased the accountability of the environment, solving the issue presented in the **RQ1**.

The participants also stated that they felt immersed throughout the entire experience and that the different functionalities were well integrated. This was achieved by using as few user interfaces as possible, only having used one, the actions panel, taking advantage of affordances, which led to the participants intuitively finding the different actions of the system, with the minimum input from the guide. This solves the problem presented in the **RQ2**.

Chapter 6

Conclusion and Future Work

The amount of collaborative systems in VR is increasing and with it the need for methods to increase the accountability of its users. Although this subject has been extensively studied in desktop applications, applying these concepts to VR is not a straightforward process. Since VR differs from desktop applications in some aspects, the interface design is, by extension, also distinct.

The objective of this dissertation was to present a solution that increases accountability in immersive environments, by addressing the research questions presented in chapter 1. By analyzing the current state of the art on collaborative systems in VR and versioning tools, presented in chapter 2, it was possible to detect the lack of research on the matter of accountability in immersive environments. It also allowed for the definition and planning of the features to be developed, as presented in chapter 3.

The development phase of the prototype previously suggested is detailed in chapter 4. It explains the process of developing the prototype and the different design choices taken, and why they were chosen.

To assess whether the prototype developed in response to the research question outlined in chapter 1, user tests were conducted and the results analyzed, as discussed in the preceding chapter. A description of how these were conducted and the gathered data can be found in chapter 5. A total of 29 user studies were conducted. Overall, participants provided positive feedback, indicating that the prototype successfully increased the system's accountability.

Regarding Research Question 1, participants found it easy to locate information on the "who, what, and when" of actions taken in a given scene. The cubes representing different actions had distinct transformations applied to them, allowing participants to quickly identify the timeline and action location. The information panel that followed the user's hand ensured visibility at all times. Consequently, the system effectively enhanced accountability, resolving the issue outlined in Research Question 1.

Participants also reported feeling fully immersed in the experience, with different functionalities seamlessly integrated through the use of a single user interface - the actions panel - that leveraged affordances to guide participants through the system with minimal input from the moderator. This resolved the problem outlined in Research Question 2.

6.1 Future Work

Future research in accountability for immersive environments could encompass a broad range of topics. For instance, one promising area of investigation could involve developing more sophisticated version control systems that incorporate more comprehensive accountability features. Another possibility is to examine the potential of Artificial Intelligence (AI) algorithms in enhancing accountability in virtual environments. Additionally, researchers could evaluate the impact of various feedback mechanisms on user accountability within immersive and collaborative settings, as well as explore the effects of accountability on team performance, collaboration, and trust in virtual environments.

Other areas of research could include examining the potential of Block-Chain technology to enhance accountability in virtual environments, assessing the usability and effectiveness of different user interfaces for displaying accountability information, and investigating the relationship between accountability and privacy in virtual environments, particularly with regard to data collection and sharing. Furthermore, developing standardized metrics and evaluation frameworks for assessing accountability in virtual environments could be a valuable contribution to the field. Additionally, researchers could explore how different cultural norms and values impact the perception and implementation of accountability in virtual environments, as well as study the long-term effects of accountability interventions on user behavior and performance in virtual environments.

Having successfully implemented the versioning system, the next step to further improve the prototype would be to integrate it with an actual collaborative modeling system. This would entail some decisions not covered in this dissertation, such as thinking of a way to store this information in a decentralized architecture, or on a dedicated server for the purpose.

The current implementation of the prototype stores various actions, but doesn't store when an object is deleted. The implementation of this feature would entail finding a way to display a list of deleted objects, as they aren't in the scene and thus can't be selected like a normal object, without breaking the immersion of the system.

The prototype could also be further developed, adding some new features such as a restore function, which does not directly relate to accountability and is thus outside the scope of this dissertation. Nonetheless, it is a useful feature for a collaborative environment, allowing for the reproduction of a given state of the scene or object, without having to recreate all the steps taken previously.

One other feature that might be handy to develop is the grouping feature, where the users can create groups of objects, and the manipulation and history are applied to all the objects in the group, not having to manually edit each of the objects when you want to do the same action to multiple objects.

Lastly, another nice-to-have feature would be the possibility to comment on the changes made, which would allow users to communicate and review in an asynchronous manner.

References

[1] Features | 3D Repo BIM Online.

[2] Find what's changed in a file - Computer - Docs Editors Help.

[3] Photorealistic Design Collaboration Using VR | NVIDIA Holodeck.

[4] View activity and file versions - Computer - Google Drive Help.

[5] What is version control | Atlassian Git Tutorial.

[6] Tech Talk: Linus Torvalds on git - YouTube, 2007.

[7] Sang-Gyun An, Yongkwan Kim, Joon Hyub Lee, and Seok-Hyung Bae. Collaborative Experience Prototyping of Automotive Interior in VR with 3D Sketching and Haptic Helpers. In *Proceedings of the 9th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, New York, NY, USA, 2017. ACM.

[8] Naem Baron. CollaborativeConstraint: UI for collaborative 3D manipulation operations. In *2016 IEEE Symposium on 3D User Interfaces, 3DUI 2016 - Proceedings*, pages 273–274. Institute of Electrical and Electronics Engineers Inc., apr 2016.

[9] Clay Bavor. Project Starline: Feel like you're there, together, may 2021.

[10] Micah Blumberg. Nick Whiting from Epic Games explains the origin of XR (XR = Virtual Reality + Augmented Reality). | by SVGN.io | Silicon Valley Global News SVGN.io | Medium, 2018.

[11] Fabio Calefato, Giovanna Castellano, and Veronica Rossano. A Revision Control System for Image Editing in Collaborative Multimedia Design. In *2018 22nd International Conference Information Visualisation (IV)*, pages 512–517. IEEE, jul 2018.

[12] Scott Chacon and Ben Straub. *Pro Git*. Apress, Berkeley, CA, 2014.

[13] Yosun Chang. Sur.faced.io: Augmented reality content creation for your face and beyond by drawing on paper. In *ACM SIGGRAPH 2019 Appy Hour, SIGGRAPH 2019*. Association for Computing Machinery, Inc, jul 2019.

[14] Chien-Wen Chen, Jain-Wei Peng, Chia-Ming Kuo, Min-Chun Hu, and Yuan-Chi Tseng. Ontlus: 3D Content Collaborative Creation via Virtual Reality. In Klaus Schoeffmann, Thanarat H. Chalidabhongse, Chong Wah Ngo, Supavadee Aramvith, Noel E. O'Connor, Yo-Sung Ho, Moncef Gabbouj, and Ahmed Elgammal, editors, *MultiMedia Modeling*, volume 10705 of *Lecture Notes in Computer Science*, pages 386–389. Springer International Publishing, Cham, 2018.

[15] Mays El-Dairi and Robert James House. CONTENT CREATION. In *The VR Book*, page 223. Association for Computing Machinery, oct 2015.

[16] Thomas Erickson and Wendy A Kellogg. Social translucence. *ACM Transactions on Computer-Human Interaction*, 7(1):59–83, mar 2000.

[17] Helen Eriksson and Lars Harrie. Versioning of 3D city models for municipality applications: Needs, obstacles and recommendations. *ISPRS International Journal of Geo-Information*, 10(2):55, jan 2021.

[18] João Ferreira, Daniel Mendes, and Rui Rodrigues. Immersive Multimodal and Procedurally-Assisted Creation of VR Environments. In *2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, 2021.

[19] Adrien Girard, Yacine Bellik, Malika Auvray, and Mehdi Ammi. Collaborative approach for dynamic adjustment of selection areas in polygonal modelling. In *Proceedings of the 18th ACM symposium on Virtual reality software and technology - VRST '12*, page 205, New York, New York, USA, 2012. ACM Press.

[20] Victor Kaptelinin and Bonnie Nardi. Affordances in HCI. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*, page 967, New York, New York, USA, 2012. ACM Press.

[21] Vlasios Kasapakis, Damianos Gavalas, and Elena Dzardanova. Mixed Reality. In *Encyclopedia of Computer Graphics and Games*, pages 1–4. Springer International Publishing, Cham, 2018.

[22] Wei Ke and Lap-Man Hoi. A Graph-Based Design of an Open Database for 3D Models with Versioning. *International Journal of Information and Education Technology*, 7(5):385–391, 2017.

[23] Sotiris Makris, Loukas Rentzos, George Pintzos, Dimitris Mavrikios, and George Chryssolouris. Semantic-based taxonomy for immersive product design using VR techniques. *CIRP Annals*, 61(1):147–150, jan 2012.

[24] Liang Men and Bryan-Kinns Nick. LeMo: Exploring Virtual Space for Collaborative Creativity. In *Proceedings of the 2019 on Creativity and Cognition*, New York, NY, USA, 2019. ACM.

[25] Kazunori Miyata, Katsuhiro Umemoto, and Takeo Higuchi. An educational framework for creating VR application through groupwork. *Computers Graphics*, 34(6):811–819, dec 2010.

[26] Ali Gürcan Özkil. Collective design in 3D printing: A large scale empirical study of designs, designers and evolution. *Design Studies*, 51:66–89, jul 2017.

[27] Vasco Pereira, Teresa Matos, Rui Rodrigues, Rui Nobrega, and Joao Jacob. Extended Reality Framework for Remote Collaborative Interactions in Virtual Environments. In *2019 International Conference on Graphics and Interaction (ICGI)*, pages 17–24. IEEE, nov 2019.

[28] Danny Schott, Patrick Saalfeld, Gerd Schmidt, Fabian Joeres, Christian Boedecker, Florentine Huettl, Hauke Lang, Tobias Huber, Bernhard Preim, and Christian Hansen. A VR/AR Environment for Multi-User Liver Anatomy Education. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*, pages 296–305. IEEE, mar 2021.

[29] R Silva, J C Oliveira, and G A Giraldi. Introduction to Augmented Reality.

[30] Mike Skalnik. 3D File Diffs | The GitHub Blog, 2013.

[31] Maurício Sousa, Daniel Mendes, Daniel Medeiros, Alfredo Ferreira, João Madeiras Pereira, and Joaquim Jorge. Remote Proxemics. In Craig Anslow, Pedro Campos, and Joaquim Jorge, editors, *Collaboration Meets Interactive Spaces*, pages 47–73. Springer International Publishing, Cham, 2016.

[32] Maximilian Speicher, Katy Lewis, and Michael Nebeling. Designers, the Stage Is Yours! Medium-Fidelity Prototyping of Augmented Virtual Reality Interfaces with 360theater. *Proceedings of the ACM on Human-Computer Interaction*, 5(EICS):1–25, may 2021.

[33] Y M Tang, · K M Au, · H C W Lau, · G T S Ho, and · C H Wu. Evaluating the effectiveness of learning design with mixed reality (MR) in higher education. *Virtual Reality*, 24:797–807, 2020.

[34] Zhenhai Wang, Hongming Cai, and Fenglin Bu. Nonlinear Revision Control for Web-Based 3D Scene Editor. In *Proceedings - 2014 International Conference on Virtual Reality and Visualization, ICVRV 2014*, pages 73–80. Institute of Electrical and Electronics Engineers Inc., sep 2015.

[35] Tim Weissker and Bernd Froehlich. Group Navigation for Guided Tours in Distributed Virtual Environments. *IEEE Transactions on Visualization and Computer Graphics*, 27(5):2524–2534, may 2021.

[36] James Webb Young. *A Technique for Getting Ideas*. MCGRAW-HILL EDUCATION - EUROPE, 1975.

Appendix A

User Questionaire

A.1 Demographics Questions

Gender	Male
	Female
	Prefer not to say
	Other

Age	<18
	18 - 23
	24 - 30
	31 - 35
	36 - 40
	>40

Are you a FEUP student or alumni?	Yes
	No

A.2 Virtual Reality Proficiency

Have you had experience with Virtual Reality in the past?	Yes
	No

	1	2	3	4	5
How experienced are you with Virtual Reality?	☐	☐	☐	☐	☐

How often do you use Virtual Reality?	Less than once per month 1 - 5 times per month 1 - 7 times per week More than once per day

A.3 Versioning Systems Experience

Have you ever used Git or other versioning systems such as SVN	Yes No

	1	2	3	4	5
How experienced are you with Versioning Systems?	☐	☐	☐	☐	☐

How often do you use Versioning Systems?	Less than once per month 1 - 5 times per month 1 - 7 times per week More than once per day

A.4 System Usability Scale

This next section is composed by 10 questions regarding the usability of the system. The Likert scale will be used as metric. Please evaluate the following statements on a scale from 1 to 5, where 1 means "Strongly Disagree" and 5 means "Strongly Agree".

	1	2	3	4	5
I think that I would like to use this system frequently.	☐	☐	☐	☐	☐
I found the system unnecessarily complex.	☐	☐	☐	☐	☐
I thought the system was easy to use.	☐	☐	☐	☐	☐
I think that I would need the support of a technical person to be able to use this system.	☐	☐	☐	☐	☐
I found the various functions in this system were well integrated.	☐	☐	☐	☐	☐
I thought there was too much inconsistency in this system.	☐	☐	☐	☐	☐
I would imagine that most people would learn to use this system very quickly.	☐	☐	☐	☐	☐
I found the system very cumbersome to use.	☐	☐	☐	☐	☐
I felt very confident using the system.	☐	☐	☐	☐	☐
I needed to learn a lot of things before I could get going with this system.	☐	☐	☐	☐	☐

A.5 Final regards

In this last section I ask you once again to respond to a couple of questions with the Likert scale, and then to leave some feedback on the solution. (1 - Strongly Disagree, 5 - Strongly Agree)

	1	2	3	4	5
I felt motion sickness or nauseated while using the system.	☐	☐	☐	☐	☐
I was able to see who made the changes easily.	☐	☐	☐	☐	☐
I was able to see what were the changes easily.	☐	☐	☐	☐	☐
I was able to see when the changes were made easily.	☐	☐	☐	☐	☐
The scene view increased my awareness of the state of the system at the time of the change.	☐	☐	☐	☐	☐
The current solution increases the Accountability of the system's users.	☐	☐	☐	☐	☐

Appendix B

Consent Form

DECLARAÇÃO DE CONSENTIMENTO

(Baseada na declaração de Helsínquia)

No âmbito da realização da tese de Mestrado do Mestrado Integrado de Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto, intitulada Accountability in Immersive Content Creation Platforms, realizada pelo estudante Luís Guilherme da Costa Castro Neves, orientada pelo Prof. Daniel Filipe Martins Tavares Mendes e sob a co-orientação do Prof. Rui Pedro Amaral Rodrigues, eu abaixo assinado, _____, declaro que compreendi a explicação que me foi fornecida acerca do estudo no qual irei participar, nomeadamente o carácter voluntário dessa participação, tendo-me sido dada a oportunidade de fazer as perguntas que julguei necessárias.

Tomei conhecimento de que a informação ou explicação que me foi prestada versou os objetivos, os métodos, o eventual desconforto e a ausência de riscos para a minha saúde, e que será assegurada a máxima confidencialidade dos dados.

Explicaram-me, ainda, que poderei abandonar o estudo em qualquer momento, sem que daí advenham quaisquer desvantagens.

Por isso, consinto participar no estudo e na recolha de imagens necessárias, respondendo a todas as questões propostas.

Porto, __ de _____ de 202_

_____

(Participante ou seu representante)