Atmospheres for Virtual Architectural Works

A Framework for Creating Believable & Realistic Atmospheres in Real-time Virtual Reality

Master's Thesis

MTA 181035

Aalborg University
Medialogy

Department of Architecture, Design and Media Technology
Medialogy, 10th Semester

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Atmospheres for Virtual Architectural Works

Project Period:
Spring Semester 2018

Semester Theme:
Master's Thesis

Supervisor:
Martin Kraus

Collaborator:
Jacob Hilmer

Project Group No.:
MTA 181035

Members:
Karina K. Ebbesen
Mircea M. Pop
Rossen D. Terziev

Abstract:

Virtual architectural works can be useful in a variety of ways, serving different purposes. However, to truly experience a virtual architectural work, it has to be as close as possible to real life. Creating a believable and realistic scene in VR can be a complex process with many factors to consider. Thus, in collaboration with the architect Jacob Hilmer, we made a framework for creating a believable and realistic atmosphere in virtual reality. A set of requirements was formulated, and through participatory design with the collaborator, containing multiple design cycles, a virtual environment was created, with a user interface system that can control aspects of the environment such as time of day, location and weather effects. Furthermore, a user test was carried out in order to configure a user-friendly UI system for controlling the virtual environment.

Contents

Chapter 1

Abstract

Virtual architectural works can be useful in a variety of ways, serving different purposes. However, to truly experience a virtual architectural work, it has to be as close as possible to real life. Creating a believable and realistic scene in VR can be a complex process with many factors to consider. Thus, in collaboration with the architect Jacob Hilmer, we made a framework for creating a believable and realistic atmosphere in virtual reality. A set of requirements was formulated, and through participatory design with the collaborator, containing multiple design cycles, a virtual environment was created, with a user interface system that can control aspects of the environment such as time of day, location and weather effects. Furthermore, a user test was carried out in order to configure a user-friendly UI system for controlling the virtual environment.

Chapter 2

Introduction

This Master's Thesis project was carried out in collaboration with Jacob Hilmer, the architect from Jacob Hilmer Architecture [14], who proposed the project of a Virtual Reality (VR) system for creating believable and realistic atmospheres for architectural works.

Virtual architecture is used for a variety of purposes ranging from a design tool to a mode of communication between architects and their targeted groups [48]. Part of the communication process can often be to convey a certain type of mood through the architecture. However, the perception of moods in the environment is subjective and is mainly experienced through ambient processing [28]. Different architectural works are created with different intentions, therefore, the atmosphere varies depending on the motivation behind the architectural work. Furthermore, in order to truly experience an architectural work inside a virtual environment (VE), the environment must be as close as possible to how a real life version would be [5]. The addition of a Head-Mounted Display (HMD) to the VR simulation is considered beneficial for the levels of immersion [3]. Although VR through an HMD is possible to use with 360 degree imagery, creating a dynamic environment, where shadows and reflections are not static, as this can further increase the sense of realism and believability [47]. Therefore, our aim is to design a framework for implementing the features for visualizing atmospheres, while ensuring that the system runs smoothly in real-time VR.

Chapter 3

Literature Review

The following chapter discusses the relevance of VR technology for virtual representations of architecture. Architectural experience itself is defined and the aspects of atmosphere, believability and realism are examined.

Architectural projects have always been presented as drawings, models, and in the most recent years, as computer simulations and animations, which are important for the potential clients and competitions [5]. According to Böhme [5] the representation of the accomplished architectural projects is more important than the building itself. Architecture must be advantageous, functional and be appealing to the existing market, therefore, the representation is important. Böhme [5] argued that in order to truly experience the representation of an architectural project, one must try to recreate the virtual experience relatively to the real space being seen by a person, special importance being given to perspectival representation, 3D visualisation and binocular vision to provide distance estimation.

Hermund and Klint [13] compared the similarities between a physical space and a virtual one experienced through an HMD. The goal of the study was to find, to what extend objective and subjective attributes of an architectural space can be conveyed through VR Building Information Modelling models. Participants were shown both the real and virtual auditorium at the Royal Danish School of Architecture and were asked about their perception of it. The participants had to describe the two spaces using only three words and then fill out a questionnaire. However, they did not provide sufficient information on the virtual model or their questionnaires for it to be replicated.

Yu [48] argued that VR can be seen as an advanced solution for experiencing building, selling and decorating an architectural structure. Yu [48] continued on to provide several unique advantages of virtual estates, them being: best natural

communication mode, best convenient design tool, advanced marketing measure and fastest propagation mode. Yu [48] argued for the best natural communication mode, stating that VR technology developers can estimate characteristics, advantages and disadvantages of each plan and that provides the opportunity for wrong decision avoidance and can potentially increase market value. The author continued to argue that VR is not only a demonstration media, but can also be seen as a design tool, since it can visualise the designer's thoughts. Virtual estates can also be beneficial for market measure since it provides the possibility for clients to experience the building as close as possible to reality. Furthermore, VR can propagate product information fast and wide through online media.

Every architectural simulation is designed to be experienced by one distinctive representation or another. Virtual representations of architecture prove to be beneficial in a variety of ways, such as enhancing the viewer's sense of immersion and involvement and serving as a tool for communication and design. However, the virtual world must be experienced as close as possible to real life, for a truly architectural experience to be achieved [5]. Therefore, this project aims to create a believable and realistic environment for an architectural experience.

3.1 Architectural Experiences

The following section defines architectural experiences and examines factors that have an influence on the experience.

Rooney, Condia, and Loschky [28] hypothesized that architecture is experienced by consciously assessing it through focal object processing in the central vision and pre-consciously assessing the atmosphere with ambient spatial processing of the peripheral vision. The ambient processing determines where we are in the world and is the coordination of the senses including audio, vision and kinesthesis [28]. Rooney, Condia, and Loschky [28] claimed that architecture causes intellectual and atmospheric awareness, where peripheral view holds a great importance. They proposed that the atmosphere of an environment is perceived particularly from the ambient vision provided through the peripheral vision. Based on this, the researchers argued that being in a built environment creates a different experience from seeing an image of it. Viewing an image of, e.g., the inside of a church uses conscious attention through focal processing and can let us judge the visuals of the church, whereas being in the church, means the surroundings are ambiently processed too, providing an atmosphere that can impact posture, walking and mood in ways the image cannot. The researchers claimed that we mainly experience environments through ambient processing, making the atmosphere an important aspect of any environment.

According to Rooney, Condia, and Loschky [28, p. 15], the atmosphere is,

> "the overall emotional impression, or mood, a person feels in a built environment"

Meaning it is the subjective perception of the mood you feel when in that environment. In addition, Rooney, Condia, and Loschky [28] stated that the atmosphere is influenced by both the structure of the environment and the lighting, as the lighting can enhance or create a certain mood.

3.2 Atmosphere

When experiencing architectural representations in 3D environments, atmospheres must be taken into consideration to enhance the viewer's perception of the environment. This chapter explains how the viewer perceives atmospheres, what impact the atmospheres have on the audience and how atmospheres are affected.

Seamon [30] described the architectural atmosphere as the ineffable architectural presence that creates or enhances the ambiance that makes a building unusual or unique. However, Seamon [30] used a further distinction between the spirit of a place, as the quality of an environment infusing it with a unique ambiance, and the sense of a place, as people's largely unconscious ability to feel and sense that spirit. In this distinction, both the user and the built environment contribute to the atmosphere, meaning different users might experience different degrees of the same atmosphere or even experience different atmospheres in the same built environment.

Similarly, Böhme [6] stated that the phenomenon of atmosphere is itself something extremely vague, indeterminate, intangible and is a part of interior design, advertising and all fields related to the art of the set stage. Atmospheres surround everything, they give substance, aesthetics, they give a certain meaning and unify a variety of feelings in a single emotive state. Böhme [6] stated that atmospheres are subjective and to define them, you must expose yourself to them and experience them in terms of your own emotional state. Furthermore, Böhme [6] found that atmospheres are not purely subjective, as they are produced by someone to have a specific mood that can be experienced by everyone, regardless of recipients' moods or pre-conceptions. However, their construction is restricted to setting the conditions in which the atmosphere appears, imitating the model it is supposed to resemble from the observer's viewpoint.

According to Murdoch, Stokkermans, and Lambooij [21], 3D visualisations of indoor architecture have become increasingly common as a tool to convey impres-

sions. However, Murdoch, Stokkermans, and Lambooij [21] created an accurate simulation tool for indoor lighting, focusing on perceptual attributes of light in order to convey reality through previews of virtual illuminated environments. The researchers focused on 10 perceptual attributes divided in three groups: overall impression, perceived atmosphere and uniformity of light distribution. Murdoch, Stokkermans, and Lambooij [21] stated that it is already possible to efficiently simulate physically-based indoor lighting at qualities similar to photorealistic, however, they aimed to improve the perceptual accuracy by testing different lighting conditions. They found that the participants were able to assess the perceptual attributes through the rendered 3D visualisations of indoor environments shown on large screen displays, in a similar manner to corresponding real world environments providing baseline measurements. The crucial factors for perceptual accuracy were found to be the 3D models, the tonemapping and the display, as some displays could not show attributes such as glare and immersive Field Of View (FOV). Furthermore, the atmosphere was accurately conveyed with the exception of dimmed light conditions.

3.3 Believability and Realism

The aim of this project is to create a believable and realistic virtual atmosphere, therefore, those aspects are examined in the following chapter.

In video games, the realism is defined by the graphics and sounds, where the graphics include realistic physical light sources as well as conceptual and behavioural aspects [3]. The realistic lighting simulates the interaction between the rays from the light sources and the objects, so that shadows are cast, and light reflected depending on the light source and object materials. The conceptual realism relates to how well the virtual world and the characters in it conform to the framework and expectations from, e.g., genre and previous titles. In the same way, behavioural realism is the believability of both the characters' behaviours and how they interact with the objects and the players, based on the expected behaviours and the rules of the video game universe. This believability depends on the players' willingness to ignore the limits of the medium and accept what is being mediated as meaningful and real, and Bostan and Tingoy [3] stated that believability and realism are closely correlated, so that the more believable the VE, the higher the realism and vice versa. In addition, the believability is influenced by the sense of agency, as the players' feelings of control over the actions and their results, consistency of the behaviours and the fidelity, as the realism of the virtual world as a whole. The realism is connected to the sense of presence, the feeling of being present in the virtual world, which depends on the players' interpretations of, and

experiences in, the virtual world. According to Bostan and Tingoy [3], this realism is closely related to the immersion in the virtual world, which is higher for VR using HMDs than for monitors, as the HMDs can shut out the real world. Bostan and Tingoy [3] stated that the reason for this close coupling between realism, presence and immersion, is that when players believe they are present in the virtual world, this blurs the line between reality and the virtual world. This suggests that using immersive VR for the visualisation of virtual architecture is beneficial when creating believable and realistic atmospheres.

In computer graphics, whether or not a scene is designed to be photorealistic, the lighting must be believable to the viewers. A believable scene has to be internally consistent with lights that are studied in real life and designed to tell the story similarly to reality [2]. In order to make up for the flaws and limitations of the hardware and software and create believable scenes, Birn [2] stated that physical effects are added to the scene, simulated through texturing, lighting, and compositing. The audience never notice the light but feel it, therefore, it is important to create a mood that enhances the viewer's emotional experience. Birn [2, p. 16] said that,

> "If it looks like computer graphics, it is not good computer graphics"

Meaning that when viewers are faced with well-rendered visuals, the fact that they were made through computer simulations is not the first thing the viewers realise.

According to Mortensen et al. [20] the core elements of visual realism are realistic illumination and accurate geometric models. This project focuses on the implementation of realistic illumination, because of its relevance to realism and believability and the impact of illumination on the atmosphere.

Chapter 4

Illumination

Illumination in architectural representations has a great impact on the experience of an atmosphere [6] and on the realism and believability of the VE [3]. The following chapter describes various aspects of illumination with a focus on enhancing the believability and realism of 3D environments through lighting and shadows. This chapter also explores different ways of producing high quality and accurate renderings in real-time.

Birn [2] stated that when designing lighting, the motivation and the story behind what it represents have to be taken into consideration. This determines what quality the light has to possess and what kind of light sources are needed to achieve the desired effect. This is further supported by Ahearn [1], who stated that lighting is an important way to create or enhance the atmosphere.

Lights can have many different qualities, but the main qualities of light are colour temperature, brightness, softness, throw pattern and angle [2]. The *colour temperature* describes the colour of the light giving meaning and enhancing the mood in the scene. *Brightness* is the quality of light that depends on the exposure of the camera. Both the brightness and the colour temperature are relative to the camera adjustments. The *softness* of the light depends on parameters such as penumbra, which sets the softness of the edge of the light, decay of the light, which describes how the light fades away with distance, and soft shadows, which create the impression of diffuse light and contribute to the realism of a scene. The *throw pattern* describes how the light falls on different surfaces, shaping the light. Lastly, the *angle* of the light describes where the light is coming from, e.g., the light for a late afternoon comes from the sun at a lower angle, compared to at midday.

According to Birn [2], there are two types of lighting: direct light, which comes directly from a source, and indirect lighting that is bouncing or reflected from

illuminated objects. Within these two types, Birn [2] differentiated between the following types of light sources, each with their own advantages:

- Point lights that emit uniform light in all directions from a single point in space.
- Spotlights, which radiate light from an infinitely small point towards a certain direction, limited to a specific cone pattern.
- Directional lights, which are useful for simulating sunlight, illuminating all objects in the scene from the same angle regardless of where the light source is situated.
- Sky domes or boxes useful for simulating light coming from the sky, providing illumination all around the scene.
- Area lights that are useful for soft illumination and shadows simulated by a panel of lights with customizable shape and size.
- Lastly, physically based lights designed to simulate real world types of lights. Therefore, these physically based lights help create a more realistic environment.

According to Descottes and Ramos [7], there are six visual principles of light: illuminance, luminance, colour and temperature, height, density as well as direction and distribution. These principles address factors that are not easily measured, however, Descottes and Ramos [7] aimed to describe these principles to create a common understanding of lighting design, as lighting can be described using different terms. An example of these different terms describing aspects of lighting is that Birn [2] used brightness as a separate term, whereas Descottes and Ramos [7] have brightness as a part of luminance.

The *illuminance* affects our emotional response in the same way our instinctual fear of darkness makes us gravitate towards light. The illuminance of the sun depends on the latitude, time of year as well as the amount of cloud cover. *Luminance* is the measurement of the light intensity per unit of area, whereas brightness is what the viewer experiences. Different materials have different physical properties, and therefore, reflect light differently, e.g., smooth surfaces reflect light waves unobstructed, while textured surfaces reflect light waves from multiple angles. Similarly, contrary to the name, moonlight is not emitted from the moon, as the moon does not emit light, but reflects sunlight of its surface. The *colour and temperature* of light, shapes how the viewer perceives the surroundings and is linked to the perception of space and time. Usually, a space is remembered based on its colour and temperature, where certain colours, such as red, are perceived as warm, while, e.g., blue is considered cold, all due to cultural connotations. *Density* is defined by the

number of light fixtures and their placements, controlling the sense of movements and rhythm. The principles of *direction* and *distribution* of light govern the form of the light in terms of its shape, the characteristics of the light source and its aim, where a wide beam illuminates a larger area. In terms of the *height* of the light sources, the placement of a light source in relation to the ground, aids the viewer in understanding and exploring the surroundings, as it can be used as a reference point or to create spatial effects.

The previously mentioned principles are all impacted by the time of day, the weather and time of year, therefore, developing an application, where the user is free to adjust and change these settings, would allow the user to customize the atmosphere.

These six visual principles of light can be used to create lighting, however, another aspect of lighting not included in these principles is shadows. Birn [2] stated that shadows participate to the degree of realism a scene reveals by adding depth and contrast between elements, enhancing the shape of the objects. Contrast and brightness, are crucial for the interpretation of space, since it affects the eye's ability to perceive the differences between light and dark and influences the brain's perception of depth and complexity [27]. Birn [2] and Edensor [8] suggested to think of the illumination process as half being lighting design and the other half shadows, each being equally important. Shadows improve the content of an image by adding tones, richness and grouping elements together or even revealing objects or characteristics that are otherwise unnoticed.

For a realistic approach, it is beneficial to use soft shadows[2]. Birn [2] identified two different methods to calculate shadows, depth maps or shadow maps, which have finite resolution but are efficient and fast to compute, and raytraced shadows, which are more accurate but take more time to compute. Shadows created using raytraced soft shadows enhance the realism because the shadows are spread out and dissipate with distance.

4.1 Daylight Cycle Illumination

Rockcastle [27] stated that when experiencing architecture our visual perception is heavily impacted by the ephemeral and dynamic conditions of the surrounding environment. Architects and interior designers often use daylight to sculpt and shape an indoor environment [27]. Changes in the sky, time of day, time of year are variables that can alter the perception of an indoor environment [27, 5] and an outdoor environment [16]. The conditions of light and shadow are the primary ones that shape the sensual interpretation of the architecture [27, 8]. The visual

effects of daylight, such as shadow, depth, contrast and texture are considered crucial factors for the narrative of the design and the mood the narrative can transmit [27, 5, 19, 8, 2].

Birn [2] stated that creating daylight illumination for any time of the day, takes three elements: direct sunlight, soft fill light, representing light from the sky, and indirect light, simulating light that has bounced off surfaces in the scene.

The most important light in the scene is the sun, creating *direct sunlight*, casting strong shadows and being the brightest of all the lights present in the environment. It can be implemented as a directional light along with raytraced shadows to assure accurate shadowing of all elements in the scene. The shadows from the sun are the most important in the outdoor part of a scene, where the length of the shadows allow the prediction of time, e.g., longer shadows being cast by a low sun, meaning at a late time of day [7]. Birn [2] argued that values for the light angle of the shadows are between one and five degrees during a whole day, because shadows from the sun can become considerably softer on a cloudy day or at sunset, while the shadow colour parameter of sunlight should be set to pure black.

Birn [2] indicated that it is useful to use a second light to represent the spill from the sunlight, as is using a copy of the sunlight with the same direction, but with much softer and dimmer shadows, to provide more realistic lighting and shadows. Around sunset or sunrise, the saturation of the spill should be changed to deep orange or rich red, while for midday the light can be overexposed and desaturated, having an almost white colour. Implementing these colour changes over time, can add to the realism of the scene [2].

To add the realistic *soft fill light* from the sky illumination to the scene, the gradient of the sky blue should vary from the horizon line, where it is a light blue, to a much deeper blue at the top of the sky [2]. According to Birn [2], sky illumination can be simulated using an array of lights coming from different locations, using dome lights, using a shader for both sun and sky illumination, or using Image Based Lighting (IBL) to illuminate a scene with a High Dynamic Range (HDR) image. Birn [2] stated that the best way to enhance the realism would be to use IBL with a texture map matching the simulated environment, as this allows for improvements in the reflective surfaces by using different maps.

Realistic daylight cannot be accomplished without *indirect light*, through either bounced lights or Global Illumination (GI), which automatically implements the indirect illumination. In addition, Birn [2] stated that for indoor indirect lighting, GI provides the easiest and most realistic simulation, allowing the colour and brightness of the surfaces in the scene to contribute to the illumination of the surfaces around them. Therefore, setting up extra lights is not needed if GI is implemented.

The daily cycle of light and dark governs the circadian rhythm, allowing the use of the changing colours and patterns from the natural daylight and the position of the sun to tell time of day and month [7]. The path of the sun changes the perception of its distance from Earth, despite the sun staying approximately 149 million kilometres from Earth [7]. Here, the horizon serves as the baseline for measuring the height of the sun throughout the day, where this height is linked to the perception of the time of day. In particular, sunrise and sunset are considered more intimate and comforting, as the sun appears warmer and closer.

According to Hempe [12] and Halawani and Sunar [10], the intensity of the sunlight varies based on date, time of day and location on Earth, with the lowest intensity during the winter season in the Arctic and the highest at noon in the Tropical region. As the sky gets its light from the sun, the position of the sun changes the colour of the sky. The sky appears blue during noon as the shorter wavelengths are scattered more effectively than the longer wavelengths, such as red and yellow. However, at dusk and dawn, light rays are scattered by particles in the air, e.g., dust and water vapor, giving the sky an orange-red tint.

According to Wang [46], a realistic sky greatly enhances the realism of the VE, although the complex mechanisms involved make it difficult to render a realistic sky. The simplicity of the skybox technique does not allow simulations of different realistic weather conditions. Therefore, Wang [46] proposed a new physically based method to model and render for different weather conditions. The new method considered atmospheric scattering model and refraction, using a path tracing algorithm, as Wang [46] stated that traditional rendering methods are not suitable for large outdoor spaces. The author used the scattered volume model and path tracing to accelerate calculations of light coming from the sky. For future work, Wang [46] stated that the realism could be enhanced by adding real-time rendering of clouds.

Including different weather conditions and their effects on the VE, greatly improves the viewer's perception of the VE [12]. Skyboxes can easily be implemented using images of the real sky and provide realistically looking results, however, this results in a more static sky. For a dynamic sky, there are two common types of clouds: cumuliform and stratiform clouds. Cumuliform clouds appear volumetric and have irregular densities with flat bottoms and rounded tops. Hempe [12] implemented cumuliform clouds using metaballs and rendering them using splatting, for a computationally cheap approximation of the clouds. The stratiform clouds are flatter and have varying thicknesses, but their flatness means they can be approximated using textured layers, which can be rendered using a sky dome with scrollable 2D textures to make the sky more dynamic. The textures can either be images of real world clouds or dynamically generated from noise functions, which have the advantage of being adaptable to changing lighting and weather conditions. Apart

from clouds, rain is another important weather effect and the rainfall consists of droplets which interact with the VE, splashing onto the surfaces and causing materials to change appearances as they get wet, affecting how the materials reflect light [12].

4.2 Real-time Rendering of Illumination

Computer graphics illumination models that use physically accurate calculations increase the computational complexity, whereas, illumination through simplified models, reduce the computation [10]. However, the simplified models also provide a lower realism [10], and according to Hempe [12], realistic real-time rendering of VEs increase user acceptance and are suited for testing virtual prototypes. Therefore, we explore real-time rendering techniques for different degrees of realism for scenes in VR.

Hempe [12] focused on techniques for real-time rendering of realistic and dynamic lighting for outdoor VEs. The author stated that compared to GI models, local illumination models simplify the lighting conditions to increase performance. However, these simplifications, such as ignoring scattered light and adding ambient light uniformly in the VE, mean that the resulting lighting differs from real-world lighting as it is not physically accurate.

In order to implement physically accurate lighting, HDR rendering is needed to represent the high ranges of lighting intensities commonly found in daylight scenes [12]. This point was also discussed by Voloboi et al. [45], who stated that real life intensities of light can be accurately represented by floating point values, which are part of the construction of HDR images. Contrarily, Low Dynamic Range (LDR) rendering has a range of 8 bit per colour channel, which is too low to fully represent the contrasts of high intensity range lighting conditions. Nevertheless, it is possible to map HDR images to 8-bit LDR displays using tone mapping to display the HDR colour. According to Okura, Kanbara, and Yokoya [24], HDR tone mapped images allow users to see textures with various radiances. Okura, Kanbara, and Yokoya [24] claimed that it is possible to implement a fully real-time GPU based computation of HDR imaging using an HMD, where the intensity in the scene can vary depending on the user's view direction. Furthermore, the author claimed that HDR images can be used for immersive HMDs, IBL and applications that use spherical images and that adding tone mapping methods to the process improves the naturalness of the scene.

Hempe [12] and Mardaljevic [18] stated that simulating light from the sky in the real-world atmosphere using fog improves the appearance but is insufficient. In-

stead the authors recommended using atmospheric and volumetric light scattering for realistic perception.

Atmospheric light scattering is a dynamic lighting model capable of simulating outdoor lighting at any time of day in real-time [12]. Atmospheric light scattering factors in:

- Aerial perspective, to make objects in the distance appear paled and colour shifted compared to those closer to the viewer.
- Absorption by particles in the atmosphere, affecting the amount of light reaching the viewer.
- Out-scattering, when light rays hitting a particle are scattered to diffuse directions, reducing the amount of light reaching the viewer.
- In-scattering, when light rays hitting a particle are scattered in the direction of the viewer, increasing the amount of light that reaches the viewer.

Volumetric light scattering, scattering light through occlusion, creates the effect of light shafts due to the differences in brightness between the illuminated and occluded areas [12].

Another aspect that greatly increases the realism of the VE is realistic real-time performance shadows, however, there are two categories of techniques addressing different lighting types: direct and indirect light shadowing. *Direct light shadowing*, is for shadows on directly illuminated surfaces [12]. For VEs with dynamic objects and/or dynamic lighting conditions, the shadows need to be calculated in real-time. Here, the image-based technique of shadow mapping is suitable, as it is less sensitive to scene complexity compared to other techniques, however, a common problem is the effect of the shadow map resolution on the rendered quality of the shadows. Hempe [12] suggested combining multiple methods to hide typical shadow artefacts from low resolution shadow maps through simulated soft shadows. *Indirect light shadowing* stems from the ambient light, which is assumed to uniformly illuminate objects, while in reality, occlusion by nearby surfaces changes the amount of ambient light [12]. For a realistic simulation of ambient lighting, Hempe [12] recommended using ambient occlusion, as it approximates the amount of ambient light on partly occluded objects. The effect is subtle but can improve the realism of the VE dramatically. Additionally, as ambient lighting is independent from the light direction, it can be precomputed and applied to the textures.

According to Ahearn [1], real-time lighting has recently begun being calculated per vertex, manipulating the geometry in real-time, instead of the rendered pixels. This per vertex real-time lighting adds a smoothness and high degree of detail to the experience, and improves the function of normal maps [1]. This is due to the

per vertex lighting creating a gradient on the polygon face by taking the brightness value for each vertex [1]. However, per pixel lighting is more accurate, as the lighting is calculated for every pixel, not vertex [1]. This implies that the per vertex lighting is the computationally cheapest light rendering method.

4.2.1 State of the Art Real-time Illumination

The following section explores state of the art regarding the use of illumination and its effects on the users of the system.

According to Slater et al. [32], users experience the virtual world through an egocentric point of view within an immersive VR system, meaning their bodies appear to be inside the VE. Such immersive systems can often be found in experiments involving life-threatening situations or psychotherapy purposes, since, although the users know they are not in actual danger, they still respond to the scenario as if it was real. A response to a scenario as if it was real within a virtual world can be defined with the concept of presence. In their paper, the researchers consider the impact of visual realism on the sense of presence. Visual presence has two components: geometric realism and illumination realism, where geometric realism refers to the how real the virtual objects look, while illumination realism refers to the fidelity of the lighting model. Both components can have a dynamic and a static aspect. An object may look statically realistic, but with dynamics the perception of the realism may change. Good static lighting can be achieved with different methods, one of which being radiosity, which precomputes all diffuse interreflections, but then shadows may not behave in a realistic way. The researchers compared two different styles of rendering, one being per-pixel local illumination without any shadow effects and the other using recursive ray-tracing that included rendering shadows and reflections. Slater et al. [32] tested with 33 people, where participants were placed in a virtual room with a pit inside it. The results showed participants felt a higher level of presence within the virtual pit room when there were shadows and reflections.

Yu et al. [47] continued the research by Slater et al. [32], and here they found that dynamic shadows provided a higher level of presence within a VE. However, Yu et al. [47] hypothesized that the overall higher illumination quality of the raytraced scenario, might be the reason for the higher impact. The researchers compared two scenarios, where the first included standard OpenGL Gouraud-interpolated shading with dynamic shadows and umbras, while the second relied on GI that featured soft shadows and mirror reflections. The researchers hypothesized that since both scenarios included dynamic illumination, any difference would come solely from the overall illumination quality. The experiment was conducted with 21 people, using a CAVE system where participants were placed in a virtual library,

where they could only see their avatar in shadows and reflections. Furthermore, participants had to examine books and after one minute the books started to fall from shelves and later a virtual boy appeared in the room and floated around in circles three times. The results showed no differences in presence between the two conditions, which lead to the finding that dynamic changes to shadows and reflections were responsible for the higher level of presence. However, Slater et al. [32] used immersive VR with an HMD for their experiment, while Yu et al. [47] used a CAVE system, therefore, the validity of the conclusion might be biased.

Halawani and Sunar [10] developed methods for photorealistic images with high realism, resulting in excessive computational time problems. When modelling the sky, Halawani and Sunar [10] and Kolivand and Sunar [16] used a sky dome in the shape of a hemisphere with mathematical equations. In order to compute GI Halawani and Sunar [10] used Radiosity with a hemicube computation calculation, which is a progressive refinement technique. Their results suffered from a low frame rate even with low polygon counts [10]. Halawani and Sunar [10] found that the number of Frames Per Second (FPS) are inversely proportional to the complexity of the scene. They did not use specular reflection, refraction or texturing. However, Kolivand and Sunar's [16] system was able to work in real-time, although, their system did not include shadows, which would have added to the realism.

Natephra et al. [22] implemented a prototype system providing realistic visualization of indoor lighting conditions using an interactive and immersive VR environment that allowed for changing and visualizing lighting scenes and creating different scenarios. The interaction was made through HMD, keyboard and mouse. Natephra et al. [22] created a 3D model and imported it in Unreal Engine. The built-in functions of the game engine were used to create indoor lighting and daylight. The prototype system offered interaction in real-time. The authors implemented first-person movement to allow the user to move in VR, and the User Interface (UI) controls allowed for customization of different parameters such as time, to observe the dynamics of sunlight, lighting fixtures, types, orientation, intensity, and colour temperature of the bulbs. In addition, the UI allowed changes in the location of furniture, textures and materials of indoor envelopes. The GI algorithm in the Unreal Engine was used to automatically produce lighting to illuminate the scenes and calculate detailed shadows at runtime. Directional light was used in order to simulate sunlight, and visual scripting was used to automatically control the yaw and pitch of the sun and the position of the sun based on the time and the season. Natephra et al. [22] found that the system delivered a semi-realistic perception of artificial and day lighting. The system also generated realistic and false-colour scenes at runtime while performing modifications according to the user's desires, which can be very time-consuming using conventional lighting

simulation applications. Natephra et al. [22] included future improvements and requirements such as: HMD's with high resolution displays; effects of different sky conditions that influence outdoor and indoor illumination over time; and improvement to the conventional input devices, as mouse and keyboard are difficult to use while wearing an HMD. In terms of limitations they encountered problems related to the use of hardware and the developing tool. They found that factors such as the FOV and the pixel density of the headset influence the accuracy of perceiving lighting phenomena, such as glare, brightness, and darkness. Furthermore, according to Natephra et al. [22] using the Unreal Engine physics engine for simulating accurate and realistic glare was not possible. These technological limitations constrained the system to semi-realistic, however, the real-time interactive indoor lighting simulation using an HMD was still deemed a helpful design tool. The usability of such VR systems as a design tool is also supported by Yu [48].

Mortensen et al. [20] presented a method for rendering immersive VR using the CAVE system with real-time GI. Real-time GI gives the scene a more realistic look but raises the problem of computational capability. Mortensen et al. [20] found two possible solutions: constructing an ad hoc system with support for tracking and synchronization, display management and others, or integration of the GI within a more general rendering system such as CAVELib. Therefore, they used the Virtual Light Field paradigm as a solution, as they believed that it allows for realistic illumination of VEs. This method was applied on GPU because of the need to provide an efficient light transport. Mortensen et al. [20] stated that the core elements of visual realism is realistic illumination and accurate geometric models. In terms of the realistic illumination, ray tracing and radiosity are often used to provide partial GI for VR rendering, but these methods are quite low in terms of the FPS they can provide when complex scenes are presented. Other methods include path tracing and photon mapping, which offer a more complete illumination solution, but they do not exceed the real-time rendering. Precomputed Radiance Transfer can provide an approximation to GI for static scenes and Mortensen et al. [20] applied this method to the rendering in their CAVE system. In addition, the static scene was illuminated by dynamic environment maps for realistic rendering. The advantage of this system is that the rendering is independent from the complexity of the scene, however, it is limited to a static scene.

Similarly to Natephra et al. [22], our aim is to implement realistic daylight illumination in an immersive VE, including interactive controls for sun position and changes to the sky conditions over time. Unlike Mortensen et al. [20], our aim is to investigate if this can be done in real-time for immersive VR, as this provides more realistic illumination but is also more computationally heavy, as Halawani and Sunar [10] found.

Chapter 5

Design & Evaluation Cycles

To create a realistic and believable atmosphere, we collaborated with the architect Jacob Hilmer from Jacob Hilmer Architecture [14] through the principles of participatory design [31], where the architect provided requirements for the system and gave feedback on the prototypes developed through the design cycles.

The collaborator's feedback was then used to re-design and improve the system, serving as the main evaluation of the system as he proposed the project and is the intended end user.

The requirements from the collaborating architect are:

- Real-time rendering of believable and realistic atmosphere in a VE through:
  - Day and night illumination.
  - Weather changes.
- Customizable:
  - Location.
  - Time of year.
  - Time of day.
- UI for customizing all the above in real-time while in VR.

The requirements obtained from Chapter 4 are:

- An HDR camera with a tonemapping effect.
- Real-time rendering with either the per-vertex or the per-pixel method, depending on the performance impact.
- A satisfactory level of performance, which is a steady frame rate of minimum 60 FPS [9, 26].
- A daylight cycle illumination using a direct light, fill lighting and indirect lighting.
- Direct light shadowing, which is the implementation of dynamic and soft shadows.
  - With the use of shadows maps for the creation of shadows since it is more suitable for real-time implementation.
- Indirect light shadowing, which is the implementation of ambient occlusion.
- Automatic changes to the intensity of sun based on the time of day.
- Volumetric light scattering.
- The implementation of a realistic sky with:
  - Colour changes based on time of day.
  - Atmospheric scattering.
  - A dynamic sky with volumetric or flat clouds.

Throughout this chapter, the aim is to achieve these criteria, with the main focus on the collaborator's criteria.

As part of the collaboration, the architect provided a scene. The scene involves a trekking cabin in Amber Road at the Latvian coastline, which is meant to represent the cellular structure of amber, glowing and warping the light, welcoming hikers and inviting them inside to the cave like interior with wooden surfaces and a fireplace. The cabin is supposed to give of a cosy feeling, where the amber representation creates light reflections in the sea and shelter for hikers.

We focus on the indoor scene with the trekking cabins, as a starting point for creating a realistic and believable atmosphere using lighting within a VR scene. Since the architect made the model with a specific atmosphere in mind, we collaborated with him to ensure that the end product was believable and realistic to him, while providing the desired atmosphere.

The first step was to decide which program to use for the creation of the VR scene. We considered the two popular game engines Unity and Unreal Engine, comparing them to each other on a variety of factors such as: graphics, usability, familiarity and VR integration. There are no big differences between the two engines in terms of their graphics or performances, and it depends more on preference [23]. Examples of this can be seen in Figures 5.1 and 5.2, which show the same approximate scenes rendered using the Unreal and Unity engines.

Figure 5.1: The same approximate indoor scene, modelled and rendered by Oculov [23] using Unity 5 (left) and the Unreal Engine 4 (right).

Figure 5.2: This shows the same approximate outdoor scene, again modelled and rendered by Oculov [23] using Unity 5 (left) and the Unreal Engine 4 (right).

For the usability, Unreal has a different and more limited scripting option using Blueprints visual scripting system, instead of standard shader scripting as in Unity

[25]. In terms of familiarity, the project group has more experience using the Unity scripting style and, furthermore, the Blueprint system can have limitations that will be time consuming to overcome. In addition, the Unity asset store has many free assets available and is also bigger than the Unreal asset marketplace, which is to be expected as the Unreal store is younger [25]. Both engines support VR integration, although Unreal is more focused on 3D games than Unity, which is normally better for 2D games but also supports 3D [25].

The collaborator did not have a preference, as he did not see any major differences between the two engines either. Therefore, we chose Unity based on personal preference, as the graphics and performance are not significantly different, and we have previous experience scripting and creating environments for VR with the Unity engine.

5.1 First Cycle

The following section explains the steps for the first cycle of creating a virtual scene in Unity, which include import to Unity, render settings, light setup, use of assets and collaborator evaluation. The scene in this cycle is made for a desktop version, to allow for a fast and easy testing of features.

5.1.1 Importing to Unity

There were initial issues with importing of the indoor scene into Unity. The original scene was provided in Autodesk 3ds Max and used unique plug-ins for material rendering such as, Cordana and V-ray. The scene had to be exported as an FBX files in order to be imported into the Unity game engine. However, the 3ds Max FBX export does not support the conversion of third party materials, thus the scene had to be manually modified to use Autodesk Standard materials only. In turn, this procedure reduced the quality of the textures that the collaborator had previously used and that required some modification inside the Unity engine.

In particular the beach terrain itself was recreated using the Unity terrain creation, since the original terrain had graphical artefacts. Normally the artefacts could not be seen from a standard rendered image created from a fixed point of view by an Autodesk software, but since the VR scene allowed the user to navigate to some extent, the artefacts became visible. In addition, the texture used for the sand was replaced by a higher quality version, since the original one was pixelated, and a new material was used for the rocks in the scene. The texture for the cabin wood

was changed to be smoother and more reflective, while the windows in the cabin were given a custom glass material.

5.1.2 Render Setup

The first step to making believable visuals inside Unity was to prepare the render settings. Unity offers two workflows for rendering: gamma and linear. Textures usually come in already gamma corrected states, while shaders usually expect linear colour space, thus gamma-based values lead to inaccurate results [40]. Working in linear colour space provides more accurate rendering than gamma space [40]. The linear render mode assumes that the textures are in gamma colour space. Afterwards, Unity uses the systems GPU's sRGB sampler to crossover from gamma to linear colour space. Finally, the inputs are passed to the shader where the calculations are done [40].

Considering the input values are different for the two render modes, the lighting computation provides different results. In Figure 5.3 the difference between linear and gamma workflow can be seen.

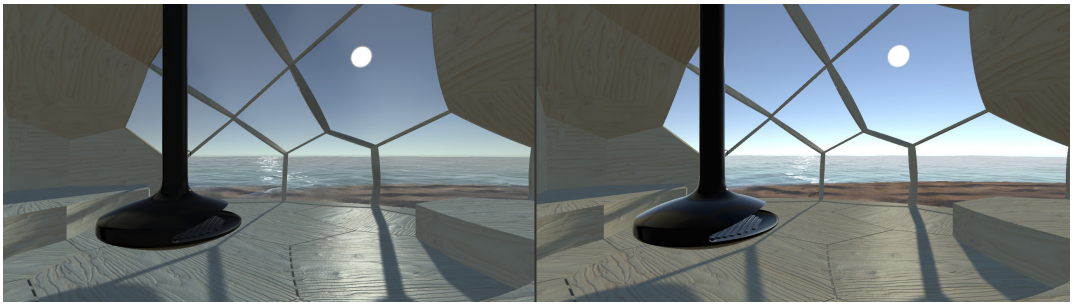

Figure 5.3: Linear workflow (left) and gamma workflow (right).

In linear workflow the light radius appears larger and the lights edges appear more clearly. The colour intensity is also impacted as the high luminance areas appear brighter in gamma workflow than in linear. Furthermore, colour blending is also impacted since linear workflow computes it in a mathematically correct way and provides correct results, while in the gamma workflow nonlinear colours get blended together, which is not mathematically correct. Therefore, the scene was set to use a linear workflow for its physically accurate math when performing lighting and shading calculations [35].

The camera within Unity was set to be an HDR camera, to avoid clipping lighting values and emissive surfaces with brightness higher than one, in order to reflect the light levels more accurately. However, for Unity to be able to display the HDR lighting properly, a tonemapper has to be introduced to the scene by creating a post

processing effect on the camera. The tonemapper maps the colour values from the HDR to LDR. The use of an HDR along with a tonemapper allows for dynamic effects such as the simulation of the natural adaptation to light when entering or leaving a dark area [41].

Unity offers two rendering techniques: Forward Rendering and Deferred Rendering. In *Forward Rendering*, each object is rendered for each light that affects it and the more lights affecting each object, the slower the rendering performance becomes. Therefore, each object might be rendered with multiple passes depending on how many lights are within range. The technique renders a set number of the brightest lights affecting each object in per-pixel mode. Afterwards, up to four point lights are calculated per-vertex and all other lights are rendered as spherical harmonics, which is faster, but less accurate. Thus, Forward Rendering can be fast, can handle transparency quickly and allows for the use of multi-sample anti-aliasing, which is not available in the other rendering technique. In *Deferred Rendering* the rendering cost of lighting is proportional to the number of pixels that the light illuminates, instead of the number of lights themselves. Deferred Rendering first renders the geometry of the scene into a 2D image, after that it uses the depth and the normals buffer to calculate the lights and shadows into the final image. Therefore, the performance is not dependant on the number of lights present in the scene, however, it generally requires more powerful hardware.

We chose to use the Forward Rendering technique because we have a low number of lights in the scene, thus performance wise it is not heavy and in terms of quality, Forward and Deferred Rendering appear similar.

5.1.3 Light Setup

After adjusting the render settings it, was time to focus on lighting the scene.

There are multiple strategies for lighting virtual scenes. A typical scene would consist of three lighting components: direct lighting, fill lighting and indirect lighting [34, 2]. Figure 5.4 demonstrates the previously mentioned components in use.

Direct lighting is the light that originates from direct sources of light, such as the sun, lamps, etc.. Fill lighting is the light originating from the sky. Indirect lighting is the light bounced off from objects in the scene.

There are multiple ways to create and combine these components, which in turns provides a variety of setups. The setups can vary in complexity and performance cost, thus, for this project we chose the setup that can deliver satisfying quality visuals with a stable performance.

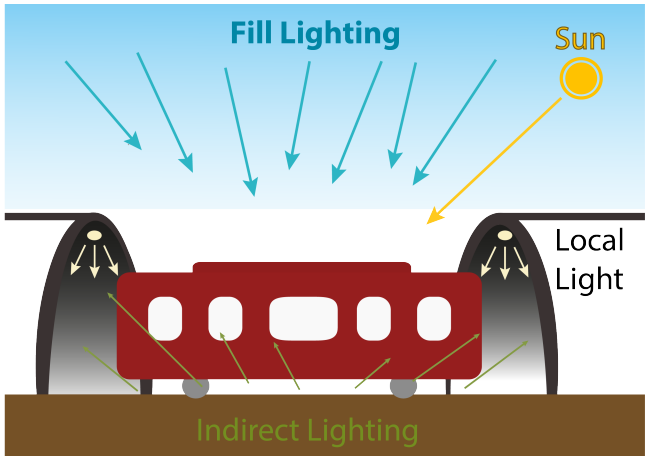

Figure 5.4: A scene illuminated by the sun and local lights as the direct light sources, the sky as fill light and indirect light from the light bouncing off surfaces in the environment.

The scene consists of a dynamic day and night cycle that the user can change, therefore, real-time light performance is optimal. The real-time lighting will allow the scene to get real-time specular light and real-time shadows, which in turn contribute to the believability. However, creating a real-time illuminated scene can be computationally heavily, which is why at the end of each cycle, the performance is checked to see if it is at a satisfactory level for the collaborator. The performance improves the experience and it grows proportionally according to the FPS, where, as previously mentioned in the requirements, a satisfactory level would be a steady frame rate of minimum 60 FPS [9, 26]. During the implementation, the performance of the scene was continuously checked by making sure the FPS was at a satisfactory level.

For the creation of a hemisphere light for the scene procedural skybox materials are preferable to cubemaps [33]. Procedural materials in Unity are materials with textures that can be generated at runtime instead of being predefined and stored. The script that generates the texture can also be made to modify the visual properties of the material during runtime [41]. These types of materials can serve well as the material for the skybox, since then the colour properties of the sky itself can be adjusted to match the sky colour at that time of the day.

The indirect light in Unity is created using the GI function. GI makes the VE seem more realistic, as it allows for objects to affect each other's appearances through the light that is reflected off them [38]. Computing GI in real-time is too costly to be considered a viable option, thus it usually calculates the indirect light for objects and surfaces that are known to be static in the scene, since the objects do not move the indirect light will be correct during runtime [38]. The process of precalculating and storing these calculations is called baking [38]. The extra computation power

given from this one-time precalculation is channelled into the creation of more realistic soft shadows from the indirect lighting [38]. An example of the effect of GI can be seen in Figure 5.5.

Figure 5.5: The scene without GI (left) and with GI (right), where the addition of indirect lighting from GI makes the inside of the cabin in shadow more visible, as the shadows are more illuminated.

In addition, Unity offers a technique called Precomputed Real-time GI, which works similarly to a standard baked GI, but does not just precompute how the light is bounced only during startup [38]. The technique precomputes all possible light bounces and encodes the information at runtime for later use [38]. Thus, whenever a change is made for the position and direction of a light source, the indirect lighting updates accordingly [38].

Shadows are an important aspect of creating a believable scene, since they bring out the scale and clarify position of objects [43]. Shadows are created by Unity by emitting a ray from the position of the light source and whenever that ray hits an object it does not travel further, and anything situated behind that object is not illuminated [43]. Unity has a depth buffer system, which tracks the surfaces that are closest to the light source and holds the information on which areas receive light and which receive shadows [43]. The depth maps created by this depth buffer are also known as Shadow Maps and the shadows created by the Shadow Maps can be distinguished into two types, hard and soft shadows [43]. Hard shadows are less computationally heavy, but have aliasing effects on them [43]. Soft shadows tend to reduce the jagged edges and look smoother at the cost of heavier computational power [43]. The two types of shadows can be seen in Figure 5.6.

For this project we decided to use soft shadows as they contribute to a higher graphical quality for our scene.

Another technique that can be applied to increase the believability of the scene is the addition of Light probes and Reflection probes. Light probes work similarly to lightmaps, but instead of baking the light that is hitting a surface, they bake the parameters of the light passing through the empty space [39]. They have two main uses when implemented in a scene, the first being providing high quality lighting

Figure 5.6: Soft shadows turned on (left) and hard shadows turned on (right), where the hard shadows have more jagged edges and the soft shadows look smoother.

on moving objects in the scene and the second is providing light information for static scenery when the scene is using a level of detail system [39]. Although the indoor scene does not need the addition of light probes, because of its simplicity and small scale, having light probes is considered beneficial for the believability of larger scenes. Reflection probes are Unity's improved version of reflection mapping [42]. Reflection mapping is a technique, often used in games, that simulates reflections from objects and it assumes that all reflective objects in the scene can see the same surroundings [42]. Unity improves on this technique with reflection probes, which sample the VE at different locations [42]. The effects of a reflection probe can be seen in Figure 5.7.

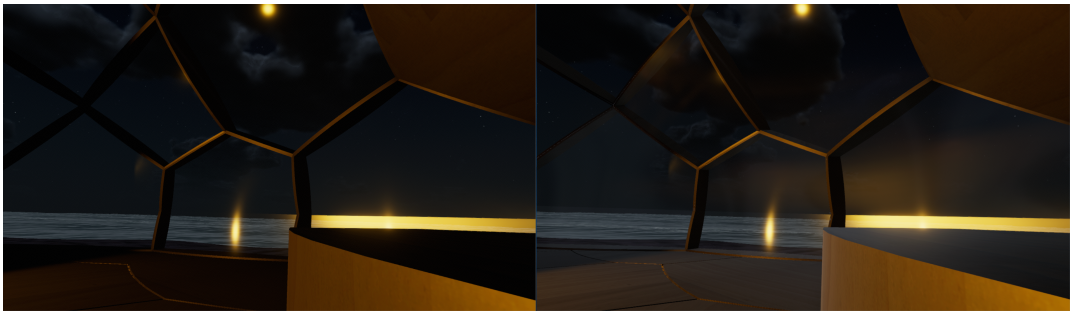

Figure 5.7: The scene without reflections (left) and with reflection from the reflection probe (right).

When an object passes near a probe, the reflection sample of the probe can be given to the reflection map of the object, where the reflection map is represented as a cubemap of the local surroundings [42]. When multiple probes exist, Unity interpolates between them and gradually changes the reflections of the object passing by [42]. We used reflection probes in our scene to simulate realistic light reflections and enhance the realism.

5.1.4 Post Processing Effects

Post processes are view dependent rendering effects that are layered on top of the rendered virtual scene before generating the final render. Although they are not necessary for achieving a believable scene, they can still contribute with enhancing the VE even further. There are multiple components that can impact the scene heavily, some of which are: anti-aliasing, ambient occlusion, bloom, tonemapper and eye adaptation.

Anti-aliasing is the method used for repairing aliasing problems, which is when a 3D polygon is rasterized into a 2D screen with limited resolution, creating visible pixelated edges. Anti-aliasing reduces the jagged lines created around the objects by surrounding them with intermediate shades of colour [37]. The effects of anti-aliasing can be seen in Figure 5.8.

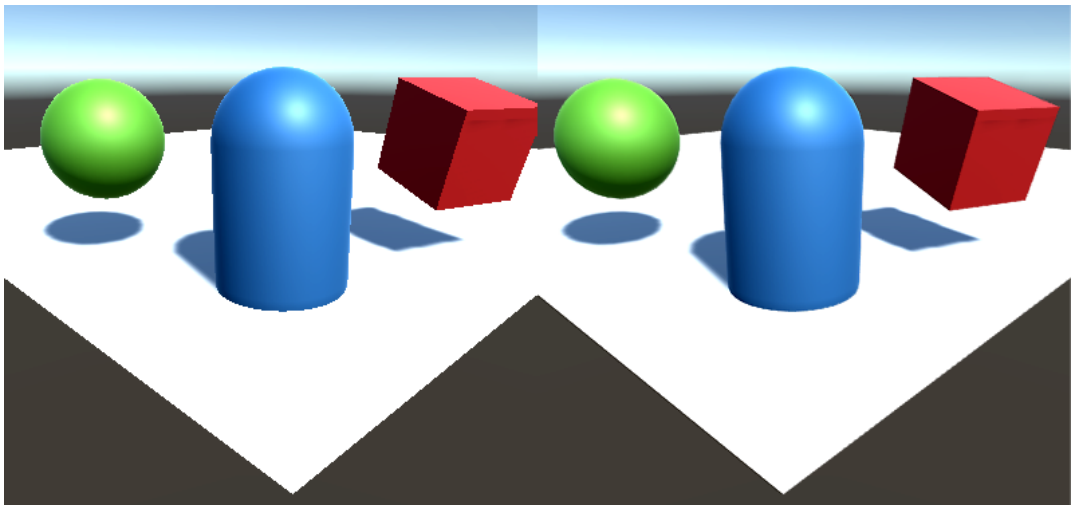

Figure 5.8: A virtual scene rendered without anti-aliasing (left) and with anti-aliasing (right) [37]. The anti-aliasing blends the pixelated edges, making them appear smoother.

Two of the most popular anti-aliasing techniques, that are currently provided in Unity's post process stack, are the Fast Approximate Anti-Aliasing (FXAA) and the Temporal Anti-Aliasing (TAA) [36].

FXAA is the cheapest technique and is a pure post process anti-aliasing [36]. The edges of the rasterized image are analysed, and an algorithm smooths the jagged lines [36]. The amount FXAA smooths can be adjusted, thus establishing control over the trade-off of quality and performance.

TAA provides much smoother edges, but is more complex than FXAA, since it uses jittering and the previous frame as additional data for blending the pixels [36]. Furthermore, motion vectors are used to predict which pixels are needed for the

final render [36]. Unfortunately, TAA is currently unsupported in VR applications, therefore, FXAA is used for this project.

Another post processing effect is the *ambient occlusion*, which is the approximation of ambient occlusion based on screen space data, thus, it is often referred to as Screen Space Ambient Occlusion (SSAO). It darkens holes, creases and other areas where surfaces are close to each other. SSAO is not directly dependant on scene complexity, but rather on screen resolution and the set parameters such as, radius size and sample count. Ambient occlusion is used in the creation of the VE, in order to enhance the realism and give more depth.

Bloom is an effect that aims at simulating the imaging artefact created by real-world camera lenses. Bloom creates fringes of light emanating from the borders of a light emitting object and can contribute to the illusion of a very brightly light source, therefore, we added the bloom effect to our scene.

As already mentioned in Section 5.1.2, using an HDR camera requires the use of a *tonemapper*. Unity post process effects offer two types of tonemappers: Neutral and Academy Colour Encoding System (ACES). The neutral tonemapper does range-tonemapping but it has a minimal impact on colour hue and saturation. However, the neutral method offers full parameter control over the tonemapper curve and, therefore, allows for customizability. The ACES, otherwise known as the filmic tonemapper, is set to automatically create a more filmic look, as it strongly impacts the colour hue and saturation, which is the reason it provides stronger contrast of colours than the neutral. ACES is considered the simplest and requires no user input to setup in order to get a filmic look in the scene, which is also why we used it in this project.

The *eye adaptation* post processing effect refers to the ability of the human eye to adapt to different levels of brightness. It adjusts the exposure of an image, over time, according to the brightness levels the image contains. We chose to use this effect in order to increase realism by simulating the effect of eye adjustment to light, which happens when the user is moving the gaze from a dark environment towards a bright one, and vice versa.

5.1.5 Unity Assets

The following section describes Unity assets used for the creation of the virtual scene.

Unity Standard Assets

In Unity a sea asset was added to the scene, which was modified to have waves and foam, by changing the wave amplitude and foam intensity, in order to add to the realism of the scene. The specular light of the sea surface was set to depend on the directional light that was used as the sun, so that the specular reflection was dependant on the position of the main source of light of the scene.

The Enviro - Sky and Weather Asset

In order to create the desired atmosphere for our architectural 3D model we investigated the existing solutions for weather system control. We found several solutions but the best in terms of performance, believability, realism and compatibility with VR systems, is the Enviro Sky and Weather asset by Haupt [11]. We added this comprehensive dynamic weather system to our scene because it delivered a complex system with numerous features, such as day and night sun and moon simulation, rain and snow particles, terrain shaders, light simulation from the sun and moon, weather and sound presets, high quality visuals, as well as volumetric light and clouds. Besides the overall good performance and quality, the asset has good and numerous reviews, is widely recommended amongst other users and is maintained, with the author constantly updating the asset.

In order to actually achieve our goal of creating a believable and realistic environment, we had to explore the functions of the asset, understand them and change some of the default values to fit our goals for the environment and its atmosphere. All those values are retained in profiles, which Enviro is using as different presets that create different atmospheres and can be seen in Figure 5.9. The values in the profile responsible for the overall look are grouped in different categories of setting.

We created our own preset, for which we have changed values for lighting, sky, clouds, volumetric lighting, light shafts, and audio default settings. The settings were implemented for the desktop version of the scene and, therefore, are not final. The final settings are shown in the second cycle.

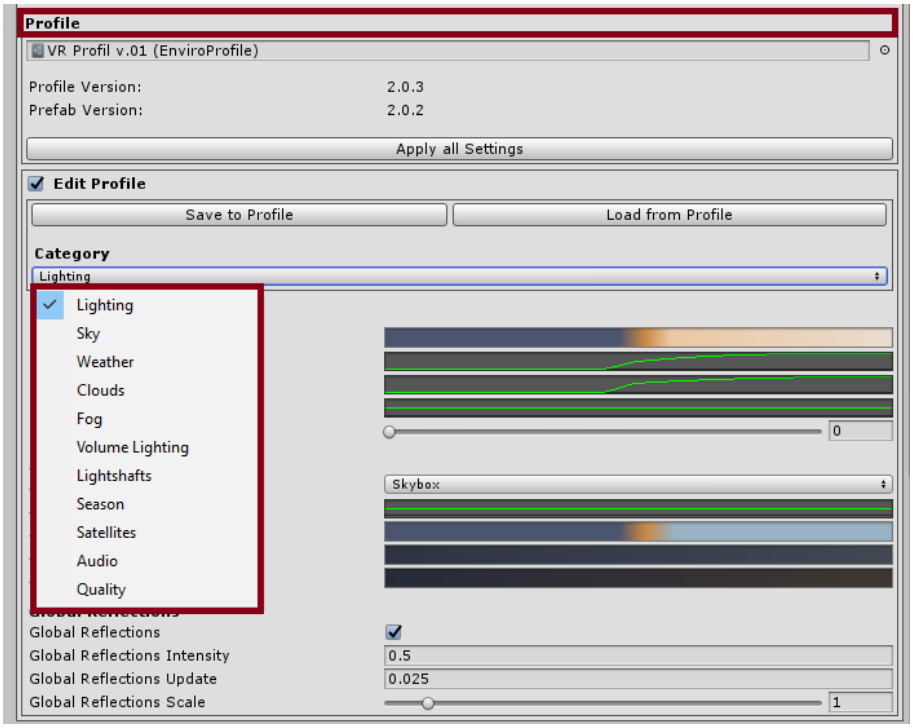

Figure 5.9: The Enviro categories panel with the profiles, which in this image is set to Lighting.

5.1.6 Collaborator Evaluation

The first cycle ended with the creation of a desktop version of the scene. A first person avatar that could be controlled with the keyboard was added in the scene, to allow for exploration. However, the scene did not have a UI system, therefore, controlling the day and night cycle and changing weather effects was not possible.

The scene was sent to the collaborating architect for feedback. When showing the prototype to the architect, we received feedback on further improvements, such as adding more illumination in the entry hallway, as this part is in shade and is quite dark, and improving the reflections on the windows.

The intensity of the reflection probe for the windows was increased to show the reflections clearer and make them appear more believable. Furthermore, the Enviro settings are finalized in the next cycle.

5.2 Second Cycle

The second cycle mainly focuses on making the scene into a VR scene that uses the HTC Vive. This section explains issues noticed after the first cycle, the assets we used for enhancing the scene and the UI created for interacting with the system. Furthermore, the section contains the second cycle of collaborator evaluation.

An issue we noticed was culled surfaces in the scene. The wooden cabin in the scene was constructed of several patches of wooden tiles, used both for the floor and walls. However, some of the tiles inside the cabin had normals facing outwards, meaning that from the inside of the cabin, the user could see the back faces of the wooden tiles, which usually are not rendered and, thus, created missing patches in the walls. The issue was solved by manually flipping the normals of each of the unrendered wooden patches inside 3ds Max and then re-importing the model into Unity.

5.2.1 Unity Assets

The following section describes the assets used for the improving the virtual scene from the first cycle.

Unity Standard Assets

The collaborator wants fire in the fireplace at night-time, so we added a fire to the fireplace, using the flames particle effect from the Unity Particle Pack asset by Unity Technologies [44]. The embers were disabled, as they were unnecessary and went through the lid of the fireplace, and we use a spotlight prefab as the light source in order to have the light from the fire cast shadows. The angle and intensity of the spotlight were changed to provide a softer light spreading from the opening of the fireplace.

The Enviro - Sky and Weather Asset

Based on the given feedback and the fact that the scene will be converted to use the HTC Vive, changes were made to the Enviro asset settings, with the following settings for each of the categories.

The *Enviro lighting* category is responsible for the light, the intensity of light emitted by the sun and moon present in the scene, the ambient lighting and global

reflection. Through it, we adjusted the colour of the sun and moon light to a light-orange to dark-blue gradient and the result can be seen in Figure 5.10, while the settings can be seen in Figure 5.11.

Figure 5.10: The environment when the direct and ambient lighting settings are applied.

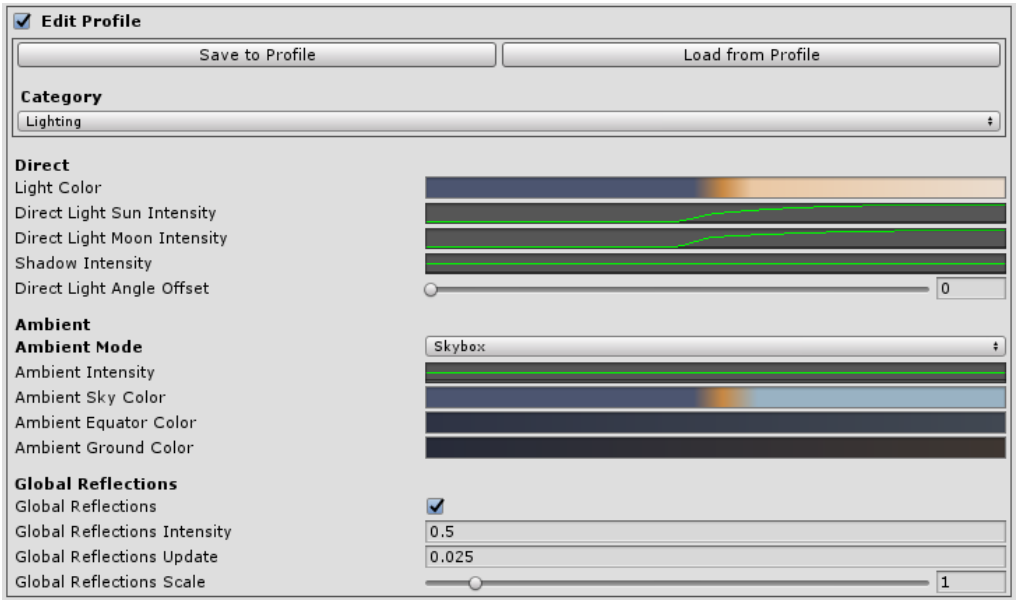

Figure 5.11: The final settings used for the lighting category.

The ambient mode is set to the skybox, which means that the ambient light is coming from the colour of the selected skybox.

The Enviro sun and moon light are built as a real-time directional light, which was scripted to light up the scene according to the position of the sun and moon

respectively. In terms of the day and night cycle, the Enviro asset handles the sun, moon and stars position by computing their positions based on inputs, such as time of day, time of year, longitude and latitude. Furthermore, the Enviro asset allows for automatic changes of the sun intensity based on the time of day.

The *Enviro sky* category is responsible for the visual aspect of the sky. Enviro implements atmospheric scattering and through the sky category, we can adjust it along with the sun and moon rendering options, and sky illuminance.

We decreased the rayleigh to give the sky a deeper blue colour. The atmospheric scattering curve and colour sculpt the sky by blending colours according to default curves. In Figure 5.12 the effect on the change of the atmospheric scattering curve is shown.

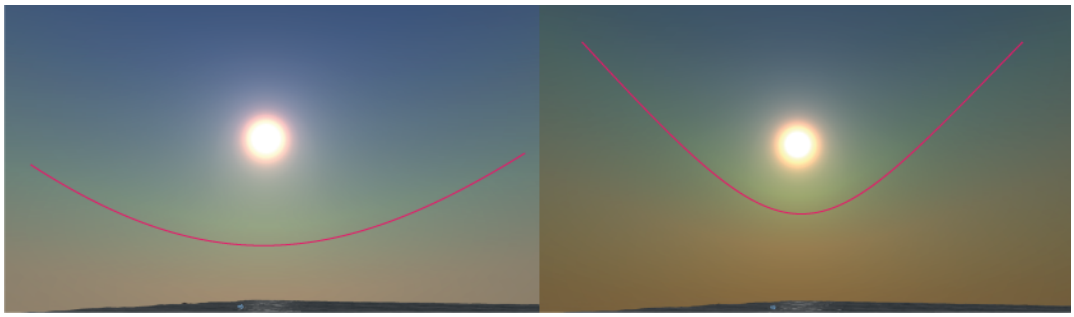

Figure 5.12: The change of the atmospheric scattering curve from a lower value (left) to a higher one (right).

The end result of our changes for this category can be seen in Figure 5.13, with the final settings in Figure 5.14.

Figure 5.13: The end result of our settings for the sun and sky.

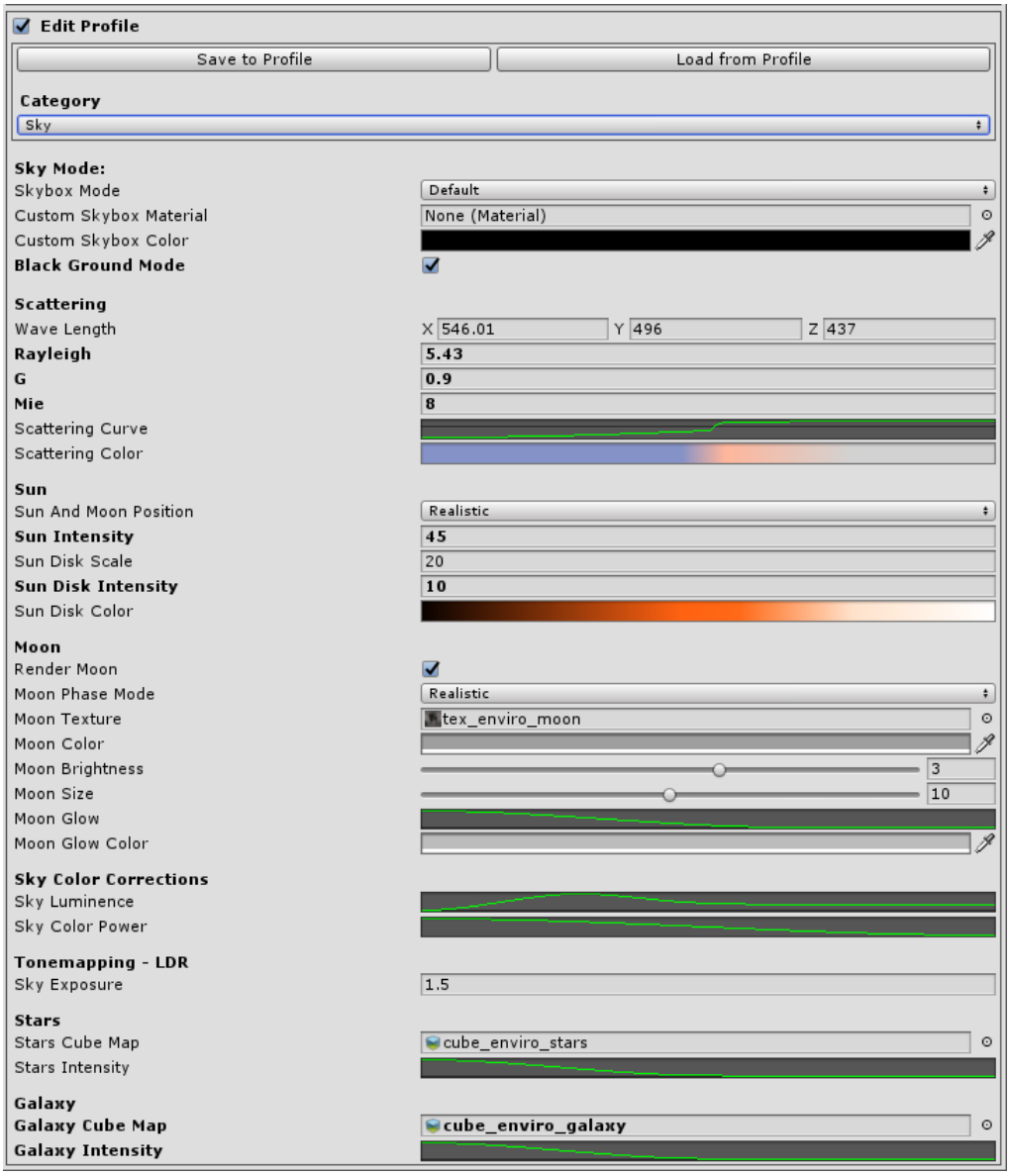

Figure 5.14: The final settings used for the sky category.

We increased the sun intensity, and reduced the sun disk intensity to obtain a more realistic shape for the sun, with a smaller disk and stronger lighting. The sun disk colour curve has been adjusted to show a deeper red colour at sunset. The sun intensity feature changes the intensity of the sun disk, giving it a bigger or reduced size scale depending on the given value. The sun intensity feature had an optimal value for our scene by default, therefore, we have not changed it.

The sky luminance and colour power features refer to the visual aspect of the sky and determines how the light scatters and diffuses and respectively it controls the contrast of the sky blending colours.

The *Enviro weather* category is responsible for transition settings between weather presets and their effects. Those values were considered adequate for our scene, therefore, they were not changed.

The *Enviro clouds* category is responsible for defining the cloud quality, lighting, shadows, and how they are affected by the speed of the wind.

In Figure 5.15 the end result is shown, and the final settings can be seen in Figure 5.16.

Figure 5.15: The results of the settings we applied for clouds and shadows.

In Enviro, the clouds are rendered in two different ways, flat clouds and volumetric clouds. The flat clouds are part of the sky box, for which the material is changing depending if the flat clouds are being rendered or not. The volumetric clouds were modeled using ray marching framework starting from the camera and sampling noises together with a collection of gradients to illustrate the cloud shapes using a standard sampler. The clouds were built with two different levels of detail, a low frequency cloud base shape and high frequency detail and distortion shape. The noises are made using Perlin and Worley and Curl noises, while in order to control the density of the clouds and their coverage, a collection of presets was used. The clouds are animated towards the wind direction.

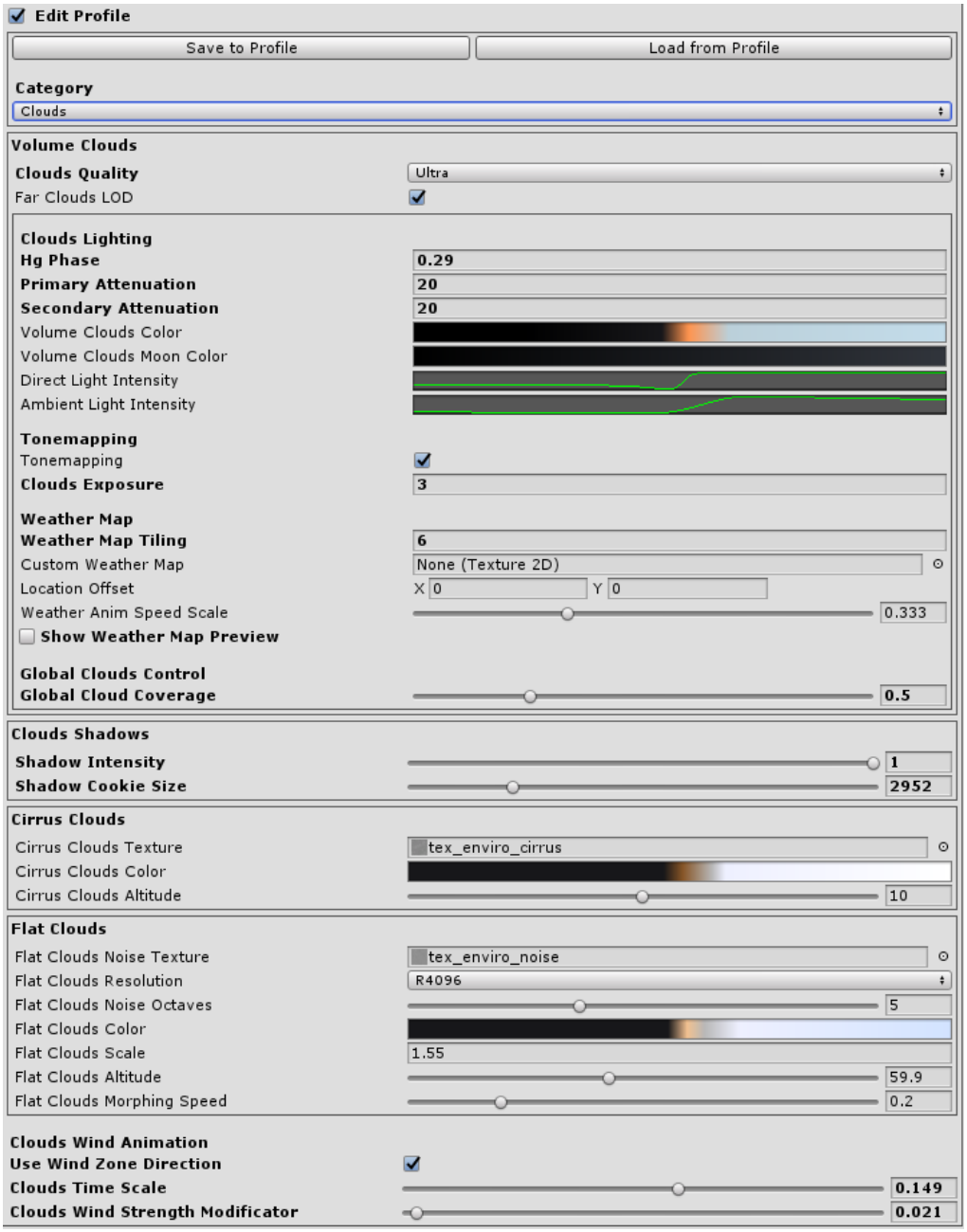

Figure 5.16: The final settings used for the cloud category.

The *Enviro volumetric lighting* category is responsible for adjusting the volumetric light rendering. It uses a light space shadow map and a depth buffer, and was optimized using ray marching and blur to work in real-time. The result of our modifications can be seen in Figure 5.17, with the final settings in Figure 5.18.

Figure 5.17: The outcome of the settings we applied for the volumetric lighting.

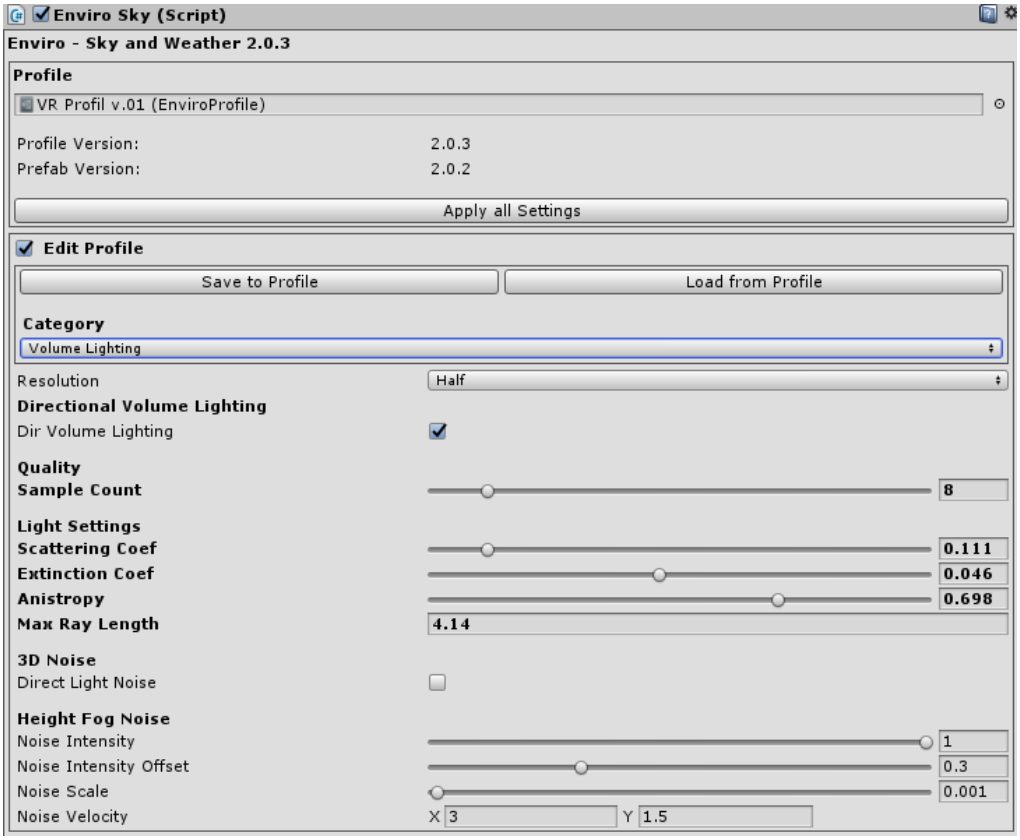

Figure 5.18: The final settings in the volume lighting category.

The Enviro light shafts feature influences the light beams emitted from the sun or other light sources that reach the users eyes as result of light scattering off of particles that settle in the air. We changed the colour grade of the light shafts emitted from the sun. The end results can be seen in Figure 5.19, and the final settings can be seen in Figure 5.20.

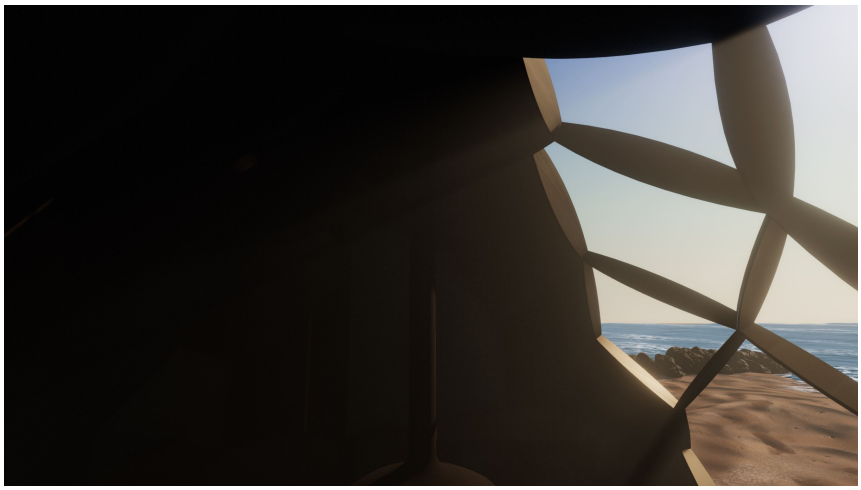

Figure 5.19: The light shafts from the sun as a result of our settings.

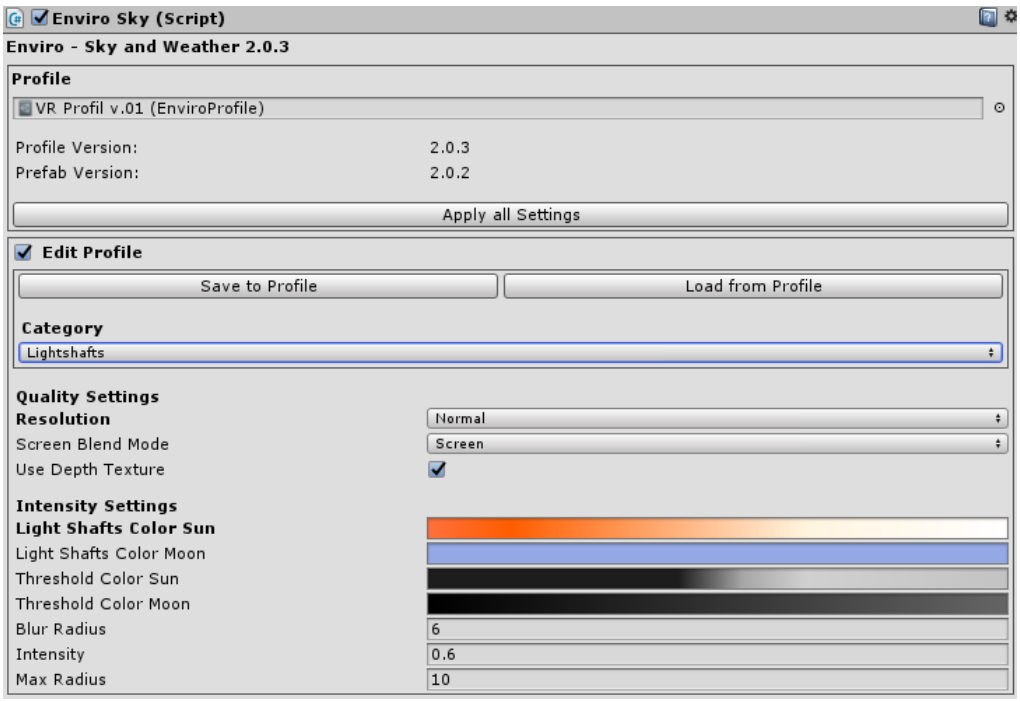

Figure 5.20: The final settings used for the light shafts category.

The *Enviro audio category* holds the audio files that are being played for different weather presets and effects. We changed the audio profiles for all the weather presets to match the audio with our own VE using audio by klankbeeld [15].

5.2.2 User Interface

UIs are complex elements integrated at the core of the application that allow communication between the user and the simulation. It consists of elements such as canvases, panels, images, buttons, sliders and dropdown menus. In Unity, the canvas is necessary and represents the support of all the other UI elements present in a UI. A canvas can be rendered in different modes, Screen Space Overlay, Screen Space Camera and World Space. The first render mode allows the UI to overlay the scene, therefore, all the existing UI elements will be rendered on top of everything else in the scene. Screen Space Camera resembles the first render mode, but instead of filling the screen space with the UI, it will render it to a specific camera present in the scene. The World Space render mode allows for rendering the canvas on static or dynamic objects inside the scene space anywhere inside the world space.

The UI was developed to support easy and accurate interaction and consists of one World Space canvas, a panel containing text boxes to specify the meaning of the interaction, buttons, sliders and dropdown menus, as seen in Figure 5.21. We aimed to create a simple and concise UI.

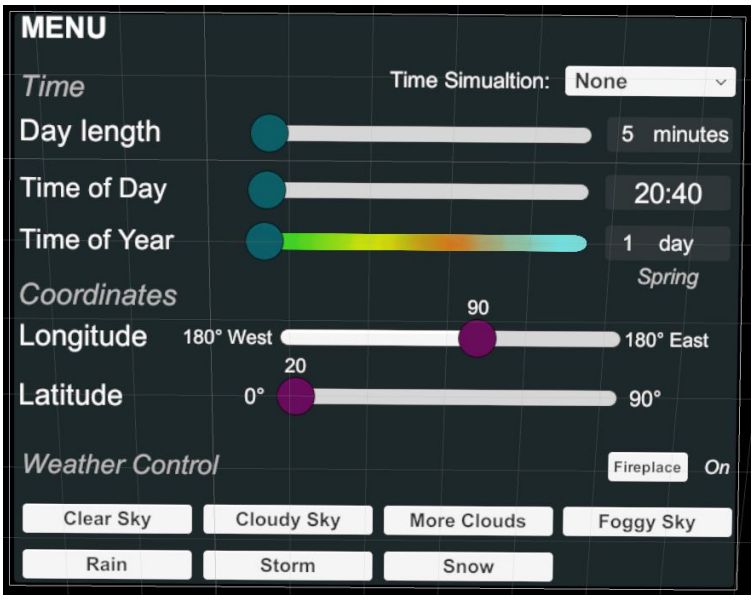

Figure 5.21: The initial UI setup with text descriptions of the UI elements.

The UI contains the following features in a top-down order:

- A dropdown menu for time simulation which offers three options:
  - None, which makes the scene static.
  - Simulated, which makes the scene dynamic by automatically running a day and night cycle and gives the day a pre-set length.
  - And System Time, which makes the scene dynamic and mirrors the time in the scene with the one in the system the scene is being run on.
- A slider for day length which permits the user to set the virtual day length between zero to 60 minutes.
- A slider that allows the user to control the time of day inside the scene.
- A slider that changes the day of the year through 365 days.
- Two sliders to allow the user to change the location around the globe, one that allows the change of longitude from 180 degrees West to 180 degrees East and another that allows for the change of latitude from 0 degrees Equator to 90 degrees North or South Pole.
- A button for the fireplace light control.
- And seven buttons to control the weather inside the scene.

In addition, the UI contains a number of text elements describing the use of each interactable UI element. Where six of the text components are used by a script to provide the current value of the corresponding UI element in real-time. We investigated the Enviro asset in order to find out different functions form different scripts that we had to use to create the desired atmosphere. After we determined the scripts and necessary functions, we created our own script that calls the corresponding function from the scripts in the Enviro asset, when activating or changing the value of a UI element.

User Interface Interaction

In order to create user interaction with the UI we added the two assets, SteamVR Plugin and VRTK Plugin, to our project. SteamVR connects the HTC Vive headset and controllers to the Unity project and allows the user to interact with the VE in real-time. The VRTK plugin was used for an optimal and easy interaction between the user and the UI, as our attempts to only use the SteamVR scripts for the interaction between the HTC Vive controllers and the UI failed, whereas VRTK permits the SteamVR controllers to interact with the canvas and the elements in the canvas.

A number of VRTK scripts for pointer render and events were added to the right controller, allowing the user to click on buttons, drop the menu and drag sliders.

At first, the pointer was rendered only when the touch pad of the right controller was pressed, and then the click would happen by pressing and releasing the trigger button. However, we decided to limit the point renderer to the canvas that is holding the UI elements, and have it appear instantly when the controller is hovering over the canvas, in order to simplify the interaction and visually limit the pointer to the UI, as it is the only element in the VE the user can directly interact with. Limiting the point renderer was done by creating a script that communicates with the VRTK scripts and turns the pointer invisible when it is not hovering over the canvas.

User Interface Design and Placement

The design of the UI is simple and classic to avoid distracting the user too much from the scene but also to that the UI feels like a tool rather than as part of the environment.

The UI was placed on the left controller, on the left side and perpendicular to it, so that it can be used in the same manner as a wrist watch. A script was added to allow the UI to appear and disappear based on the rotation of the controller, making it visible only when the user would intend to look at it by turning the controller.

5.2.3 Collaborator Evaluation

Once everything was implemented, this version was sent to the collaborator for feedback. The collaborator made several suggestions, including several small changes to assets. Further suggestions for changes were: small adjustment to the scale of the wood material, templates for world coordinates for major places so they can be chosen via a drop down menu, template for season changes with a drop down menu, a dynamic sea influenced by the current weather, repairing the leaks, where water drops and snow passing through the ceiling, an avatar for the user that could help with scaling the VE. Amongst the proposed changes was the sea asset itself, since it was not responding to the light in the scene, the reflection of light on the surface was either too bright or too dark, depending on the time of day.

Another suggestion was a change in the toggling of the UI. Instead of turning the wrist, the UI would toggle at the press of the left touchpad, with the UI appearing on top of the controller. An icon could also be added to the touchpad to symbolise the purpose of it.

After the feedback, the tweaks for the scale of the wood were implemented. The templates for coordinates and seasons, as well as the dynamic sea were considered interesting ideas and were left for a future cycle. The avatar idea was not implemented, although it can help the user with the scaling of the world, as the collaborator's initial criteria was that the user should feel as if he is there only as an observer with a tool to change the environment and should not get too immersed. The sea asset was decided to be replaced in a future cycle.

The new UI toggle option with a button instead of the wrist was taken into consideration and we decided to further investigate it with a user evaluation.

Chapter 6

User Evaluation

As mentioned in Chapter 5, the evaluations by our collaborating architect Jacob Hilmer served as the main evaluation of the system. However, during the Participatory Design process, we came across two UI toggle options, where the collaborator proposed pressing a button instead of turning the wrist. We decided to conduct a user evaluation based on the idea for changing the UI toggle method, as we thought it would be interesting to determine which type had a higher usability.

As previously mentioned, the first type of UI used text boxes to describe the different options the UI offered. However, text in VR can be problematic for some users, depending on the VR headset and the user's eye-sight. Therefore, we built a second type, UI with icons, where we created icons for the UI buttons to visualise the weather presets. The icons can be seen in Figure 6.1. In addition, the time of year slider has images visualising the seasons.

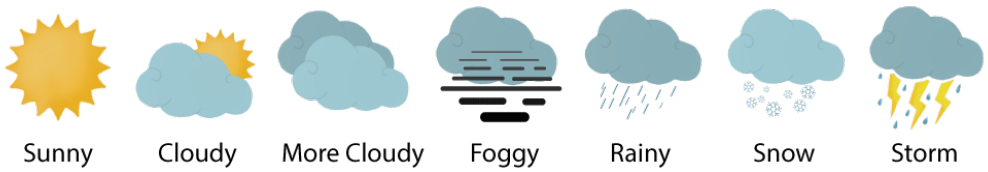

Figure 6.1: The icons that we created for the weather presets.

The two UI types can be seen in Figure 6.2. For the button elements we added a script to change the colour of the buttons in order to show the user which of the weather conditions is on at the current time.

Both types of UI were made to be placed on the left controller, in two possible setups for toggling the UI on and off. For the first toggle type, the UI is placed on the left side and perpendicular to the left controller, using it as a wrist watch,

while for the second toggle type, the UI is placed on top of the controller. For the second toggle type, the UI is activated and deactivated by pressing the touchpad on the left controller using a script that communicates with VRTK plugin.

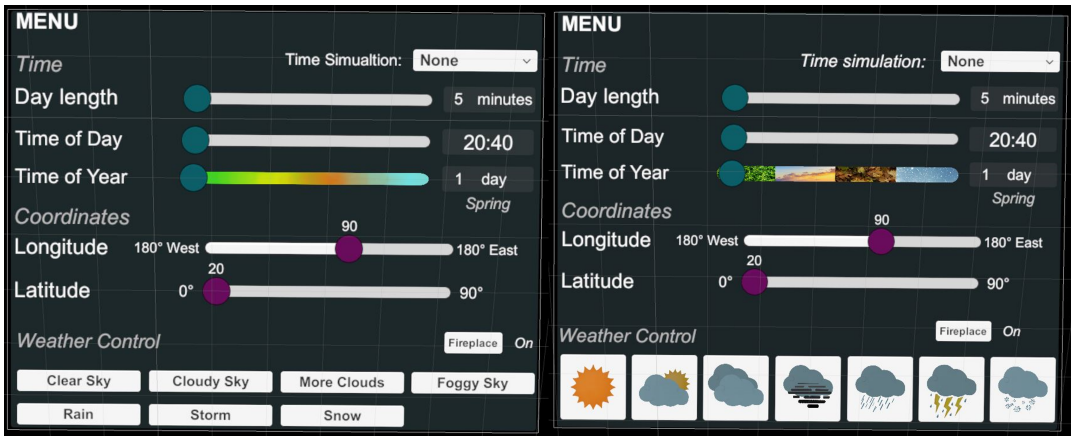

Figure 6.2: The left shows the first UI with text, while the right shows the UI with icons as visual description of UI elements for weather and time of year.

We also added a weather icon to the left controller to make it more intuitive for the user to open the UI. Figure 6.3 illustrate the two weather icons that were attached to the touchpad button, one for each UI type.

Figure 6.3: The controller touchpad icons made for the UI version with icons (left) and for UI version with text (right).

The two types of UI for controlling the day and night cycle, weather conditions and fireplace are evaluated based on their usability and user preference. The evaluation compares the usability of using text against icons for the UI, as well as the two ways of toggling the UI on and off.

To determine their individual usability, counterbalancing for effects of having text or icons with a certain type of UI toggle, the versions for testing are as follows:

1. UI text with wrist toggle
2. UI icons with button toggle
3. UI text with button toggle
4. UI icons with wrist toggle

Participants are randomly assigned to test versions one and two or versions three and four, in a randomized order, so that all participants test both text and icons as well as both UI toggle methods.

The evaluation is carried out using the HTC Vive.

6.1 Procedure

First, the participants sign the informed consent form and fill out the questionnaire for demographics and then enter the VE with one of the testing versions of UI and toggle type.

Inside the VE, the participants are introduced to the system and how to control the UI. They are then given tasks to change the time of year, time of day and weather as well as to light the fireplace, in order to prompt them to interact with the UI and see if they understand the different functionalities of it. After finishing the tasks for the first version, the participants fill out the usability questionnaire.

Then the participants carry out the same types of tasks with the second version of UI and toggle type and fill out the usability questionnaire for this version.

Lastly, a semi-structured interview is conducted to determine the users' preferences for the UI and toggle types and obtain feedback on their preferences and improvements to the UI and interaction.

6.2 Results

Out of the 16 people who participated in the evaluation, 11 were male and five female, with ages ranging from 21 to 27 years old, with a mean age of 24. Out of the 16 participants one had no experience with VR, seven had tried using it a few times before, four use it roughly once a month, two use it once a week, and the last two use VR multiple times a week. This means that the participants had varying levels of experience with VR.

6.2.1 Results on Usability

We determine the usability of the UI and toggle versions using the System Usability Scale (SUS) by Brooke [4]. The 10 statements in the SUS are rated on five-point Likert scales ranging from strongly disagree to strongly agree, where the SUS score from these statements ranges from 0 to 100 [4].

According to Sauro [29] and Brooke [4], the SUS method is both reliable and valid, even with small sample sizes. For our evaluation the 10 statements are adapted for both the UI toggle method and then for the UI itself, in a total of 20 statements seen in Appendix A.

To get the SUS score, first the scores from the individual statements are calculated. Each statement has a positively and a negatively worded version, and for each positively worded statement, one point is subtracted from the five-point Likert scale user response, and for the negatively worded statements, the user responses are subtracted from five. This procedure scales the values from zero to four, where four is the most positive response. These values are summed, and the total is multiplied by 2.5 to convert the range of possible values to between zero and 100. The average SUS score for all systems in the database is 68, meaning that any score above 68 is considered above average. However, according to Sauro [29], the percentile rank and letter-grades provide a better interpretation.

The SUS scores for the four version and their different UI and toggles types can be seen in Table 6.1, and the individual SUS scores for the toggle and UI types across the four versions are visualized in Figure 6.4.

	Button toggle	Wrist toggle	UI text	UI icons	Average SUS
Version 1		77.5	73.1		75.3
Version 2	88.4			87.5	88
Version 3	82.2		77.2		80
Version 4		81.6		78.4	80
Average SUS	85.3	79.5	75.2	83	

Table 6.1: The individual SUS scores for the four versions and the two toggle and two UI types, along with the average SUS scores for each version and type.

The individual SUS scores for the toggle types, show that the button has a higher usability than the wrist toggle, regardless of the UI type. Similarly, the UI icons have higher usability scores than the text, regardless of the toggle types. However, the scores all vary, depending on the combination, suggesting that the combination of UI and toggle types influence each other.

Interestingly, the third and fourth versions have the same average SUS score, despite using different UI and toggle types. Here, the SUS scores for the button and wrist toggle types are only 0.6 points apart, and the UI text and icons are 1.2 points apart, and so they are quite similar. This suggests that combining button toggle and text for the UI provides the same approximate usability as combining the wrist toggle and icons. However, when combining the wrist toggle and UI text, the usability is at its lowest, although this is still above average usability.

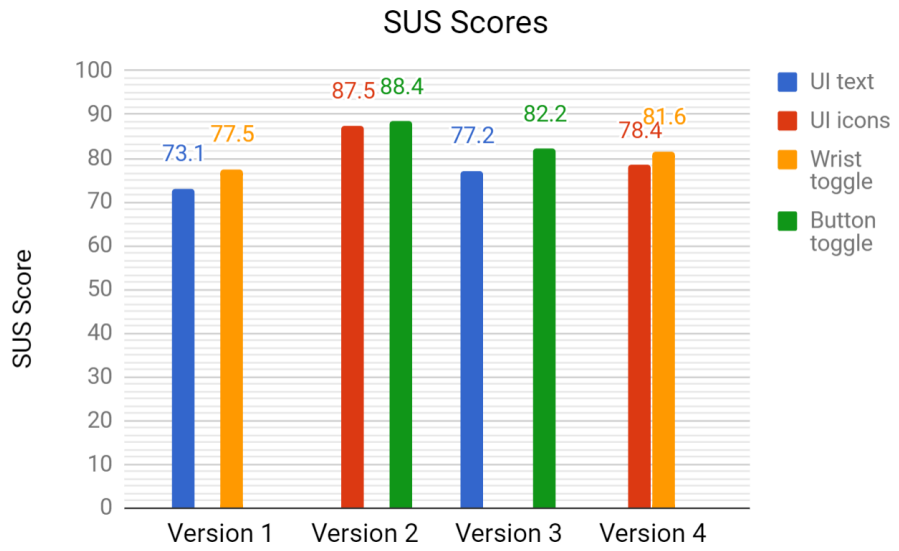

Figure 6.4: The SUS scores for the UI text and icons as well as the wrist and button toggles across the four versions, which combined the toggle and UI types differently.

Oppositely, combining the button toggle and UI icons, the usability is higher than for the other combinations, which correlates with the button toggle and UI icons scoring higher usability scores than the wrist toggle and UI text types.

When looking at Table 6.2 with the percentile rank and curved grading for the four versions, it is clear that combining the button toggle with the icons for the UI provide a higher usability, showing that the second version has a higher usability than 98.7% of the systems in the database for SUSs, rating it at an A+ grade for its usability.

	Version 1	Version 2	Version 3	Version 4
Average SUS	75.3	88	80	80
Percentile rank	73	98.7	88	88
Curved grading	B	A+	A-	A-

Table 6.2: The results of the four versions in terms of their average SUS scores, percentile rank, as the percentage of systems in the SUS database below the usability level for each version, and the corresponding curved grading, providing a grade for the system usability.

In addition, Table 6.2 and Figure 6.5 show that version one has the lowest usability of the four, versions three and four have the same usability, while the second version has the highest usability of the four, with the highest possible grade.

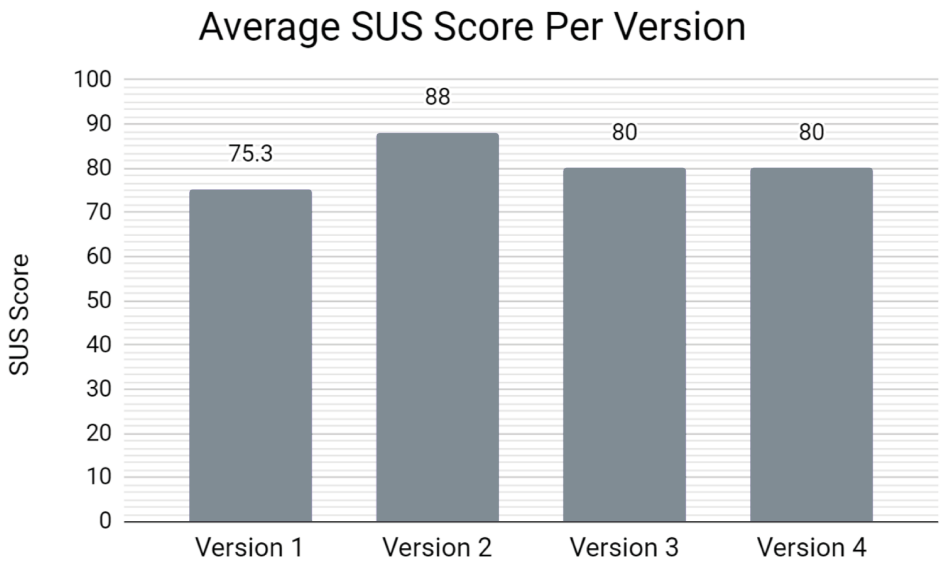

Figure 6.5: The average SUS score for each version, with the possible SUS scores ranging from zero to 100.

Figure 6.5 visualises the average SUS score for each version, clearly showing that version two has the higher usability at 88 out of 100.

Looking at the results for the toggle and UI types in Table 6.3 and the visualization of the average SUS scores in Figure 6.6, the button toggle is more usable than 97% of the systems in the database, getting the highest possible grade of A+. The wrist toggle is at a lower usability, which is still at an A-, well above average. The UI text has the lowest of the four SUS scores, getting the grade of B, still above average and usable, however, the UI icons get an A for usability, being more usable than 90% of the systems in the SUS database.

	Button toggle	Wrist toggle	UI text	UI icons
Average SUS	85.3	79.5	75.2	83
Percentile rank	97	87	73	90
Curved grading	A+	A-	B	A

Table 6.3: The results of the two toggle types and two UI types, in terms of their average SUS scores, the percentile rank, which is the percentage of systems in the SUS database scoring below the percentile rank, and the corresponding curved grading, showing the grade for the system usability.

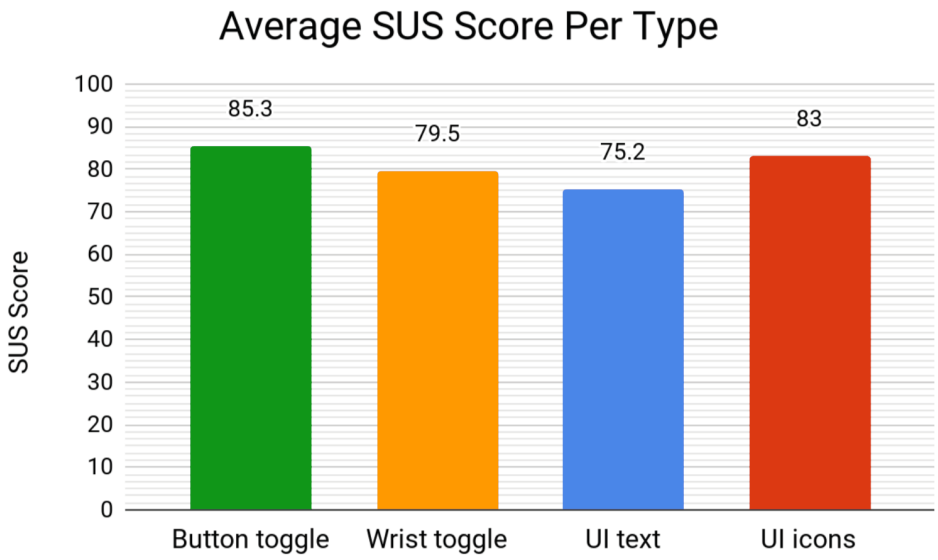

Figure 6.6: The average SUS score for each toggle and UI type, in the range of possible SUS scores from zero to 100.

In summary, the usability results show that the button toggle is more usable than the wrist toggle, and that the UI with icons have a higher usability than the UI with text. This correlates with the results showing that version one, combining the wrist toggle and UI with text, provides the lowest of the four usability scores, while version two, combining the button toggle with the icons for the UI, has the highest usability score of the four versions.

6.2.2 Results on User Preference

Figure 6.7 shows an overview of the participants' preferences.

For the toggle types, nine of the 16 participants preferred using the button, while four preferred using the wrist. The remaining three did not have a specific preference for toggling, as they found trade-offs to both types. Of these three, one found them both usable, noting only that the wrist toggle felt a bit more awkward to use than the button, the two others stated that they found the wrist toggle cooler but more likely to cause fatigue, where one of them went on to elaborate that the wrist seemed more natural, but that the button felt better for long term use. The wrist toggle causing fatigue was also the reason for six of the participants to prefer the button, while two others found the wrist toggle annoying, and therefore, preferred the button toggle as well. Another participant though the hand position for the button toggle felt more natural, preferring this over the wrist toggle.

On the other hand, four participants based their preferences on the wrist toggle feeling more natural, with one adding that this was especially true for those used to smartwatches.

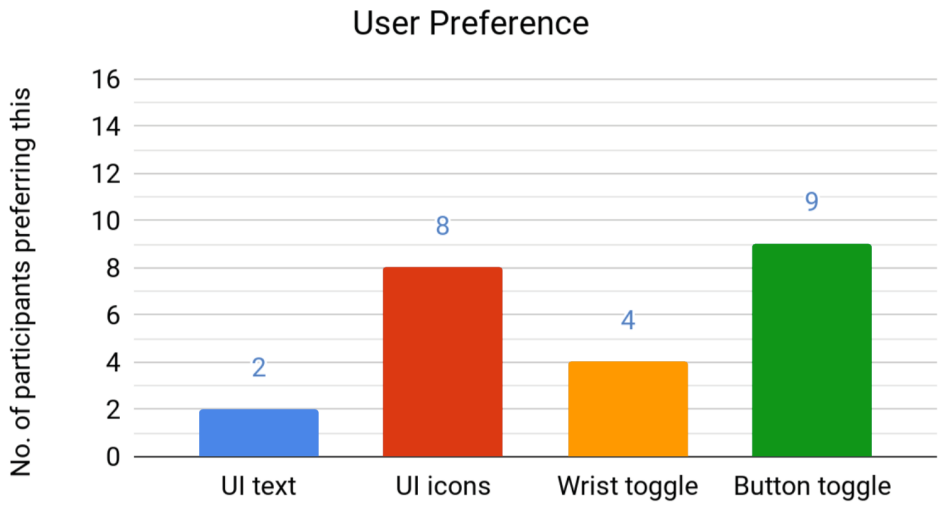

Figure 6.7: The number of participants out of 16 who preferred the button toggle, wrist toggle, UI text and UI icons respectively. Three participants did not have a preference when it came to the toggle types, as they found trade-offs for both types. Additionally, six participants did not prefer one UI type over the other, instead four wanted to combine different parts of the two UIs.

The aforementioned suggests that for six of the participants, the toggle type preference would depend on whether they had to use it for long or short term. However, overall, the wrist toggle was thought to be very interesting, while the button was easier to use, something that the usability results support.

Only two of the 16 participants preferred the UI with text over the one with icons, stating that the text was easier to understand than the icons. Contrarily, one participant preferred the UI with icons, as they were more intuitive and easier to use both for the time of year slider and the weather changes.

One participant preferred the icons as they provided better visuals and meant that she didn't need to read. Similarly, two other participants also preferred the icons, as they were easier to use, and they had a hard time reading the text in VR.

Two participants preferred the icons, especially because of the time of year slider, as this type provided a clear split between the seasons. One of them added that the weather icons were better than just having text, although she would have liked to combine them and have both the icon and text for clarity.

Not all participants had a specific preference in terms of the UI. One did not have a preference, as he understood both UI types. Another preferred the time of year slider from the icons UI type, as it made the seasons more distinctive, however, he did not prefer one type over the other as a whole.

Four other participants also did not prefer one type over the other, and instead wanted to combine different parts of the two types, in order to have the text for the time of year slider and the icons for the weather buttons. Two of them said that the text made it clearer to them which season it was, instead of, e.g., having to assume for the icons that the green colour meant spring. The other two said the slider for the UI with text made it seem like they did not have to be as accurate. In a similar manner, two of the participants also liked how the weather buttons were bigger in the icons type, and all four participants agreed that the visual representations of the weather conditions from the icon type were clearer than the text type. However, two of them noted that the meaning of the fog button was unclear. Equivalently, another participant also preferred how the icons made the weather buttons especially clear, as they visually showed the weather, whereas, e.g., storm could mean many things to him.

Oppositely, another participant preferred the text for the weather buttons, as he did not notice the difference between cloudy and overcast in the icons, whereas the text made that clearer. In addition, he preferred the time of year representation from the UI with icons, as it had a clearer distinction between seasons.

In total, two participants preferred the UI type with text, eight preferred the UI type with icons, two had no preference, and the remaining four would have preferred having the text type for the time of year slider with the icons for the weather buttons.

Note that, of the seven participants who normally wear glasses but could not wear them in VR, four of them preferred the UI with icons, none preferred the UI with text, one had no preference and the remaining two preferred the text type for the time of year slider and the icons for the weather buttons.

As half of the 16 participants preferred the UI with icons, while the other half's opinions differed, it seems UI with icons is the preferred type. However, as a fourth of the participants suggested combining different parts of the two UI types, it would be interesting to evaluate this combined type and compare it to the tentatively preferred UI with icons type, in order to determine if the combined type would be preferred.

Chapter 7

Discussion

During the design cycles, several issues were encountered. The following chapter expands on these issues and ends with a discussion of the user evaluation.

One of the main issues we encountered was with the 3D model itself. The wooden patches that construct the cabin were causing several problems, including leaking of volumetric light scattering as well as raindrops and snow. These issues are suspected to be caused by the patches themselves being modelled to be too thin, beyond what the Unity game engine can compute correctly, which would explain why the leaks are happening. The raindrops leaking through might also be caused by the Bullet Through Paper problem, where an object passes so fast near a thin surface that the game engine does not record the two meshes hitting each other, thus the objects never collide, and the raindrops pass through unobstructed. Currently, the cabin has two layers of wooden patches, an outer and inner layer, which prevents the light scattering leaks and decreases the amount of leaking raindrops.

When modifying the scene to be used with an HTC Vive, the Enviro asset caused a problem where the sky could not be seen properly through the HMD. The scene rendered the sky separately for each eye on the HMD, thus, when toggling weather effects such as clouds, the clouds were different for each eye. The issue was fixed after the asset creator Haupt [11] made a hotfix.

The unity standard sea asset caused issues with reflections. The sea asset seemed be unable to adapt to a dynamically illuminated environment, thus, for a future cycle it has to be replaced.

Amongst the versions of UI we implemented for the user evaluation, we considered another type of UI display, where instead of the UI appearing on the controller it would appear somewhere in the VE. The idea was scratched since then the canvas would become part of the world and this might impact the believability of the

scene. The canvas needs to feel more like a tool for the user than a part of the world, as this is also something the collaborating architects wants.

After the second cycle, the collaborating architect provided ideas for possible future features such as, a snapshot button that can take pictures and small videos from inside the VE, an addition to the UI that allows the user to call out a plan of the building and section drawing with measurements and dynamic section cuts. Furthermore, another idea is the implementation of teleportation inside VR, to allow the users to explore an external architectural work or explore both the inside and outside of one.

7.1 Discussion of the User Evaluation

For the third version with the button toggle and UI icons, the button toggle feature had a bug, where the menu text on the button itself sometimes did not appear, which might have caused a bias compared to the second version, where the sun and sky icon on the button was present for all participants. However, as the controls were always explained, the bias of the visible icon on the button should not have caused a great enough impact for this third version to become the most usable version, or for the button toggle to become less usable than the wrist toggle. Although, it might have influenced the usability results for the third version, making it more or less usable than the fourth version it is currently tied with. As previously mentioned, we always explained the controls, meaning we cannot say, which type is more intuitive, but it would be interesting to evaluate that in the future, with an icon on the button and possibly an icon showing the use of the wrist type.

In terms of the questionnaire, it confused some of the participants, as it contained two sets of the approximate same 10 questions, one just for the toggling feature and the other for the UI itself. Those who were confused, commented and asked about it to be sure they understood it correctly, and after we explained it, they would go back and correct their answers if they had misunderstood something.

From the interviews, one participant suggested adding an icon for the fireplace button, which can be implemented in a future cycle, as we also observed several participants who had trouble finding the button for the fireplace. They were trying to find it around the virtual fireplace and needed to be informed that the button was on the UI. Even then, some participants moved the UI closer to find the button, and then away again to hit it. This suggests that the button for the fireplace should be more visible, perhaps bigger and with an icon. Alternatively, the button could also be placed on the fireplace itself, however, that would require the user to move to the fireplace and directly interact with the environment, whereas the UI is

always readily available on the controller.

In general, the participants had problems being precise when moving the sliders, which makes sense, as they had to hold both the UI and the laser pointer steady while moving the sliders. Correspondingly, Lindeman, Sibert, and Templeman [17] argued that it is difficult to be precise in VR when using your hands without a physical surface for support, as the hands are unsteady. In addition, Lindeman, Sibert, and Templeman [17] found that adding a physical surface made the participants perform faster and more accurately than without it, and the participants also preferred using the physical surface. This is worth considering for future improvements, however, their experiment used hand tracking instead of HTC Vive controllers.

Chapter 8

Conclusion

VR can serve as a crucial tool for visualisation of architectural works, as it can have many benefits not only for the architect, but for the target group of people such as, investors and clients. The aim of the project is to create a framework for a believable and realistic atmosphere for virtual architectural works, where the atmosphere itself is heavily impacted by the illumination of the scene. After the investigation of illumination in VEs in Chapter 4, we formulated a list of requirements that are found to be necessary for creating a believable and realistic atmosphere. The requirements include the use of an HDR camera with tonemapping, real-time rendering with a satisfactory level of performance, daylight cycle illumination, direct and indirect shadowing, automatic changes to the intensity of the sun based on the time of day, volumetric light scattering and the implementation of a realistic sky.

We created a scene in VR, using the Unity game engine that allows for the user to explore an architectural work through the HTC Vive.

We collaborated with architect Jacob Hilmer using the participatory design method in order to evaluate each design cycle. The second cycle produced a VR scene that implemented all the previously mentioned requirements and the collaborator found it satisfying in terms of both visuals and performance. Although the scene achieved its main goal in being believable and realistic to our collaborator, further design cycles are needed for future improvement.

Furthermore, a user evaluation was conducted in order to create a more user-friendly UI system. The evaluation compares two ways of toggling the UI on and off, where in one, the UI is toggled on by the user looking at his wrist, and in the other it was toggled by the user pressing a button. In addition, two ways of showing UI information are also compared, one being text information and the other being icons. 16 participants used the UI to control different elements in the

scene and afterwards completed a SUS questionnaire and participated in a short semi-structured interview. The user evaluation showed that the button toggle and the UI with icons were found not only to be rated the highest in the SUS, but were also preferred by the participants.

Bibliography

[1] Luke Ahearn. *3D Game Environments: Create Professional 3D Game Worlds*. en. CRC Press, Mar. 2017. ISBN: 978-1-317-41816-0.

[2] Jeremy Birn. *Digital Lighting and Rendering*. en. New Riders, Nov. 2013. ISBN: 978-0-13-343917-5.

[3] Barbaros Bostan and Ozhan Tingoy. "Game Design and Gamer Psychology". en. In: *Gamer Psychology and Behavior*. International Series on Computer Entertainment and Media Technology. Springer, Cham, 2016, pp. 105–121. ISBN: 978-3-319-29903-7 978-3-319-29904-4. DOI: 10.1007/978-3-319-29904-4_7. URL: https://link.springer.com/chapter/10.1007/978-3-319-29904-4_7 (visited on 02/20/2018).

[4] John Brooke. "SUS: A quick and dirty usability scale." In: *Usability Evaluation in Industry* 189.194 (1996), pp. 4–7.

[5] Gernot Böhme. "Atmosphere as the subject matter of architecture". In: *Herzog & DeMeuron: Natural History* (2006), pp. 398–407.

[6] Gernot Böhme. "The art of the stage set as a paradigm for an aesthetics of atmospheres". en. In: *Ambiances. Environnement sensible, architecture et espace urbain* (Feb. 2013). ISSN: 2266-839X. URL: http://journals.openedition.org/ambiances/315 (visited on 02/21/2018).

[7] Hervé Descottes and Cecilia Ramos. *Architectural Lighting: Designing with Light and Space*. New York, UNITED STATES: Princeton Architectural Press, 2011. ISBN: 978-1-61689-209-8. URL: http://ebookcentral.proquest.com/lib/aalborguniv-ebooks/detail.action?docID=3387533.

[8] Tim Edensor. "Light design and atmosphere". en. In: *Visual Communication* 14.3 (Aug. 2015), pp. 331–350. ISSN: 1470-3572. DOI: 10.1177/1470357215579975. URL: https://doi.org/10.1177/1470357215579975.

[9] Rickard Hagström. *Frames That Matter : The Importance of Frames per Second in Games*. eng. 2015. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-263379 (visited on 05/25/2018).

[10] S. M. Halawani and M. S. Sunar. "Interaction between Sunlight and the Sky Colour with 3D Objects in the Outdoor Virtual Environment". In: *2010 Fourth*

Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation. May 2010, pp. 470–475. DOI: 10.1109/AMS.2010.96.

[11] Hendrik Haupt. *Enviro - Sky and Weather - Asset Store*. Mar. 2018. URL: https://assetstore.unity.com/packages/tools/particles-effects/enviro-sky-and-weather-33963 (visited on 05/24/2018).

[12] Nico Hempe. "Real-Time Rendering Approaches for Dynamic Outdoor Environments". en. In: *Bridging the Gap between Rendering and Simulation Frameworks*. Springer Vieweg, Wiesbaden, 2016, pp. 69–140. ISBN: 978-3-658-14400-5 978-3-658-14401-2. DOI: 10.1007/978-3-658-14401-2_4. URL: https://link-springer-com.zorac.aub.aau.dk/chapter/10.1007/978-3-658-14401-2_4 (visited on 03/09/2018).

[13] Anders Hermund and Lars Simon Klint. "VIRTUAL AND PHYSICAL ARCHITECTURAL ATMOSPHERE". In: June 2016.

[14] Jacob Hilmer. *Jacob Hilmer Architecture*. 2018. URL: http://jharch.dk/ (visited on 05/28/2018).

[15] klankbeeld. *Freesound*. Feb. 2013. URL: https://freesound.org/people/klankbeeld/ (visited on 05/27/2018).

[16] Hoshang Kolivand and Mohd Shahrizal Sunar. "Real-Time Sky Color with Effect of Sun's Position". In: *International Journal of Scientific and Engineering Research* 2 (Jan. 2011), pp. 2229–5518.

[17] R. W. Lindeman, J. L. Sibert, and J. N. Templeman. "The effect of 3D widget representation and simulated surface constraints on interaction in virtual environments". In: *Proceedings IEEE Virtual Reality 2001*. Mar. 2001, pp. 141–148. DOI: 10.1109/VR.2001.913780.

[18] John Mardaljevic. "Daylight, Indoor Illumination, and Human Behavior". en. In: *Sustainable Built Environments*. Springer, New York, NY, 2013, pp. 69–111. DOI: 10.1007/978-1-4614-5828-9_456. URL: https://link-springer-com.zorac.aub.aau.dk/referenceworkentry/10.1007/978-1-4614-5828-9_456 (visited on 03/09/2018).

[19] Maic Masuch and Niklas Röber. "Game graphics beyond realism: Then, now and tomorrow". In: *Level UP: Digital Games Research Conference. DIGRA, Faculty of Arts, University of Utrecht*. 2004.

[20] J. Mortensen et al. "Real-Time Global Illumination for VR Applications". In: *IEEE Computer Graphics and Applications* 28.6 (Nov. 2008), pp. 56–64. ISSN: 0272-1716. DOI: 10.1109/MCG.2008.121.

[21] Michael J. Murdoch, Mariska G. M. Stokkermans, and Marc Lambooij. "Towards perceptual accuracy in 3D visualizations of illuminated indoor environments". In: *Journal of Solid State Lighting* 2 (Dec. 2015), p. 12. ISSN: 2196-1107. DOI: 10.1186/s40539-015-0029-6. URL: https://doi.org/10.1186/s40539-015-0029-6.

[22] Worawan Natephra et al. “Integrating building information modeling and virtual reality development engines for building indoor lighting design”. en. In: *Visualization in Engineering* 5.1 (Dec. 2017), p. 19. ISSN: 2213-7459. DOI: 10.1186/s40327-017-0058-x. URL: https://link-springer-com.zorac.aub.aau.dk/article/10.1186/s40327-017-0058-x (visited on 05/07/2018).

[23] Vitalee Oculov. *UNITY 5 AND UE4 COMPARISON*. en. 2016. URL: http://not-lonely.com/blog/making-of/unity-ue-comparison/ (visited on 05/15/2018).

[24] Fumio Okura, Masayuki Kanbara, and Naokazu Yokoya. “Aerial full spherical HDR imaging and display”. en. In: *Virtual Reality* 18.4 (Nov. 2014), pp. 255–269. ISSN: 1359-4338, 1434-9957. DOI: 10.1007/s10055-014-0249-x. URL: https://link-springer-com.zorac.aub.aau.dk/article/10.1007/s10055-014-0249-x (visited on 05/07/2018).

[25] Pluralsight. *Unreal Engine 4 vs. Unity: Which Game Engine Is Best for You?* en. Nov. 2014. URL: https://www.pluralsight.com/blog/film-games/unreal-engine-4-vs-unity-game-engine-best (visited on 05/16/2018).

[26] Matthew Regan and Ronald Pose. “Priority Rendering with a Virtual Reality Address Recalculation Pipeline”. In: *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’94. New York, NY, USA: ACM, 1994, pp. 155–162. ISBN: 978-0-89791-667-7. DOI: 10.1145/192161.192192. URL: http://doi.acm.org/10.1145/192161.192192 (visited on 05/25/2018).

[27] Siobhan Francois Rockcastle. “Perceptual Dynamics of Daylight in Architecture”. en. In: (2017). DOI: 10.5075/epfl-thesis-7677. URL: https://infoscience.epfl.ch/record/228895 (visited on 05/07/2018).

[28] Kevin K. Rooney, Robert J. Condia, and Lester C. Loschky. “Focal and Ambient Processing of Built Environments: Intellectual and Atmospheric Experiences of Architecture”. English. In: *Frontiers in Psychology* 8 (2017). ISSN: 1664-1078. DOI: 10.3389/fpsyg.2017.00326. URL: https://www.frontiersin.org/articles/10.3389/fpsyg.2017.00326/full (visited on 02/20/2018).

[29] Jeff Sauro. *Measuring Usability with the System Usability Scale (SUS)*. en. Feb. 2011. URL: https://measuringu.com/sus/ (visited on 05/15/2018).

[30] David Seamon. “Architecture, Place, and Phenomenology: Buildings as Lifeworlds, Atmospheres, and Environmental Wholes”. In: *Phenomenology and Place*. 2017, pp. 247–264. (Visited on 02/20/2018).

[31] Jesper Simonsen and Toni Robertson. *Routledge International Handbook of Participatory Design*. Routledge International Handbooks. Routledge, 2013. ISBN: 978-1-136-26619-5.

[32] M. Slater et al. “Visual Realism Enhances Realistic Response in an Immersive Virtual Environment”. In: *IEEE Computer Graphics and Applications* 29.3 (May 2009), pp. 76–84. ISSN: 0272-1716. DOI: 10.1109/MCG.2009.55.

[33] Unity Technologies. *Lighting and Setup*. en. URL: https://unity3d.com/learn/tutorials/projects/creating-believable-visuals/lighting-and-setup (visited on 05/24/2018).

[34] Unity Technologies. *Lighting Strategy*. en. URL: https://unity3d.com/learn/tutorials/projects/creating-believable-visuals/lighting-strategy (visited on 05/24/2018).

[35] Unity Technologies. *Preparing Unity Render Settings*. en. URL: https://unity3d.com/learn/tutorials/projects/creating-believable-visuals/preparing-unity-render-settings (visited on 05/24/2018).

[36] Unity Technologies. *Understanding Post Process Features - Unity*. URL: https://unity3d.com/learn/tutorials/projects/creating-believable-visuals/understanding-post-process-features?playlist=50020 (visited on 05/24/2018).

[37] Unity Technologies. *Unity - Manual: Anti-aliasing*. en. URL: https://docs.unity3d.com/Manual/PostProcessing-Antialiasing.html (visited on 05/24/2018).

[38] Unity Technologies. *Unity - Manual: Global Illumination*. en. URL: https://docs.unity3d.com/Manual/GIIntro.html (visited on 05/24/2018).

[39] Unity Technologies. *Unity - Manual: Light Probes*. en. URL: https://docs.unity3d.com/Manual/LightProbes.html (visited on 05/24/2018).

[40] Unity Technologies. *Unity - Manual: Linear or gamma workflow*. en. URL: https://docs.unity3d.com/Manual/LinearRendering-LinearOrGammaWorkflow.html?_ga=2.23461178.1612576461.1525768578-1879192485.1525081442 (visited on 05/24/2018).

[41] Unity Technologies. *Unity - Manual: Procedural Materials*. July 2017. URL: https://docs.unity3d.com/560/Documentation/Manual/ProceduralMaterials.html (visited on 05/24/2018).

[42] Unity Technologies. *Unity - Manual: Reflection probes*. en. URL: https://docs.unity3d.com/Manual/ReflectionProbes.html (visited on 05/24/2018).

[43] Unity Technologies. *Unity - Manual: Shadows*. en. May 2018. URL: https://docs.unity3d.com/Manual/ShadowOverview.html (visited on 05/25/2018).

[44] Unity Technologies. *Unity Particle Pack - Asset Store*. May 2018. URL: https://assetstore.unity.com/packages/essentials/asset-packs/unity-particle-pack-73777 (visited on 05/24/2018).

[45] A. G. Voloboi et al. "Simulation of natural daylight illumination determined by a high dynamic range image". en. In: *Programming and Computer Software* 32.5 (Oct. 2006), pp. 284–297. ISSN: 0361-7688, 1608-3261. DOI: 10.1134/S0361768806050057. URL: https://link-springer-com.zorac.aub.aau.dk/article/10.1134/S0361768806050057 (visited on 03/09/2018).

[46] Changbo Wang. "Real-Time Rendering of Daylight Sky Scene for Virtual Environment". en. In: *Entertainment Computing – ICEC 2007*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2007, pp. 294–303. ISBN: 978-

3-540-74872-4 978-3-540-74873-1. DOI: 10.1007/978-3-540-74873-1_36. URL: https://link-springer-com.zorac.aub.aau.dk/chapter/10.1007/978-3-540-74873-1_36 (visited on 03/09/2018).

[47] I. Yu et al. "Visual Realism Enhances Realistic Response in an Immersive Virtual Environment - Part 2". In: *IEEE Computer Graphics and Applications* 32.6 (Nov. 2012), pp. 36–45. ISSN: 0272-1716. DOI: 10.1109/MCG.2012.121.

[48] Xiao Yu. "Research and Practice on Application of Virtual Reality Technology in Virtual Estate Exhibition". In: *Procedia Engineering*. CEIS 2011 15 (Jan. 2011), pp. 1245–1250. ISSN: 1877-7058. DOI: 10.1016/j.proeng.2011.08.230. URL: http://www.sciencedirect.com/science/article/pii/S1877705811017310.

Appendix A

Usability Questionnaire

All usability statements are rated on a five-point Likert scale ranging from strongly disagree to strongly agree.

A.1 Interaction Method Usability

The usability statements used to obtain the SUS score for toggling the graphical user interface on and off.

1. I think that I would like to use this way of toggling the user interface on and off frequently.
2. I found this way of toggling the user interface on and off unnecessarily complex.
3. I thought this method of toggling the user interface on and off was easy to use.
4. I think that I would need the support of a technical person to be able to toggle the user interface on and off this way.
5. I found the various functions of toggling the user interface were well integrated.
6. I thought there was too much inconsistency in this way of toggling the user interface on and off.
7. I would imagine that most people would learn to use this way of toggling the user interface on and off very quickly.

8. I found this way of toggling the user interface on and off very cumbersome.
9. I felt very confident using this way of toggling the user interface on and off.
10. I needed to learn a lot of things before I could get going with toggling the user interface.

A.2 Graphical User Interface Usability

The usability statements used to obtain the SUS score for the general UI, in order to compare the text and icon versions of the UI.

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