PROJECT GREEN LAGOON

A thesis submitted to Academy of Creative Technology (ACT)
for the Study Program Creative Media and Game Technologies
of Hogeschool Saxion University of Applied Sciences Enschede
In Partial Fulfillment of the Requirements
for the Degree Bachelor of Science
[June 2021]

Written by
Benjamin Pazaurek
439432, CMGT, 2021

Graduation Supervisor
Matthijs van Veen

Graduation Coach
Herman Statius Muller

Client
Thales Netherlands

Company Supervisor
Ilse Driessen

Acknowledgments

This project could not have been possible without Ilse Driessen, Jeroen van der Wel, and Karol Osipovic being patient, understanding clients, and active partners during feedback sessions.
I wish to express my gratitude to Marcella Rulof Veldkamp for helping me with all the paperwork and preparation concerning my application and starting my work at Thales.
I want to thank Herman Statius Muller for always being available to provide support and feedback.
I want to commend Annemieke Muis for being a consistent and hard-working teammate and a friend.
I recognize the staff of the XR Lab for being extremely proactive and helpful and the students who participated in the playtest sessions.
I am eternally grateful to Marieke Hobiger and Vero Nagy for providing indispensable feedback about my thesis.
I can never repay all the support I got from my father, who financed my studies and living in a foreign country.
I want to show my appreciation for the great and inspiring teachers at Saxion who helped and guided me through all these years.
Last but not least, I want to thank all my friends I met during my studies in the Netherlands for sharing both happiness and sorrow.

Abstract

This thesis aims to answer the question '**How to create a well-optimized, photorealistic Virtual Reality learning environment in Unity to instruct users about radar maintenance, in which they feel relaxed and comfortable?'** for the client Thales. This international military contractor company is the largest supplier of radar systems in the Netherlands. Thales wanted a new VR learning environment for radar maintenance.
The main problem was that a previous VR learning environment was too artificial and discomforting and had performance issues. The proposed solution was to research and create a photorealistic and relaxing prototype that runs well on the target platform and fits Thales's company profile.
Research and development were structured based on the 'Design Thinking Model'. Online desk research was used to gather already available data. Previous experience in 3D modeling, texturing, sculpting, rendering was utilized. Weekly meetings with Thales provided a constant flow of feedback and enabled iterative development. The testing methods included technical performance tests to inspect optimization, and playtesting and interviews for feedback.
Based on online desk research, client discussions, and player feedback, a bright, serene VR learning environment was created. It was set in a lagoon surrounded by green nature scenery. The playable area consisted of an observatory and a platform on which the maintenance would take place.
Feedback gathered during playtest sessions aligned with the findings, and testers reported that the scene had an overall relaxing atmosphere. Though photorealism proved to be too ambitious for the size of the team, Thales was satisfied with the final visuals.
Since the target platform was VR, it was crucial to have stable frame rates of at least 90. Otherwise, the players may have experienced discomfort and nausea. The key aspects of optimization were the number of triangles, draw calls, and render settings. Optimizing for VR proved to be challenging and time-consuming. Overall, optimization was successful, and stable 90 FPS was reached in the final build. The clients were content with these results.

Preface

In this chapter, I give general information about what happened during the graduation period. Later in this report, I will refer to myself in the third person as ‘the student’. Details important for the project but not directly linked to the questions or research are mentioned here.

I worked on this project together with Annemieke Muis between February 2021 and July 2021. The project’s duration was twenty weeks, most of which were active workdays apart from holidays. We agreed to work from 9:00 until 17:00 with roughly 45 minutes of break time. Both of us worked from home due to the COVID19 regulations, except for testing at Saxion’s XR Lab roughly once per week. Meeting with our clients and our coach happened regularly online. Our company coach was Ilse Driessen. She had the final say about art and visual questions. Jeroen van der Wel was our other client. He could not always attend our meeting, but he gave relevant information about the radars and frequently shared useful input. Our graduation coach was Herman Statius Muller, an expert in 3D visual effects, so he could often provide valuable feedback. Herman also convinced us to create a Blog to help us monitor the weekly progress, design changes, and feedback.
We used Scrum with an online tool designed for game developers called Hack and Plan to keep track of our planning. Later during the project, we set up GitHub for version control, which worked out well for a few weeks, but later the file sizes became too large. We decided to backup zipped versions of the project on Google Drive, just like all our other documents and files.
Please note that some of the findings were moved to the Appendices due to reaching the word limit. I used American English.
In week 11, I had a minor domestic accident with lasting effects. I notified Thales and tried to make up for this by working one hour longer every day after taking a few days off.
Despite some of the unfortunate events, I enjoyed working on this project. Annemieke was great to work with, and our clients were active and invested. The graduation period was a valuable experience, and I look forward to starting my career.

List of Abbreviations

2D	Second Dimension
3D	Third Dimension
CPU	Central Processing Unit
FPS	Frames per Second
GPU	Graphics Processing Unit
HDRP	High-Definition Rendering Pipeline
LOD	Level of Detail
OC	Occlusion Culling
PC	Personal Computer
RAM	Random Access Memory
URP	Universal Rendering Pipeline
VR	Virtual Reality
XR	Extended Reality

Glossary

3D Model	Digital representations of objects in third dimension coordinates.
4K Resolution	Referring to a screen of 3840 by 2160 pixels.
4K Texture	Referring to an image of 4096 by 4096 pixels.
Baking/ Light Baking	The process of pre-calculating lights and shadows on a scene and using those values instead of real-time lighting, usually to save processing power.
Baking/ Texture Baking	The process of projecting texture data from a more detailed, high-polygon model unto a low-polygon model.
Draw Calls	A number referring to all the objects that the GPU must render.
Face	A unit of a 3D model consisting of four vertices.
Hardware	Physical components of a computer.
LOD	In the context of 3D scenes, it refers to objects having fewer triangle and texture resolution details depending on how far they are from the main camera.
OC	An optimization method. It describes the process of not rendering objects that are fully obstructed by other objects in the camera's view.
Optimization	Methods that reduce computational costs on a computer's hardware.
Pipeline	A pipeline refers to a sequential order of all the necessary tasks to be done to achieve a certain product.
Plugin	A software that adds additional features to an existing program.
Rendering	The process of displaying certain 3D models on a screen as 2D images.
Shader	A program that changes how certain visual aspects are rendered.
Software	Virtual electronic data run by a computer's hardware.
Texture	A 2D image meant for the surface of 3D models. It may define various details, such as color, bumpiness, glossiness, etc.
Triangle	A unit of a 3D model consisting of three vertices.
UV, UV Mapping	A UV map is a laid out, 2D representation of the faces of a 3D model.
Vertex	A single point of a 3D model.

Table of Contents

I. Introduction

This thesis is written by graduating student Benjamin Pazaurek, hereafter referred to as ‘the student’, from the study ‘Creative Media and Game Technologies’ of Hogeschool Saxion University of Applied Sciences Enschede. Thales Netherlands proposed this graduation project.
The student worked in a team of two. His teammate was Annemieke Muis from the same study. Although the research and production tasks are shared, it must be mentioned that the Graduation Report was done individually.
This document aims to showcase the research and development process of the graduation period and prove that the student is ready to work professionally according to the competencies defined by Saxion. It critically evaluates the client’s objective and the direction to follow for the related research. The company, the problem, and the related questions are presented. It then showcases the ideas, concepts, and prototypes made by the team. Results are concluded based on testing and feedback. Finally, recommendations are given to improve the product further.

II. Assignment Description

The project’s goal was to create a prototype of a photorealistic VR learning environment in Unity. This 3D environment was going to be part of a scene in which radar maintenance would be taught. It had to be an environment people find relaxing, comfortable and reflects Thales’s look and feel. Thales already possessed 3D models of various radar systems; the graduation team needed to consider these dimensions when designing the scene.
Research had to be done regarding the ways to achieve photorealism while keeping the performance high. Furthermore, Thales wanted documentation detailing the differences between Unity’s Universal and High-Definition rendering pipelines. The final scene had to be rendered using the most suitable option based on research and tests. A basic VR movement system, such as teleportation, also had to be set up for testing purposes.
According to Thales, the target audience was maintenance personnel in training from anywhere worldwide who had little to no experience in VR, which had to be considered when designing the environment.

III. Company Outline

Thales is an international company working on contracts and projects for militaries all around the globe. The student was working for Thales Netherlands, situated in Hengelo. Their motto is ‘creating a future we can trust’. Thales is the largest supplier of radar systems in the Netherlands. As stated on their website, *“From the bottom of the oceans to the depth of space and cyberspace, we help our customers think smarter and act faster - mastering ever greater complexity and every decisive moment along the way.”* (Thales, 2021). The company’s logo is shown below, in Figure 1.

During this project, the company’s preferences were definitive. If they did not support an idea, the student respected that and adjusted accordingly. The contact persons from Thales and the team agreed to have weekly online meetings. Feedback was given and applied frequently. The company supervisor was Ilse Driessen, an expert in 3D design. The secondary contact person was Jeroen van der Wel, the head of XR-Innovations.

Figure 1: The logo of Thales

IV. Problem Definition

In the past, Thales tried a similar application of a learning environment in VR. It was in the form of a ship's engine room. However, they were not satisfied with the graphical quality of the scene. They claimed that the users could not immerse themselves in the experience and that the environment was too artificial and uncomfortable overall.
Thales also mentioned performance issues on their previous hardware because of the extreme polygon count of their radar models. According to Thales, the largest model has a polygon count of more than 40 million while also consisting of hundreds of meshes. This number is much higher than the average system can handle in real-time. According to Unity, draw calls, rendering individual objects in a scene are especially taxing on the system (2020).
Since the target medium is VR, it is even more critical to have stable FPS rates of at least 90. Otherwise, the players may experience discomfort and nausea (Oculus, n.d.). The team communicated to Thales that this might be a problem and recommended further optimization of these models. Unfortunately, the only disclosure of the project was that Thales could not share the radar models for confidentiality reasons. Thales claimed that the radars are not the team's responsibility, and they have specialists to fix any possible problems regarding performance caused by the radars. To quote the email of Thales's 3D geometry specialist, Karol Osipovic, *"[…] we are R&D department and we focus on innovation and results."* (2021).
Thales claimed to own a high-end PC with the latest Nvidia 3090 graphics card, an Intel I9 processor, and 32 gigabytes of RAM, which was expected to be more than capable of running a scene that fits their requirements. In the early stage of the project, the only assumed impediment regarding performance was the extreme polygon count of the radar models. Unfortunately, many performance issues arose during production, which had to be addressed despite not having the radar models on the scene. Achieving high FPS in VR proved to be the most challenging aspect of the project.

Thales's end goal with this project is to sell the training software, including the PC and VR hardware needed to use it, to countries all around the world. These radars are hard to access otherwise, and this way, the users could join and learn about them wherever they might be. Thales also considered using cloud-XR in the near future to cut down the costs on hardware dramatically. Cloud-XR works by running an application on a remote server streamed for and controlled by the user. This technology is not widely available yet, but Thales expects it to revolutionize the VR industry in a few years. By creating a visually pleasing and well-optimized prototype, Thales can expand on this concept.

The student's role in solving the problem was research and development: gathering information to understand and define the problems, finding solutions, then creating and testing a prototype using these findings and his 3D art skills.

V. Research Questions and Approach

In this chapter, the project's main question is defined and divided into sub-questions to understand the problem better. Furthermore, suitable research methods for each sub-question are mentioned.

V.1. Main Question

How to create a well-optimized, photorealistic Virtual Reality learning environment in Unity to instruct users about radar maintenance, in which they feel relaxed and comfortable?

V.2. Sub Questions and Approach

To answer the Main Question, it must be broken down into Sub Questions.

1. What is a comfortable, realistic environment that fits Thales's company profile?

To answer this question, the clients actively took part in frequent ideating and feedback sessions, at least twice per week, in online meetings and emails. Furthermore, online desk research was done regarding Thales's style, shapes, and colors. Moodboards were made accordingly. It was expected that the heavy involvement of the clients would result in designing an environment that satisfied them.

2. What climate, season, and colors are generally considered relaxing?

To answer this question, qualitative online desk research was conducted on the topics of nature and colors. It was expected to find many articles backing up the theory of nature having a positive effect on the human brain. Playtest sessions with observation and questionnaires were done at the XR Lab for general feedback about the environment.

3. How to create optimized tree assets?

To answer this question, qualitative online desk research and general practice were done. The student has created tree assets before by 3D sculpting, which can be a time-consuming task. Blogs and forums were visited to learn about possible alternatives to creating tree models manually. Based on research, the student decided to learn a tool called SpeedTree. Online video tutorials and general practice were utilized.

4. What is the difference between the Hight-Definition Rendering Pipeline and the Universal Rendering Pipeline in Unity?

To answer this question, qualitative online desk research was conducted. Unity's official documentation on the topic was one of the leading guidelines. Additionally, video tutorials were followed. Although the scene was planned to be tested in both render setups, the team only had time to create and optimize the scene in URP.
It was expected that testing would take a long time, partially because of light baking. Rendering the scene required a fast, reliable PC. Fortunately, such a computer was available at Saxion's XR Lab.

5. How to optimize a scene for VR?

5.A. What exact methods can be used to optimize a scene in Unity?
5.B. How are similar VR simulations' graphics and performance on the target hardware?

To answer these questions, qualitative online desk research was conducted to find and collect exact optimization methods. The sources included the latest official Unity manual, relevant video tutorials, and articles. Hardware forums, where similar VR games' performance charts on the highest settings on the target hardware are compared, were analyzed and taken as reference. It was expected to find that the target hardware can handle a very high poly scene. Though none of the students had a deep knowledge of testing performance, Unity's built-in tools proved sufficient for this purpose. The team expected to be able to optimize the lagoon scene well.

VI. Scope

VI.1. Deliverables

A thematically fitting, comfortable, photorealistic, well-optimized (90 FPS) VR demo scene in Unity, consisting of a lagoon, an observatory, a platform in the middle of the lagoon on which the training will take place, a server room below the platform, and ideally an underwater scene. A basic VR controls setup in the scene, such as teleportation. A report consisting of a chapter detailing the difference between URP and HDRP in Unity, and a reasoning on why the team chose one over the other; a chapter of research; a chapter of brainstorming, concepts, moodboards, ideas, and sketches; and a chapter of recommendations for future improvements.

VI.2. Exclusions

Because of the team's limited knowledge of coding, little to no programming was done. Any programming required is mentioned in the Recommendation chapter of this document. Thales was aware of this, reassuring the team that their focus had to be on the visuals and performance.
Although the scene was planned to be tested in two different render setups, the team only had time to create and optimize the scene in URP. Neither of the students worked with Unity's HDRP before, and the pipeline was considerably different. Quality in one render setup was prioritized over quantity. Thales accepted this as only one scene was supposed to be included in the final deliverables. For more details about why URP was chosen, see chapter **VIII.4.**. Most elements of the underwater idea mentioned in chapter **VIII.1.A.** also had to be discarded and put into the Recommendations because of time constraints.

VI.3. Constrains

Restrictions enforced due to Covid19 meant that the team worked from home with the exceptions of testing days at the XR Lab. This made work challenging initially, so the team kept in contact during workdays to check and motivate each other. The graduation schedule is shown in **Appendix I**.
Measures had to be made to optimize the asset creation and divide tasks equally.
The student had little experience with creating realistic foliage. He had to learn a tool called SpeedTree explicitly designed for that.
The team did not expect any expenses and aimed to work with free tools. Nonetheless, some exterior assets were used to save time or to achieve better results. See chapter **VIII.1.A**. for more details on the exterior assets used.

VII. Methodology

This chapter summarizes the methods used for research, planning, production, and testing.

The ‘Design Thinking Model' shown in **Appendix G** was used to structure research and the graduation project. It is a design methodology focusing on human needs and solutions. (Dam & Siang, 2020)
Online desk research was used to gather already available data. The sources included scientific articles, video tutorials, blogs, and technical forums from the Internet. According to the University of Maryland (n.d.), a credible source should be unbiased and backed up with evidence. Findings were evaluated based on the ‘CRAAP Test’ (Currency, Relevance, Authority, Accuracy, Purpose), a method the student learned during the course ‘Applied Research’ at Saxion.
For communication, the team used Discord between themselves, emails to keep the clients updated, and Microsoft Teams for weekly meetings with Herman and Thales. The student was responsible for formulating professional emails for the clients, but these were never sent without the review and approval of Annemieke to ensure agreement. Thales had issues connecting to Teams several times, so they requested to switch to Zoom, which ended up being their preferred application for the rest of the graduation period.
The team’s backup and sharing tool was primarily Google Drive. Initially, a GitHub repository was set up to have better control over the changes made by the team. Unfortunately, the project’s file sizes became too large even with Git’s Large File Storage, which caused many errors. The team eventually switched to using Google Drive as a version control tool. While one was working on the scene, the other usually produced assets because that did not require Unity. When one was finished with their tasks, the entire project folder was compressed, named the corresponding date, and uploaded to Google Drive. It turned out to be a slightly slower but more reliable solution.
The planning was based on Scrum, an agile development method. Because of the size of the team, Scrum was utilized more flexibly while retaining its core values of iteration and reflection, such as having daily standups, weekly development cycles called ‘sprints’, a backlog of features, and ‘sprint reviews’. The team used Hack and Plan, an online planning tool designed for game developers to work in ‘sprints’, organize their tasks, and log their hours. The ‘MoSCoW’ prioritization method was used to order tasks by importance. Furthermore, a blog was made and updated weekly based on the suggestion of the graduation coach.
Previous experience in Maya 3D modeling, Substance Painter texturing, ZBrush sculpting, and Unity rendering were utilized. Weekly meetings with Thales provided a constant flow of feedback, which enabled the team to adjust ideas and assets iteratively. The entire project was characterized by prototyping and iterative development. General practice was done when creating certain assets in tools such as ZBrush and SpeedTree.
The testing methods included technical performance tests to check optimization, and playtesting and interviews for feedback. During the playtest session, the participants were silently observed, and notes were taken of their verbal comments and non-verbal behavior. The testers were then briefly interviewed afterward. They were encouraged to express their opinion and were rewarded for their participation with sweets.

VIII. Research and Results

This chapter summarizes the answers the student found for the related sub-questions, then presents the achieved results for each. These findings may overlap with Annemieke Muis's research.

VIII.1. What comfortable, realistic environment associates with Thales and radar maintenance?

VIII.1.A. Research and Development

Early Concepts

This question was answered mainly through having frequent feedback sessions with the client. As it is usual for large projects, it began with a fair amount of uncertainty and many questions, even after having an online meeting with the company's contact persons, Ilse Driessen and Jeroen van der Wel. Many hours were spent emailing questions to Ilse, but it was necessary to get a clear overview of Thales's vision and their requirements. These requirements were to create an environment relevant for maintenance that is applicable for any radar, is an environment in which people feel comfortable, and has Thales's look and feel. As shown in **Appendix A**, the student created a moodboard to define Thales's shapes, colors, and feel.
Furthermore, research was done about simulations in VR to prove that they can be used effectively for training purposes. Song et al. (2020) researched crane operation training in virtual reality. The results showed that virtual training significantly improved the user's ability to handle a real crane: the more immersive the experience, the better the learning curve.

After getting a better understanding of the project, the student made a graph to summarize the requirements set by Thales and outline the research questions. Based on these, the team brainstormed about potential environments, and the research questions were also formulated. The student organized these possible scene ideas in a diagram derived from the requirements, shown in **Appendix B**.

The team picked three environments that fit the requirements and inspired them the most. The scenes selected were a secret cave, a floating artificial island, and an exhibition hall. Both students created one moodboard for each and brainstormed about details that made them more attractive. The student did concept modeling on the cave and the exhibition hall idea to share his vision. Two graybox scenes made by the student are shown in **Appendix C** and **Appendix D**. After discussing the ideas, the moodboards were combined into a final version per theme and then sent to Thales for their opinion.

Thales preferred a bright and natural scene, so the cave idea was discarded. To elaborate on the island and expo hall ideas, the student searched nature documentaries for inspiration. A solution to combine the best elements was found in the form of a lagoon, shown in Figure 2. This screenshot was taken from a video made by 'The World Travel Guy' (2018) on YouTube. The student created a top-view sketch of his vision based on this image, shown in Figure 3. A basic concept model was also made, which the team believed would fit Thales's wishes, shown in Figure 4. The starting area was an observatory in this concept, overviewing the main training area, situated inside a rocky lagoon with a stone arch above the water access. The clients also really liked this idea, and these elements became the cornerstones of the later iterations.

Figure 2: The lagoon picture that inspired the student

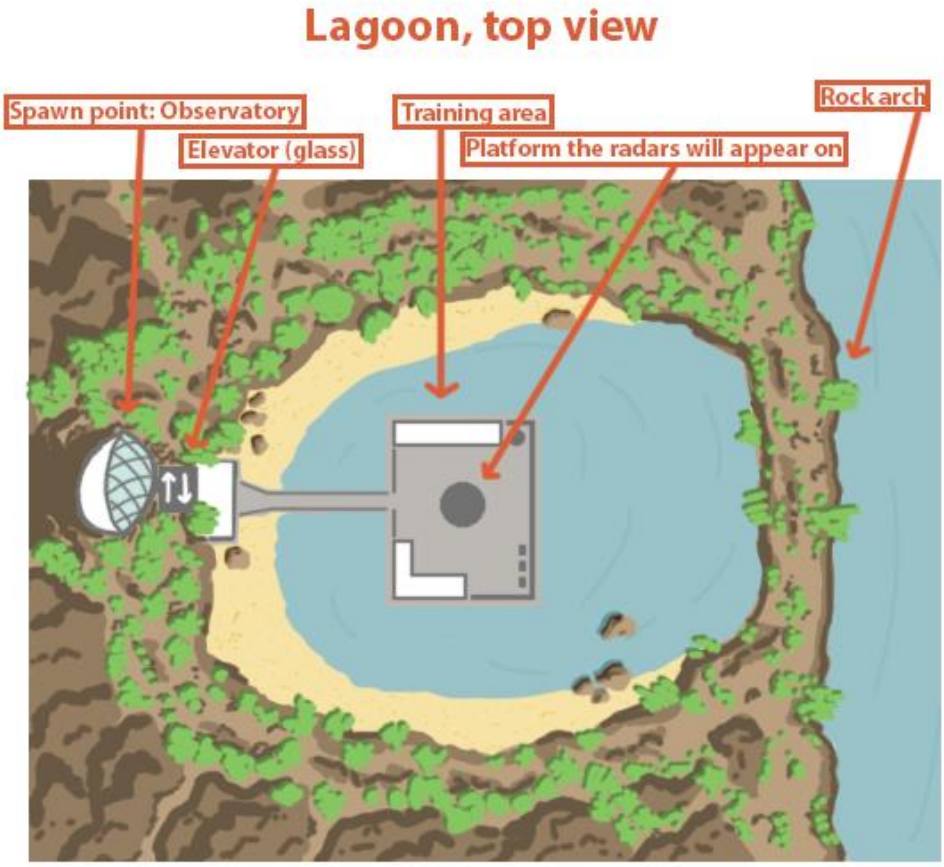

Figure 3: The lagoon sketch

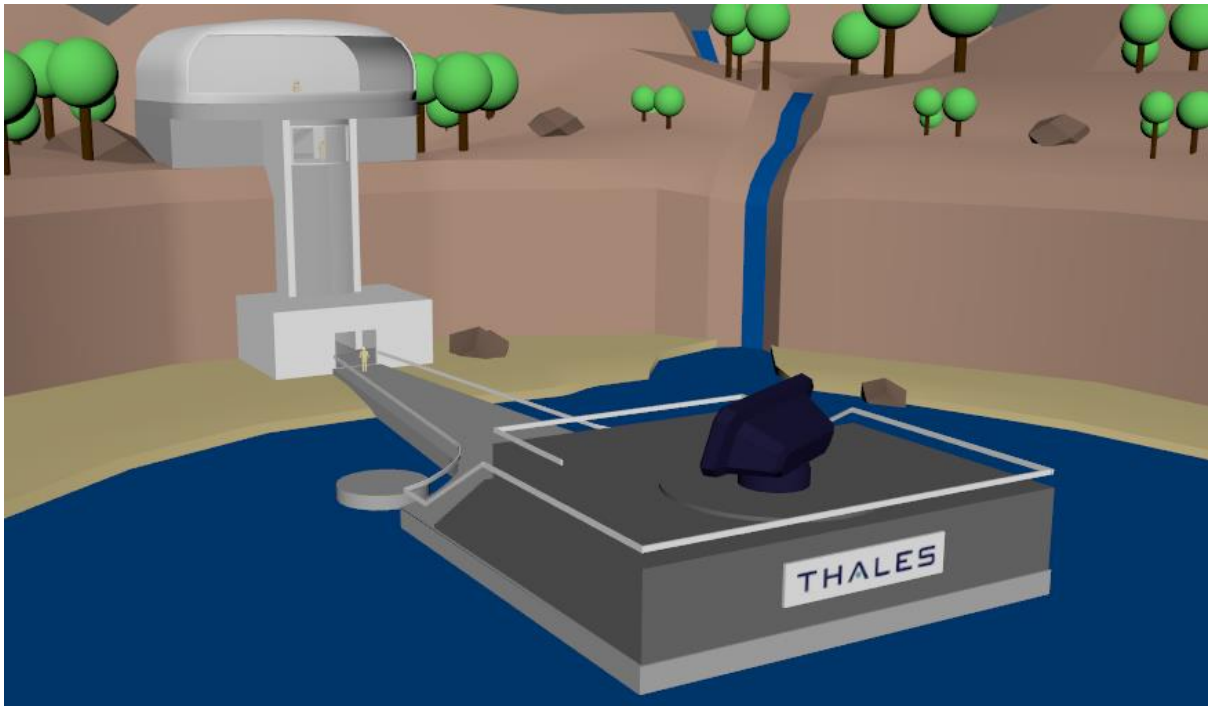

Figure 4 Early concept models

Over the weeks, many design changes had been made. The most notable changes included adding an underwater server room with a large window that would have overlooked an underwater scene, the complete removal of any elevators or stairs, and more circular and organic shape language, inspired by a theme called eco-futurism and the architecture of Vincent Callebaut. Examples of his work are shown in **Appendix H**. A corresponding version of the scene is shown in Figure 5.

Figure 5: The concept models, as of 2021 March 15

The concept model was imported into Unity. Dressing the scene and continuous replacement of placeholder assets commenced. The terrain was created using Unity. The environment was layered, the landscape was getting higher with distance, mountains being in the background. More details about the terrain are shown in chapter **VIII.5.A.**.
A free HDR skybox from Unity's Asset Store was implemented, called Skybox Series Free by Avionx (2018). During testing, every testplayer mentioned that they liked the skybox, which thus became the final one used. Some suggestions were made regarding moving clouds, which the team would have liked to implement. Unfortunately, there was no time left to do so.

Testing the scene in VR began in week four. The first test showed that scales had to be adjusted as some objects were too large. The platform was made smaller based on Jeroen's feedback. The lagoon was reduced in size so that the shoreline was closer to the platform. Further noteworthy changes at this stage included removing the dome and the rock archway because they obstructed the view. The idea of the tree-holders seen in Figures 5 and 6 was also discarded. As shown in Figure 7, teleport stands were added as travel between the top of the platform and the server room, replacing ladders. For more ideas regarding the platform, see **Appendix J**. Wind turbines were included on the scene to add simple but exciting motion points. The final wind turbine models were created and animated by the student.

Figure 6: The early Unity scene

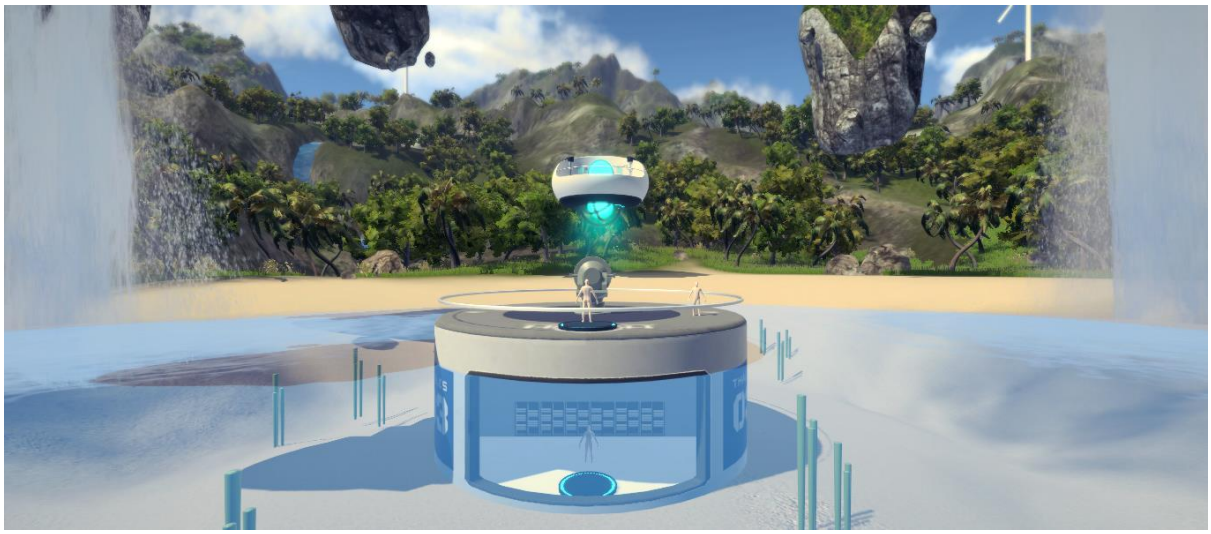

Figure 7: The scene in Unity URP, as of 2021 April 13

About the Trees

It was essential to define the type of trees that fit the scene. The team's findings showed that wide, dense canopies and short trunks are generally preferred. (Sommer & Summit, 1996) (Lohr & Pearson-Mims, 2006). The lusher the crown and the shorter the trunk, the more visually appealing the tree is. (Nelson, Johnson, Strong, & Rudakewich, 2002) (Gerstenberg & Hofmann, 2015). The studies of Gerstenberg & Hofmann (2015) and Nelson, Johnson, Strong, & Rudakewich (2002) also concluded that deciduous trees are preferable over conifers.
In the early concepts, palm trees were present in the scene because they were standard trees in Unity. They served as placeholders, though the team considered keeping a few to give the scene a vacation feel. However, they were later replaced entirely by another, more continental tree based on Thales's feedback because the palms made the scene feel too tropical, and they desired a more neutral scene. For more details about the trees, see chapter **VIII.3.A**..

About the Observatory

The final shape of the observatory was based on a concept made by Ilse, shown in Figure 8. Based on Ilse's concept, both students made their version, then combined the best elements. A large portal was added as the spawn point and potentially an exit point out of the experience. Annemieke's idea was to add plants to futuristic glass pillars. The student combined a wooden floor and steps with the clean white walls derived from the concepts of Vincent Callebaut, as seen in **Appendix H**. The observatory was later placed on the other side of the lagoon to overview the landscape, thus creating a more appealing scenery. Thales wished to further increase futurism by including holographic screens in the concepts. These were later implemented to display the controls at the spawn point.
In the earlier concepts, the observatory was suspended by the terrain. However, Thales liked the idea to have it levitate in the air because of the futuristic concept. The student made concepts for this and came up with an idea based on concept art from Titanfall 2 by Respawn Entertainment (2016), shown in Figure 9. A large gyroscope was placed on the bottom of the observatory, shown in Figure 10. It was animated to create compelling motion.

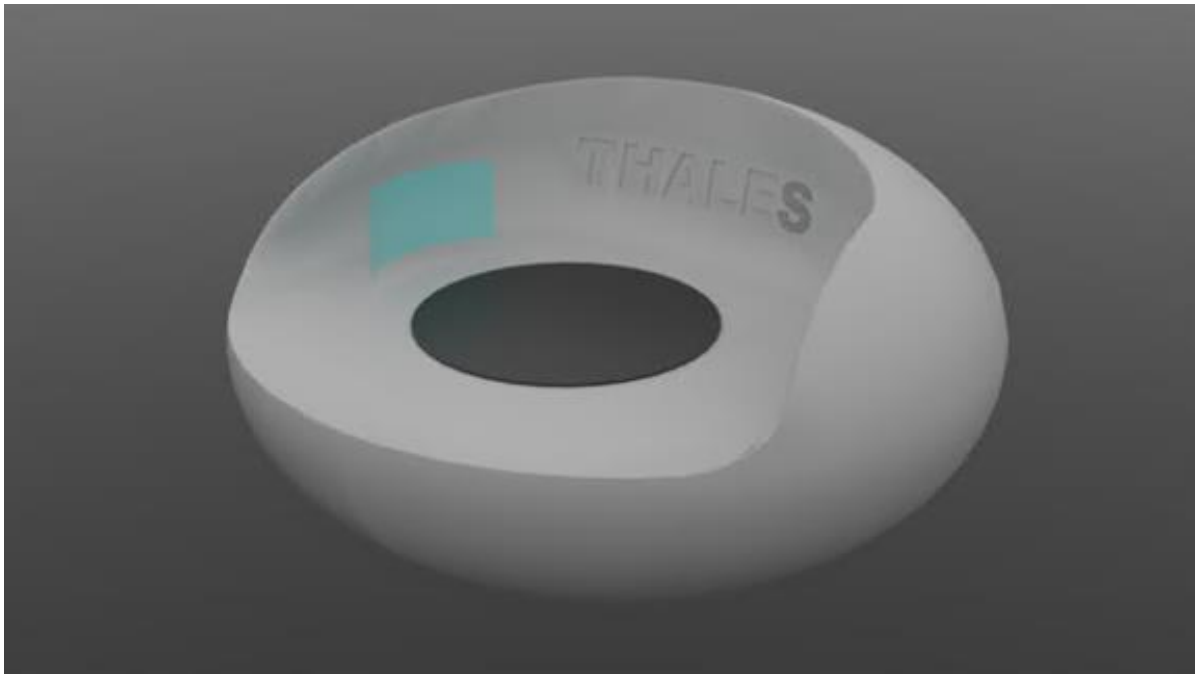

Figure 8: Concept of the observatory's shape made by Ilse

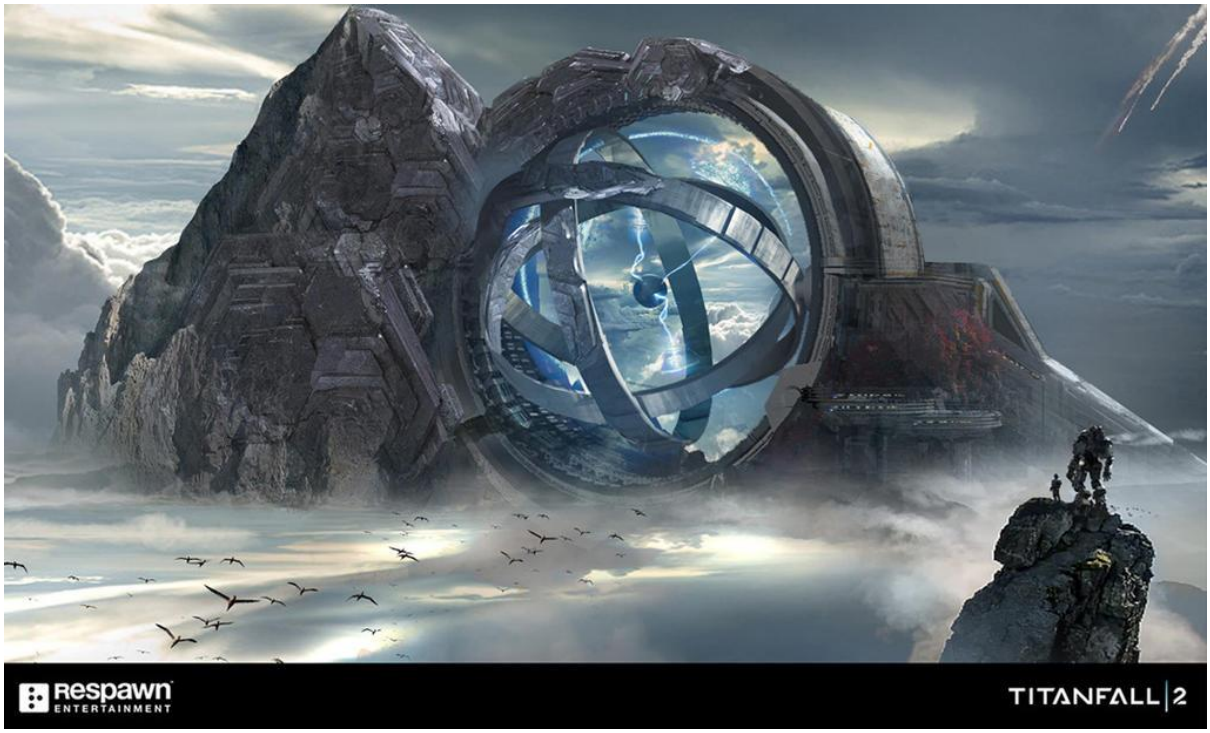

Figure 9: The gyroscope inspiration

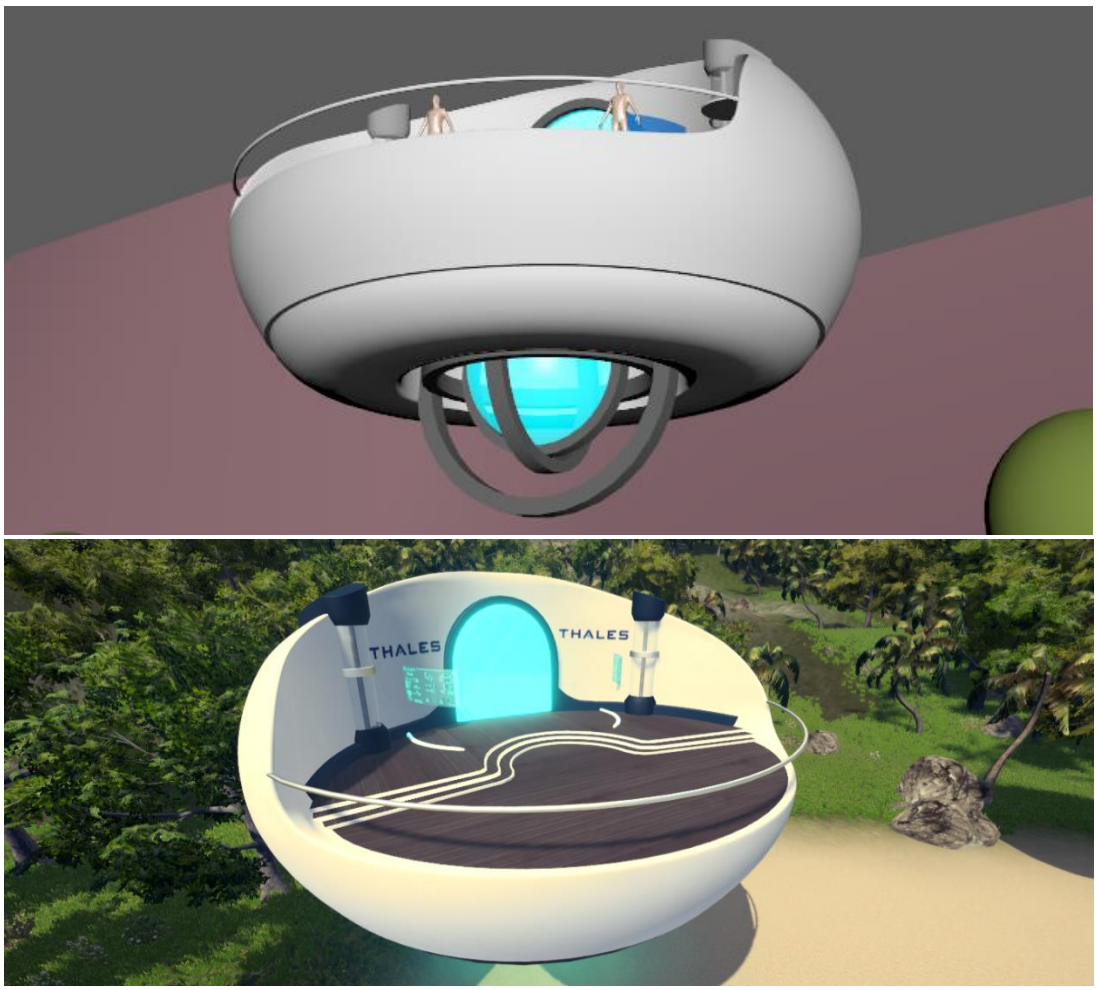

Figure 10: The gyroscope and the work-in-progress observatory

About the Floating Rocks

Thales has contracts with over 80 countries all around the world. Therefore, it was important to think of a culturally neutral scene. The team idea was to potentially include some fantasy elements in the design, which was reinforced by a suggestion from Jeroen to include floating rocks and waterfalls, as seen in the movie Avatar by James Cameron (2009). Assets were added and included in the final scene accordingly. One testplayer mentioned that they felt somewhat out of place and unrealistic. Fortunately, these rocks could be easily disabled in the hierarchy if needed.

Exterior Assets

The team did not expect any expenses and aimed to work with free tools. Nonetheless, some exterior assets were used to save time or to achieve better results. Thales provided two such priced packages from Unity's Asset Store, namely Gaia Terrain & Scene Generator by Procedural Worlds (2021) and Crest Ocean System by Wave Harmonic (2021).
The team also used free assets from Unity's Asset Store, such as Skybox Series Free by Avionx V4.2 (2018), Unity's OpenXR PluginV1.0.3, Standard Assets for Unity V1.1.6, Unity Terrain Tools Sample Asset Pack V1, Unity's Post Processing Stack V2, Unity Particle Pack V1.7, and the SteamVR Plugin, which later had to be removed.
Based on Herman's suggestion, the student downloaded free 3D rock models from the website TurboSquid by Shutterstock to save time. Three different free rock assets made by artists gizem dilara saatci (2021), ice kazim (2020), and Namabenama (2018), were used. These models were retextured to fit the environment, but their base mesh remained the same. The rocks were placed near the platform, sticking out of the sea to better use depth perception in VR.

About the VR Controls

It must be mentioned that setting up the VR controls proved to be quite challenging for the team because of their limited experience in coding. HTC Vive Pro was the platform used by Thales and at the XR Lab. Earlier, the team used an easy-to-setup plugin called SteamVR that had models of the controllers, a teleport indicator curve, a destination indicator circle, interactive hands, and fade effects when teleporting. SteamVR worked smoothly, but then the team switched to the latest Unity version as per the request of Thales, which did not support this plugin anymore. This compatibility problem could not be fixed, so a different VR controller had to be set up.
Gaia's built-in VR controllers were used later and during the first tests. This controller was much less advanced, and all testplayers mentioned that they had difficulties with it. Even though Herman believed that smooth VR controls should have been a priority, the focus shifted away from the interface because Thales reassured the team that such technical tasks were not expected.

About Sounds

3D sounds were added to the scene following player feedback, such as singing birds, seagulls, waves, and wind. This simple addition highly boosted the immersion and serenity of the scene. No interaction sounds were implemented.

VIII.1.B. Results

What comfortable, realistic environment associates with Thales and radar maintenance?

While photorealism proved to be too ambitious for the size of the team, a fairly realistic-looking and pleasant scene was created. The lagoon idea proved to fit the perquisites set by Thales, which were to create an environment relevant for maintenance that is applicable for any radar, is an environment in which people feel comfortable, and has Thales's look and feel. The heavy involvement of the clients and multiple updates per week helped direct development and ensured satisfaction. Feedback gathered during playtest sessions verified that the scene had a serene atmosphere. The clients expressed their content about the concepts, designs, and results. Screenshots of the final build in VR are shown in Figures 11 and 12. The white and red poles are teleport indicators.

Figure 11 and 12: Final results in VR

VIII.2. What climate, season, and colors are generally considered relaxing?

VIII.2.A. Research

Thales wanted to have a natural scene in contrast to one of their previous VR projects, which took place in an engine room. According to the client, users could not immerse themselves in the experience, and the environment felt too artificial and uncomfortable overall. Desk research was conducted on the topic of nature to validate these claims. The related findings are summarized in **Appendix O**.

VIII.2.B. Results

What climate, season, and colors are generally considered relaxing?

Spring or summer was set as the target season because it aligned the most with the findings. A bright, lively environment was created. Green and blue dominated the scene as these colors were shown to have the most relaxing values. Background buildings were excluded to avoid cultural associations. A temperate climate feel was achieved by using indistinctive deciduous trees instead of conifers and palms. Extremes such as tropical or tundra were avoided. Post effects were necessary to finetune the colors and contrasts. Effects such as color correction, bloom, and split toning were utilized. Depth of field was also added for more realism by simulating the blurriness of distant objects. Based on player feedback, the scene had an overall relaxing atmosphere. The addition of a few nature sounds greatly boosted the immersion. The clients expressed their content about the research and how it was implemented.

VIII.3. How to create optimized tree assets?

VIII.3.A. Research and Development

The student's previous experience with manual tree creation pushed him to search for alternatives. Manually creating trees includes modeling, sculpting, and texturing the tree's trunk and branches and creating twig and leaf textures. These are time-consuming tasks, so the student looked for an alternative. Related research findings are summarized in **Appendix N**.

VIII.3.B. Results

Creating realistic-looking and low-poly trees for VR was considerably challenging. Finding a balance between triangle counts and aesthetics proved to be laborious. The results could have been better if the student had had more experience with foliage creation, but the trees were satisfactory based on feedback.
Making use of the LOD settings was key to optimize the trees. The adult tree's triangle counts from highest to lowest LOD were 6010, 2420, and 855, while the sapling's were 2576, 1403, and 823 triangles. Their effect on performance was reasonable, though it must be mentioned that the final scene had considerably fewer trees than the previous iterations. Wind effects exported from SpeedTree worked well in URP. Overall, SpeedTree proved to be a flexible and reliable tool for 3D tree creation. The final trees are shown in Figure 13.

Figure 13: The final tree models

VIII.4. What is the difference between the Hight-Definition Rendering Pipeline and the Universal Rendering Pipeline in Unity?

VIII.4.A. Research

When creating a new project in Unity in 2021, the user may choose between two predefined render setups: the High-Definition Rendering Pipeline (HDRP) and the Universal Rendering Pipeline (URP). As the name suggests, URP is meant for a wider array of platforms, including mobile. URP uses forward rendering. HDRP, on the other hand, is not meant for mobile devices and focuses on rendering scenes as close to realism as possible on high-end hardware (Unity, 2021). Lighting in HDRP is based on lux value instead of intensity. Some features, such as deferred rendering, decals, real-time global illumination, volumetric lighting, post-processing ambient occlusion and screen-space reflection, and raytracing, are only available in HDRP as of June 2021. These are powerful but demanding tools for reaching photorealism.

Youtuber and Unity expert Brackeys (2020) advises against using HDRP if the team is smaller than five people because more experience and resources are required compared to URP. He also claims that VR with HDRP can get highly performance-heavy. HDRP used different shader setups and slightly different texture formats. This meant that materials that worked in URP had to be manually reset for every object in HDRP. Wind simulation on the trees also did not work in HDRP because the SpeedTree company did not update the shader since HDRP came out.
Performance of the early VR scene was tested in both rendering pipelines. HDRP had 27 to 40 FPS, URP had a slightly smoother 37 to 50 FPS. These results in VR showed that the URP scene performed better, though both readings were lower than expected.

VIII.4.B. Results

The issues mentioned above and more experience in URP made the team gravitate towards URP. Tests were done exclusively on URP past week nine.
The team ended up using URP because it performed better than HDRP. Furthermore, neither of the students worked with Unity's HDRP before, and the pipeline was considerably different, and it is not recommended for small teams, especially without previous experience in it (Brackeys, 2020).
Thus, quality in one render setup was prioritized over testing both. Thales accepted these terms as only one scene was supposed to be included in the final deliverables.

VIII.5. How to optimize a scene for VR? What exact methods can be used?

VIII.5.A. Research and Development

Since the target platform was VR, it was imperative to have stable frame rates of at least 90. Otherwise, the players may have experienced discomfort and nausea (Oculus, n.d.). According to Kevin Connolly (2020), each lens in a VR headset can be considered an individual display for each eye. The two lenses display slightly different images to create and maintain a sense of depth. This method called stereo rendering challenges the processor's computing power and the graphics card when rendering.
NVIDIA graphics programmer Nathan Reed (2015) states that stereo rendering may use up to twice the processing power than a single monitor with identical resolution. While some calculations, such as physics simulations, are not executed twice with stereo-rendering, the visual rendering is still done separately for each eye.
Furthermore, VR lenses are built with high-resolution displays and refresh rates to get clearer and less nauseating images. However, this means more demand on the hardware. The refresh rate of most headsets is 90 Hertz. This number defines the maximum number of frames per second displayed. Based on their official manual, Oculus (n.d.) suggests that developers aim for frame rates equal to or above the display's refresh rate to avoid discomfort in VR.
These findings showed that the lens displays of VR headsets objectively raise the system requirements compared to computer monitors, and they can take up to twice the processing power to render the same scene.

Many methods can increase performance even outside of the pipeline. Some basic tips for increased power, as suggested by LaptopNinja (2020), are: upgrading the operating system to Windows 10, updating graphics drivers, cleaning up HDD or SSD space, closing unused background processes using the task manager, and setting the power plan to 'high performance' in Windows in the Power Options menu. The author's advanced tips include defragmenting the HDD or SSD, enabling overclocking, changing the PC's virtual memory page file, and making registry changes. It must be noted that only experienced users should attempt these advanced tips. Making backups beforehand is strongly recommended.
The team used their previous experience in scene optimization. Models and textures were optimized according to their context in the scene. Background objects had lower triangle counts and texture resolutions. Many details were baked into texture data instead of using high-poly models. Some models were combined into one mesh to lower the number of draw calls. Some UV shells overlapped to save space on the texture map. All non-animated objects were set to static.

Furthermore, Unity's official manual on optimization proved as a great source. According to Unity's tutorial (2020), there are five essential methods to optimize a VR scene. These are light baking, occlusion culling, static batching, adjusting quality settings, and using single-pass rendering instead of multi-pass.
Light baking means that the lights on the scene are precomputed; thus, the system needs less time to render the scene, putting less stress on the system.
Culling is the method of only rendering objects that are in view. As opposed to frustum culling, occlusion culling not only hides the objects outside of the camera's view frustum but also any object that another model entirely obscures. This data needs to be calculated once all static meshes have their final position on the scene. It can save a great amount of computing power. (Unity, 2020)
Quality settings are built-in options to optimize a scene for either performance or quality. It was expected to use the highest available settings to achieve photorealism.

The rendering method for VR was significant. There are three of these available in Unity: multi-pass, single-pass, and single-pass instanced. To summarize, single-pass instanced is the best option for prioritizing performance, but it is not available on all devices. For more details, see **Appendix E.**
One more crucial factor is called draw calls. Unity (2020) claims that "*To draw a GameObject on the screen, the engine has to issue a draw call to the graphics API. Draw calls are often resource-intensive, with the graphics API doing significant work for each one, causing performance overhead on the CPU side.* ". Consequently, draw calls and triangle count should be kept as low as possible to reduce the computational load on the CPU and GPU. As mentioned above, this can be achieved by manually combining models into a single object. However, there is another way to get the same results without combining meshes, which is called 'draw call batching', and it is the process of Unity computing objects with the same material as one, thus reducing draw calls.
There are two techniques: dynamic batching and static batching. Dynamic batching is automatically applied to objects sharing the same material, while static batching also requires these objects to be set as static. Static batching is more efficient on the processor but uses more memory power. For overall performance, static batching is usually preferred.
Unity also suggests adjusting the lightmap size for increased lighting accuracy, which results in increased bake times and build file sizes, but it also increases the lighting's precision and quality. This value may be reduced if performance demands it.

The student spent many weeks on optimization. Key aspects of this are shown in the following paragraphs.

Reducing Terrain Density

The scene consisted of six terrains connected at the edges. Tests showed that the terrains had too many triangles. Unity edits terrains based on heightmaps which are then translated into 3D objects. The first attempt to make the terrains have fewer triangles was to reduce the heightmap resolutions - this moderately boosted performance.
The student aimed to display more details while also further lowering the triangle count. A tool made by Mike Hergaarden (2014) was used to export the center terrain as an OBJ file. The exported terrain was checked in Maya. Maya showed that it was 129706 triangles, which was still too many. Since complex details could not be sculpted on the Unity terrain, it was imported to ZBrush. A standard high-poly to low-poly pipeline was applied to sculp more details on the terrain, especially on the two rocky cliffs hugging the lagoon. Low-poly and high-poly versions of the model were exported. The sculpted terrain was UV mapped in Maya. Many unnecessary faces were deleted. The final model was 45294 triangles. It was split into five texture sets so that texture resolution would be optimal. It was then textured in Substance Painter, which provided considerably more control than painting textures on the terrain in Unity. The high-poly model was baked onto the low-poly one.

The original Unity terrain was still in use because it contained the location information of all trees and details objects. The sculpted terrain was placed in the same position as the original. Rendering of the original was disabled except for the trees and details. However, since the sculpted terrain became slightly different than the original, some trees and details did not touch the ground. It would have taken too much time to manually place the trees and details, so the student had to find a solution. This issue was solved by converting the sculped terrain back into a Unity terrain with a tool by LMHPOLY (2017). This way, the final scene also had two overlapping terrains objects: the sculpted model and the Unity terrain converted from this sculpted model. Once again, the Unity terrain itself was not rendered but merely used to paint foliage because its triangle count was too high. Trees and details had to be repainted, but this way, they did not float above the ground anymore.

Later, Annemieke suggested replacing the background mountains with planes. The student did this by taking screenshots of the mountains in Unity's isometric view, then cut different layers from them in Photoshop. These layers were applied as textures to planes in Unity. The planes were aligned with the view. The results were impressive. Because the playable area was minimal, the difference between the 3D and the 2D mountains was unnoticeable, as seen in Figure 14. This method saved thousands of triangles and a few hundred draw calls.

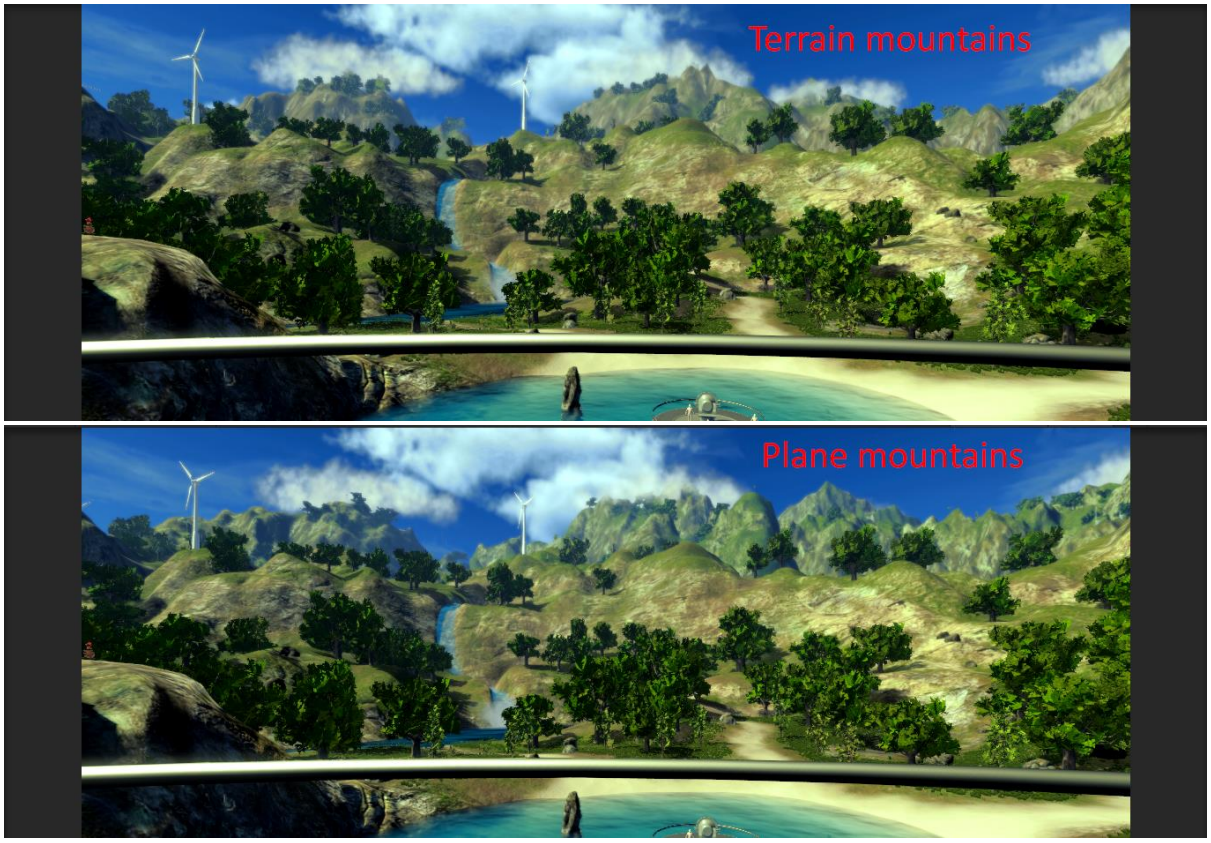

Figure 14: Comparing the mountains as terrains (above) and as planes (below)

Reducing Draw Calls

After reducing the terrain's triangle density, performance was still not satisfying, so the student looked to draw calls instead of the triangle count. The scene indeed had too many draw calls, around 2000. This number was unexpectedly high. The student investigated what may have caused it.
At this point, Unity's standard trees were still on the scene as placeholders. A test on a fresh scene confirmed that these trees caused a few hundred unwanted draw calls, which the custom trees did not, as shown in **Appendix M**.
Replacing the standard trees helped, but something was still causing too many draw calls. After disabling every object on the scene one by one, the student found that the grass asset caused this. Every single piece of grass plane added one draw call. Deleting the grass boosted performance significantly, and draw calls had been reduced to around 500. However, an alternative solution was necessary to render grass because the scene looked too flat without any.
Unfortunately, no clear solution was found before the submission of this report. Annemieke created a grass patch model consisting of multiple planes and added them to the terrain with reduced detail resolution. These worked better, but unfortunately, the grass still caused an undesirable number of draw calls. The compromise was to have fewer grass assets on the scene.

Issues with Occlusion Culling

Many attempts were made to compute better occlusion culling, including setting up occlusion areas, adjusting the smallest occluder, and decreasing the backface threshold. OC was baked at least fourteen times without success. Examples of the incorrect OC are shown below. In Figure 15, the camera was looking at a wall, but objects behind it still rendered. In Figure 16, the backsides of mountains were unaffected by OC, which was why the background mountains were turned into planes.
None of the mentioned above methods worked as expected, so the student researched the issue. A blog was found that suggested that using occlusion culling for large exteriors is not ideal because calculating it may put more stress on the CPU in runtime than it relieves. (Bonet, 2021). Based on this, the team decided not to use occlusion culling.

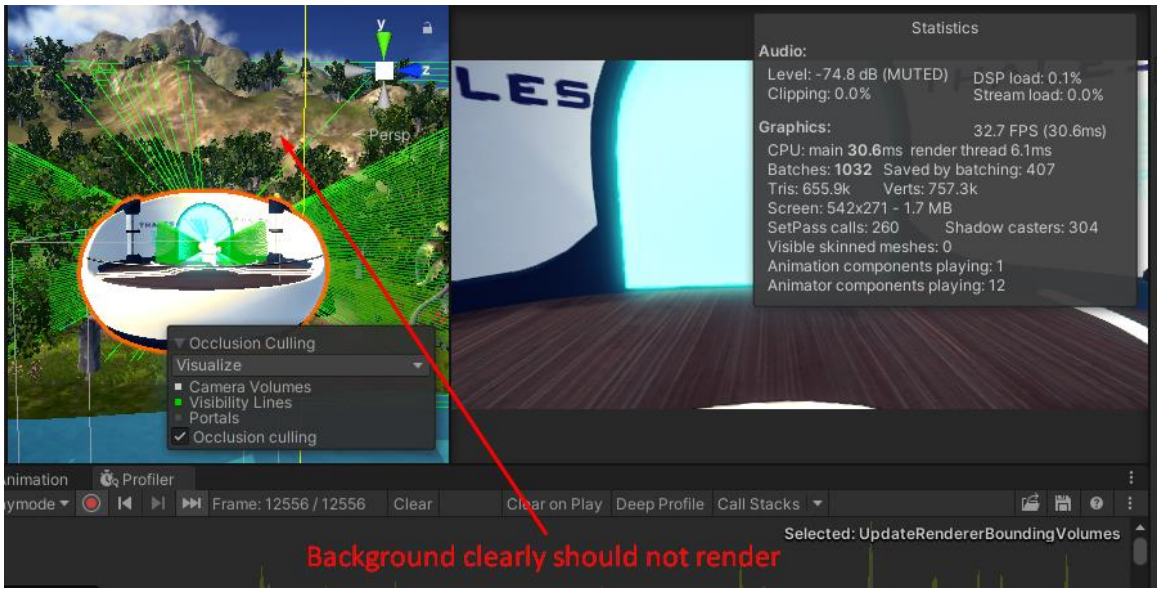

Figure 15: Example 1 of the incorrect occlusion culling

Figure 16: Example 2 of the incorrect occlusion culling

VIII.5.B. Results

How to optimize a scene for VR? What exact methods can be used?

All non-animated objects were set to static, and static batching was turned on. Only the highest quality setting was enabled, the lower levels were deleted. Single-pass instanced rendering was used for VR rendering because it was the most performant. No occlusion culling was used because it turned out to be unreliable for large exterior scenes. (Bonet, 2021). The scene ended up not having any baked lights. The main directional light was real-time so that the shadows of the trees and other moving shadows would render. Despite this, light baking had to be processed to display Crest's water effects correctly. Post effects had little to no effect on performance. An earlier test showed that when disabling all post effects, only two FPS were gained. Grass could not be fully optimized before the deadline of this report. The two overlapping terrains technique seemed to be unconventional, but it worked well nonetheless.

The maximum number of triangles displayed when looking at the scene from the most extreme angle in the final Unity scene was about four million, while the lowest was around 700 thousand. Draw calls ranged between 800 and 1600. The scene had between 45 and 70 FPS. It was assumed that the build version of the scene would work better than in the Unity editor, which turned out to be true. The final build had stable 90 FPS; thus, the target frame rate was achieved. The target HTC Vive Pro headset has a refresh rate of 90 Hz, meaning it cannot display more FPS than 90, which does not conclusively suggest that radar models mentioned in **chapter IV.** will not cause performance issues. Further tests with the inclusion of these highly detailed radar models would be required.

VIII.6. How are similar VR simulations' graphics and performance on the target hardware?

VIII.6.A. Research

Thales claimed to own a computer with one of the strongest graphics cards in 2021, an Nvidia GeForce RTX 3090, an Intel-I9 processor, and 32 gigabytes of RAM. However, since the tests were done on Saxion's computer, that one's specifications were considered when comparing frames per second data. The XR Lab's hardware consisted of an Nvidia GTX 2080 Ti graphics card, an Intel I7-7820X processor, and 64 gigabytes of RAM.

The system requirements of recent VR games with realistic graphics were taken as references. The referenced games are Half-Life: Alyx (2020), Defector (2019), and Star Wars: Squadrons VR (2020). Based on the system requirements, a table was made. The table shows that both PCs not only meet the system requirements but surpass them by a considerable margin.

	Half-Life Alyx Requirements (Minimum)	Defector Requirements (Minimum)	Star Wars: Squadrons VR Requirements	XR Lab Hardware	Thales's Hardware
Operating System	Windows 10 64-bit	Windows 10 64-bit	Windows 10 64-bit	Windows 10 64-bit	Windows 10 64-bit
Processor (Intel)	Core i5-7500	Core i7-6700K	Core i7-7700	Core i7-7820X	Core-i9
Memory	12 GB RAM	8 GB RAM	16 GB RAM	64 GB RAM	32 GB RAM
Graphics Card (Nvidia)	6 GB GeForce GTX 1060	3 GB GeForce GTX 1060	4 GB GeForce GTX 1070	12 GB GeForce RTX 2080 Ti	24 GB GeForce RTX 3090

Additionally, a specific performance test of Half-Life: Alyx played on a 2080 Ti was taken to prove the strength of the hardware further, shown in **Appendix F.** According to Poppin (2020), the RTX 2080 Ti handles the Half-Life: Alyx on the highest 'Ultra' settings with ease.
Based on these findings, it was assumed that Thales's hardware should have been more than capable of handling the scene the team was going to create, even with some 4K textures, especially regarding the fact that Thales has an even more powerful PC than Saxion, which turned out to be an overly optimistic assumption.

VIII.6.B. Results

As mentioned in the chapters above, optimization was not a simple procedure. The games compared in this chapter were created by teams of hundreds of professionals. It was incorrect to rely on the hardware for optimization and showed that a strong PC does not equate to good performance. Optimizing games takes tremendous amounts of work and testing, especially when developed for VR.

IX. Discussion

IX.1. Interpretations

The results indicate that creating a well-optimized, photorealistic VR learning environment in Unity requires constant contact with the client, relevant research data, previous experience in 3D development, performance tests, and playtest sessions. Overall, these methods worked well and could be applied again for a similar project.
The amount of effort it took to optimize a scene for VR was unexpected. Wrong assumptions were made based on the high-end PC owned by Thales. Developers should not rely merely on hardware for optimization. A strong system does not innately mean good performance. Fortunately, research was relevant and valuable; thus, the target FPS of 90 was reached in the final build.
Surprisingly, post effects had little to no effect on performance. The student was also amazed about how well the background mountains worked as planes.
As expected, player feedback aligned with findings of nature, trees, and colors. As the student anticipated, SpeedTree proved to be a flexible and convenient tool.

IX.2. Limitations

Unfortunately, it was beyond the scope of this study to conduct playtests with the target audience because of the Covid19 restrictions. Playtests should have been more frequent, nonetheless. This limitation makes the results about the atmosphere of the final scene somewhat unreliable, despite the client's satisfaction.

This project could not have been possible without previous experience in specific digital art tools. However, lack of experience in particular fields slowed down production. None of the students had enough experience creating large exterior scenes and foliage, which made the project challenging. The student learned that it might be worth buying tools or assets to save time in some cases.
Reaching the visual fidelity of photorealism was too ambitious for the size and expertise of the team. Converting the URP project into HDRP could not fit in the project's scope because of time limitations. Even though Thales instructed the team to only focus on visuals and performance, the lack of a programmer was disadvantageous. Many hours of technical research and implementation could have been saved with a programmer.
The target HTC Vive Pro headset has a refresh rate of 90 Hz, meaning it cannot display more FPS than 90, which does not conclusively suggest that radar models mentioned in chapter IV. will not cause performance issues. Further performance tests with the inclusion of the radar models are required.

Excess draw calls caused by grass could not be fully optimized before the deadline of this report. Since VR is a relatively new tool in education, not many relevant studies could be found. Unexpected issues with Git forced the team to use Google Drive as a version control tool.

IX.3. Potential

This thesis's future potentials include using the same methods and pipelines in similar projects and expanding the VR learning environment concept. Song et al. (2020) stated that VR training is an effective way of learning real-life operations. VR and Cloud XR could become mainstream learning environments soon. Streaming experiences worldwide and not transporting specific radars to another country could save tremendous amounts of money and time. Based on these statements, it can be concluded that Thales's vision is plausible.

X. Conclusion

In this chapter, all the major research findings and development results are summarized, then the main question is answered.

This thesis aimed to answer the question '**How to create a well-optimized, photorealistic Virtual Reality learning environment in Unity to instruct users about radar maintenance, in which they feel relaxed and comfortable?**' for Thales Netherlands. Thales wanted a new VR learning environment for radar maintenance. The main problem was that a previous VR learning environment felt too artificial and discomforting and had performance issues. The proposed solution was to research and create a VR prototype of an environment that is photorealistic and relaxing, runs well on the target platform, and fits Thales's company profile.

Weekly meetings with Thales provided a constant flow of feedback and enabled iterative development. The heavy involvement of the clients was crucial to direct development and to ensure satisfaction. Technical performance tests were done to inspect performance, playtesting and interviews were done for feedback. Playtest sessions were held at Saxion's XR Lab with volunteer students. Unfortunately, time and Covid19 constraints pushed further testing or interviewing the target audience beyond the project's scope.

Several studies confirmed that natural environments have positive effects on humans. People have an innate preference toward natural scenery over urban. The color green in nature reduces stress, even if it is merely a visual representation, such as paintings or digital art (Beute and Kort, 2013) (Kweon et al., 2008). A lagoon surrounded by green scenery with an observatory and a maintenance platform fit the perquisites set by Thales, which were to create a VR environment in Unity relevant for radar maintenance in which people feel comfortable and has Thales's look and feel.
A bright environment was created accordingly. The scene's season was set to spring or summer because existing research showed general preference towards bright, green nature sceneries (Beute and Kort, 2013) (Hardcastle, 2018) (Kaufman and Lohr, 2008) (Lee et al., 2015) (Park et al., 2010) (Yang and Brown, 1992). Background buildings were excluded to avoid cultural associations. A temperate climate feel was achieved by using indistinctive deciduous trees instead of conifers and palms. Extremes such as tropical or tundra were avoided. Feedback gathered during playtest sessions aligned with the findings, and testers reported that the scene had an overall relaxing atmosphere. Overall, photorealism proved to be too ambitious for the size of the team. Nonetheless, a pleasant scene was created that satisfied the clients.

URP was chosen over HDRP because it performed better. Furthermore, neither of the students worked with Unity's HDRP before, the pipeline was considerably different, and it is not recommended for small teams, especially without previous experience in it (Brackeys, 2020). Converting the URP project into HDRP could not fit in the project's scope because of time limitations.
Since the target platform was VR, it was essential to have stable frame rates of at least 90. Otherwise, the players may have experienced discomfort and nausea (Oculus, n.d.). The key aspects of optimization were the number of triangles, draw calls, and render settings. Single-pass instanced rendering was used for VR rendering because it was the most performant. All non-animated objects were set to static, and static batching was turned on. No occlusion culling was used because it turned out to be unreliable for large exterior scenes. The only directional light was real-time so that the shadows of the trees and other moving shadows of the scene would render. Grass could not be fully optimized before the deadline of this report. Background mountains were replaced with 2D planes. Post effects had little to no effect on performance.
Stable 90 FPS was reached in the final build of the scene; thus, the target frame rate was achieved. Optimizing the scene for VR took great amounts of work and testing. In general, developers should not rely on hardware for optimization. A strong PC does not equate to good performance. The clients were satisfied with the performance.

How to create a well-optimized, photorealistic Virtual Reality learning environment in Unity to instruct users about radar maintenance, in which they feel relaxed and comfortable?

Contact the client as soon as possible. Spend much time understanding the wishes and needs of the client. Collect and analyze requirements set by the client, then define the problems and the target audience. Brainstorm about possible themes in the confines of the requirements, document, and share ideas with teammates. Get inspiration from media products with similar themes. Collect images and create moodboards based on the ideas. Create an early graybox to convey ideas if necessary. Present the ideas to the client, then adjust and combine elements of the concepts according to their preferences. Create concept art. Repeat gathering feedback from the client, do not start production before determining an ideal environment which they approve. When the client is satisfied with the concept, define the scope of the project. Describe the must-have aspects of the scene, create an asset list, and order tasks based on the 'MoSCow' prioritization system. Describe known constraints, assumptions, and what features will be excluded.
Research and document relevant studies, including scientifically relaxing environments, optimization methods for VR, and the Unity rendering pipelines. Use a suitable research method and evaluate the relevance of the findings. Define the most appropriate tools to use. Apply the findings to the concepts.

Start production of the prototype after having the preliminary research and problem definition. Possess, or acquire experience in 3D modeling, foliage creation, texturing, and using Unity. Create a graybox of the concept in 3D. Import the placeholder assets into Unity, create an 'alpha' version of the scene. If applicable, replace some placeholders with Unity's tools or free relevant plugins and third-party assets. Consider investing in certain priced art assets or technical scripts if the team lacks skills in those fields to save time.
Compare the alpha scene in URP and HDRP, run early visual and performance tests in VR. Define the most fitting rendering pipeline based on the test and research results. Have VR playtest sessions, ideally with the target audience, as soon as possible. Gather feedback about the visuals and the atmosphere of the scene in VR. Create the necessary assets and implement them into the Unity scene. Add post effects to control colors and create higher visual fidelity. Develop iteratively. Send updates and have meetings frequently with the client to ensure the project heads in a direction they envisioned.

Keep track of the render settings, triangle counts, and draw calls, optimize the scene for VR. Create a final Unity build when the scene is deemed finished. Test the performance of the final scene in the Unity editor as well as the build. Organize a final VR playtest session, ideally with the target audience. Observe and interview testers about the serenity and visual realism of the scene. Evaluate the results within the team. Share the final build with the clients and assess their feedback. Conclude whether the client is satisfied with the product.

XI. Recommendations

Actions to further improve the product and continue development are suggested in this chapter.

If Thales wishes to continue developing this product, it is strongly advised to hire a team equivalent or larger than this period's and include at least one programmer with VR experience.
More testing is necessary, ideally with the target audience. Gathering relevant and reliable feedback is vital, even though it is difficult because of the Covid19 restrictions. Should Thales decide to develop this project further, it is recommended to invest even more time into testing. Testing the final scene with the radar models is vital to define the success of optimization, though Thales took full responsibility for optimizing them if necessary.
Better overall VR controls are highly recommended. The team suggests checking if SteamVR is supported again in the latest version of Unity in a few months because it provides the most intuitive VR controls. Alternatively, it could be worth investing in a customized virtual controller setup.
Creating a better teleport system is necessary. A giant collision box could be placed on the platform and the observatory. When pointing and releasing the teleport button, it could place the player on the other building in a fixed position. This effect should work both ways. Better teleportation includes the portal circles between the top of the platform and the server room. It is recommended to make the controller snap to the teleport circle. When the teleport button is released, the player would appear on top of the selected portal circle, then shortly after would appear on the opposite one accompanied by a fade-out effect and sound feedback.
Further improvements could be made to increase performance, especially regarding the grass and draw calls. A solution might be found and implemented until the end of the graduation period, but not before this report's deadline.
The underwater scene was partially implemented. Crest's underwater shader was already in use, but unfortunately, this effect triggers inside the server room. Fixing this issue is a must for completing the server room. Further suggested improvements are adding assets such as seaweed, fishes, different rocks, corals, and seashells. The underwater scene could be a different scene, introducing loading times, but it would be preferable for performance.
There could be an exit feature related to the portal in the observatory. The controller could snap on the portal when aimed at while holding the teleport button. Then, an exit confirmation screen could pop up. This way, the escape function would be diegetic.
The tutorial was not implemented. A script of events shown in **Appendix K**, written by the student, could be used to add it later.
One larger platform could be implemented to fit the biggest radars, such as the SMART-L. The corresponding platform and radar could be selected and loaded before entering the level in a menu.
Despite the team's findings, it might be worth considering upgrading the scene to HDRP in the future, resulting in a much more accurate visual simulation of real-life while also raising system requirements. A larger team with more experience in HDRP and optimization would be preferable.
Moving clouds and bird and insect particles would further increase immersion and realism, so it is recommended to include them.
Whiteboards could be added near the radars for the users to take notes about the maintenance procedure. This feature can be seen in the first level of Half-Life: Alyx (2020).

Figures

1. *Figure 1: The logo of Thales*
   Thales Netherlands (2021). [Company Logo] Retrieved March 6, 2021, from https://www.thalescybersecurity.nl/

2. *Figure 2: The lagoon picture that inspired the student*
   The World Travel Guy. (2018, March 16). Flying The World - Iceland, Indonesia, Hawaii, Maine & More 4K Drone Footage [Screenshot of the video at 3:00]. YouTube. Retrieved March 12, 2021, from https://www.youtube.com/watch?v=7SmBJamQuPw&ab_channel=TheWorldTravelGuy

3. *Figure 9: The gyroscope inspiration*
   Respawn Entertainment. (2016, November 27). Titanfall2 fold weapon by jungpark on DeviantArt. [Concept Art] Retrieved April 03, 2021, from https://www.deviantart.com/jungpark/art/Titanfall2-Fold-weapon-648025354

4. *Figure 17: The Thales moodboard*
   De Normaalste Zaak (2016). [Image] Retrieved March 6, 2021, from https://www.denormaalstezaak.nl/thales-als-banen-geschikt-wilt-maken-moet-kijken-echt-belangrijk-is
   MTD Landschapsarchitecten (n.d.). High Tech Systems Park Thales [Image] Retrieved March 6, 2021, from https://www.mtdls.nl/nl/projecten/detail/q/idproject/257/titel/high-tech-systems-park-thales
   Stern Architects & Sra-Architectes (n.d.). Thales HQ Carpe Diem tower. [Image] Retrieved March 6, 2021, from https://www.adsadvance.co.uk/thales-selects-sopra-steria-to-support-its-sap-transformation-programme.html
   Thales Netherlands (2021). [Image] Retrieved March 6, 2021, from https://www.glassdoor.nl/Fotos/Thales-Kantoorfotos-IMG1127405.htm
   Thales Netherlands (2021). [Image] Retrieved March 6, 2021, from https://www.thalesgroup.com/en/career/working-us
   Thales Netherlands (2021). [Company Logo] Retrieved March 6, 2021, from https://www.thalescybersecurity.nl/
   Tubantia (2006). Thales Hengelo. [Image] Retrieved March 6, 2021, from https://commons.wikimedia.org/wiki/File:Thales_Hengelo.jpg
   Zenber (2019). Thales shared facility centre. [Image] Retrieved March 6, 2021, from https://zenber.nl/portfolio_page/thales-shared-facility-centre/

5. *Figure 21: The RTX 2080 Ti's performance chart*
   Poppin, M. (2020). [RTX 2080 Ti's performance chart]. Retrieved March 16, 2021, from https://babeltechreviews.com/half-life-alyx-performance-iq-review-across-13-amd-nvidia-cards/

6. *Figure 22: The Design Thinking Model*
   Interaction-Design. (2020). [Graph of the Design Thinking Model]. Retrieved February 19, 2021, from https://www.interaction-design.org/literature/article/5-stages-in-the-design-thinking-process.

7. *Figure 23: AEQUOREA by Vincent Callebaut, image 1*
8. *Figure 24: AEQUOREA by Vincent Callebaut, image 2*
   Callebaut, V. (2015). AEQUOREA [Image] Retrieved March 04, 2021, from https://vincent.callebaut.org/object/151223_aequorea/aequorea/projects

9. *Figure 25: PHYSALIA by Vincent Callebaut*
   Callebaut, V. (2010). PHYSALIA [Image] Retrieved March 04, 2021, from https://vincent.callebaut.org/object/100104_physalia/physalia/projects

10. *Figure 29: The controls shown on the handles in The Lab*
    Let's STFU and Play. (2017, Nov 17). The Lab - A collection of VR experiments by Valve (full playthrough, no commentary). [Screenshot of the video at 1:29] Retrieved March 15, 2021, from https://www.youtube.com/watch?v=QYrYJHktrf0&ab_channel=Let%27sSTFUandPlay

References

1. Agile Alliance. (2021, February 23). What is scrum? Retrieved May 12, 2021, from https://www.agilealliance.org/glossary/scrum/

2. Avionx. (2018, August 9). Skybox series Free: 2D Sky: Unity asset store. Retrieved June 01, 2021, from https://assetstore.unity.com/packages/2d/textures-materials/sky/skybox-series-free-103633

3. Beute, F., & Kort, Y. d. (2013, August 7). Let the sun shine! Measuring explicit and implicit preference for environments differing in naturalness, weather type and brightness. Retrieved March 10, 2021, from https://www.sciencedirect.com/science/article/pii/S0272494413000583

4. Bonet, R. T. (2021, February 2). How to use occlusion culling in Unity - the sneaky way. Retrieved June 3, 2021, from https://thegamedev.guru/unity-performance/occlusion-culling-tutorial/

5. Brackeys. (2020, March 01). HDRP vs. Urp - Which Unity template should you choose? Retrieved May 03, 2021, from https://www.youtube.com/watch?v=5MuA92xUJCA&ab_channel=Brackeys

6. Callebaut, D. (n.d.). VINCENT CALLEBAUT ARCHITECTURES PARIS. Retrieved March 02, 2021, from https://vincent.callebaut.org/

7. Cameron, J. (Director). (2009) *Avatar* [Film]. 20th Century Fox.

8. Clay, R. (2001, April). Green is good for you. Retrieved March 02, 2021, from https://www.apa.org/monitor/apr01/greengood

9. Connolly, K. (2020, October 2). How to Build a PC for Virtual Reality: Oculus Rift, HTC Vive, and VR Gaming. Retrieved February 17, 2021, from https://www.logicalincrements.com/articles/vrguide

10. Detroit: Become Human [Computer software]. (2018) Quantic Dream.

11. EA. (2021). STAR WARS™: SQUADRONS - PC System Requirements. Retrieved March 16, 2021, from https://www.ea.com/games/starwars/squadrons/overview/pc-system-requirements

12. Gerstenberg, T., & Hofmann, M. (2015, December 17). Perception and preference of trees: A psychological contribution to tree species selection in urban areas. Retrieved March 23, 2021, from https://www.sciencedirect.com/science/article/pii/S161886671500182X#bib0155

13. Ghost of Tsushima [Computer software]. (2020) Sony Interactive Entertainment

14. Half-life: Alyx [Computer software]. (2020) Steam

15. Hardcastle, A. (2018, August 24). Do we prefer some types of nature over others? Retrieved February 24, 2021, from https://raog.ca/2018/08/24/do-we-prefer-some-types-of-nature-over-others/

16. Hergaarden, M. (2014) TerrainObjExporter [C# script] Retrieved March 13, 2021, from https://wiki.unity3d.com/index.php/TerrainObjExporter

17. James, P. (2015, June 30). NVIDIA takes the lid OFF 'Gameworks VR' - technical deep dive and Community Q&A. Retrieved February 17, 2021, from https://www.roadtovr.com/nvidia-takes-the-lid-off-gameworks-vr-technical-deep-dive-and-community-qa/

18. Johansson, A. (2018, April 13). Why natural scenery improves your mood and makes you more productive. Retrieved March 02, 2021, from https://www.nbcnews.com/better/health/why-natural-scenery-improves-your-mood-makes-you-more-productive-ncna860806

19. Kaufman, A. J., & Lohr, V. I. (2008). Does it matter what color tree you plant? Retrieved March 03, 2021, from https://www.actahort.org/books/790/790_25.htm

20. Kazim, I. (2020, June 16). Free 3D rock - turbosquid 1577378. Retrieved June 01, 2021, from https://www.turbosquid.com/3d-models/3d-rock-1577378

21. Kweon, B., Ulrich, R. S., Walker, V. D., & Tassinary, L. G. (2008, May). Anger and Stress The Role of Landscape Posters in an Office Setting. Retrieved March 02, 2021, from https://www.researchgate.net/publication/258132363_Anger_and_Stress_The_Role_of_Landscape_Posters_in_an_Office_Setting

22. LaptopNinja Editorial Team. (2020, December 08). 3 simple ways to optimize your laptop for VR Gaming. Retrieved February 17, 2021, from https://laptop.ninja/3-simple-ways-to-optimize-your-laptop-for-vr-gaming/

23. Lee, K., Williams, K., Sargent, L., Williams, N., & Johnson, K. (2015, May 02). 40-Second green roof views sustain attention: The role of micro-breaks in attention restoration. Retrieved March 02, 2021, from https://www.sciencedirect.com/science/article/abs/pii/S0272494415000328

24. LMHPOLY (October 25, 2017) CONVERT MESH TO UNITY TERRAIN [C# script] Retrieved June 4, 2021, from https://www.lmhpoly.com/tutorials/convert-mesh-to-unity-terrain

25. Lohr, V., & Pearson-Mims, C. (2006, September 1). Responses to Scenes with Spreading, Rounded, and Conical Tree Forms. Retrieved March 22, 2021, from https://journals.sagepub.com/doi/abs/10.1177/0013916506287355?casa_token=XOabNrBmWWIAAAAA:gapfOiazik_hHhxsZs2yld4jMxa7ilRXfmBrmX0hyMpcBOhfUkaNsDfAp_wVbeKLNPiTfdbLGD21PA

26. Namabenama. (2018, June 26). Free Mountain ROCK pbr 8k 3D model - TURBOSQUID 1300107. Retrieved June 01, 2021, from https://www.turbosquid.com/3d-models/mountain-rock-pbr-8k-3d-model-1300107

27. Nelson, T., Johnson, T., Strong, M., & Rudakewich, G. (2002, May 25). perception of tree canopy. Retrieved March 18, 2021, from https://www.sciencedirect.com/science/article/pii/S0272494401902233

28. Oculus. (2019, July 11). Defector on Oculus Rift. Retrieved March 16, 2021, from https://www.oculus.com/experiences/rift/1524154440988718/

29. Oculus. (n.d.). Documentation. Retrieved February 17, 2021, from https://developer.oculus.com/documentation/

30. Park, B. J., Tsunetsugu, Y., Miyazaki, Y., Kagawa, T., & Kasetani, T. (2010, January). The physiological effects OF shinrin-yoku (taking in the FOREST atmosphere or FOREST Bathing): Evidence from field experiments in 24 forests across Japan. Retrieved March 02, 2021, from https://pubmed.ncbi.nlm.nih.gov/19568835/

31. Petry, C. (2014, June 28). NormalMap-Online. Retrieved May 31, 2021, from https://cpetry.github.io/NormalMap-Online/

32. Poppin, M. (2020, April 3). Half-Life: Alyx performance and IQ review with 13 AMD & Nvidia cards. Retrieved March 16, 2021, from https://babeltechreviews.com/half-life-alyx-performance-iq-review-across-13-amd-nvidia-cards/

33. Procedural Worlds. (2021). Gaia 2021 - terrain & Scene Generator: TERRAIN: Unity asset store. Retrieved June 02, 2021, from https://assetstore.unity.com/packages/tools/terrain/gaia-2021-terrain-scene-generator-193509

34. ProductPlan. (2021, May 14). What is MoSCoW Prioritization?: Overview of the MoSCoW Method. Retrieved May 12, 2021, from https://www.productplan.com/glossary/moscow-prioritization/

35. Resident Evil 8: Village [Computer software]. (2021) Capcom.

36. Saatci, G. (2021, February 23). Free 3D Stone rock - Turbosquid 1695868. Retrieved June 07, 2021, from https://www.turbosquid.com/3d-models/3d-stone-rock-1695868

37. Sommer, R., & Summit, J. (1996). Cross-National Rankings of Tree Shape. Retrieved March 22, 2021, from https://www.tandfonline.com/doi/abs/10.1207/s15326969eco0804_2?journalCode=heco20

38. Steam. (2016, April 5). The Lab. [Computer software]. Retrieved March 15, 2021, from https://store.steampowered.com/app/450390/The_Lab/

39. Steam. (2020, March 23). Half-Life: Alyx. [Computer software]. Retrieved March 15, 2021, from https://store.steampowered.com/app/546560/HalfLife_Alyx/

40. Thales. (2021). Thales Nederland. Retrieved March 01, 2021, from https://www.thalesgroup.com/en/countries/europe/netherlands

41. The Witcher 3: Wild Hunt [Computer software]. (2015) CD Projekt RED.

42. Unity. (2020, June 5). Draw call batching. Retrieved March 10, 2021, from https://docs.unity3d.com/2019.3/Documentation/Manual/DrawCallBatching.html

43. Unity. (2021, January 26). Explore, learn and create with the new HDRP Scene template. Retrieved March 02, 2021, from https://blogs.unity3d.com/2021/01/07/explore-learn-and-create-with-the-new-hdrp-scene-template/

44. Unity. (2020, December 23). Optimizing your VR/AR EXPERIENCES. Retrieved February 17, 2021, from https://learn.unity.com/tutorial/optimizing-your-vr-ar-experiences#5fe2ad02edbc2a08ce47222b

45. University of Maryland. (n.d.). Is My Source Credible? - UMGC Library. Retrieved May 9, 2021, from https://sites.umgc.edu/library/libhow/credibility.cfm.

46. Vertigo Games. (2016, December 6). Arizona Sunshine. [Computer software]. Retrieved March 15, 2021, from https://store.steampowered.com/app/342180/Arizona_Sunshine/

47. Wave Harmonic. (2021). Crest ocean SYSTEM URP: PARTICLES/EFFECTS: Unity asset store. Retrieved June 02, 2021, from https://assetstore.unity.com/packages/tools/particles-effects/crest-ocean-system-urp-141674

48. Yang, B., & Brown, T. (1992, July). A Cross-Cultural Comparison of Preferences for Landscape Styles and Landscape Elements. Retrieved March 5, 2021, from https://www.researchgate.net/publication/43116952_A_Cross-Cultural_Comparison_of_Preferences_for_Landscape_Styles_and_Landscape_Elements

49. Zhao, J., Xu, W., & Li, R. (2017, April 30). Visual preference of trees: The effects of tree attributes and seasons. Retrieved March 22, 2021, from https://www.sciencedirect.com/science/article/abs/pii/S161886671630454X

Appendices

Appendix A: Thales Moodboard

The Thales moodboard assembled by the student, shown in Figure 17.

Figure 17: The Thales moodboard

Return

Appendix B: The Graph with the Diagram of the Potential Environments

The student summarized the brainstorming sessions and requirements set by Thales. The graph is shown below in Figure 18 and can be accessed via https://drive.google.com/file/d/1jMjigAfhQhuepSKEm3nenxoj_wSdKOaH/view?usp=sharing

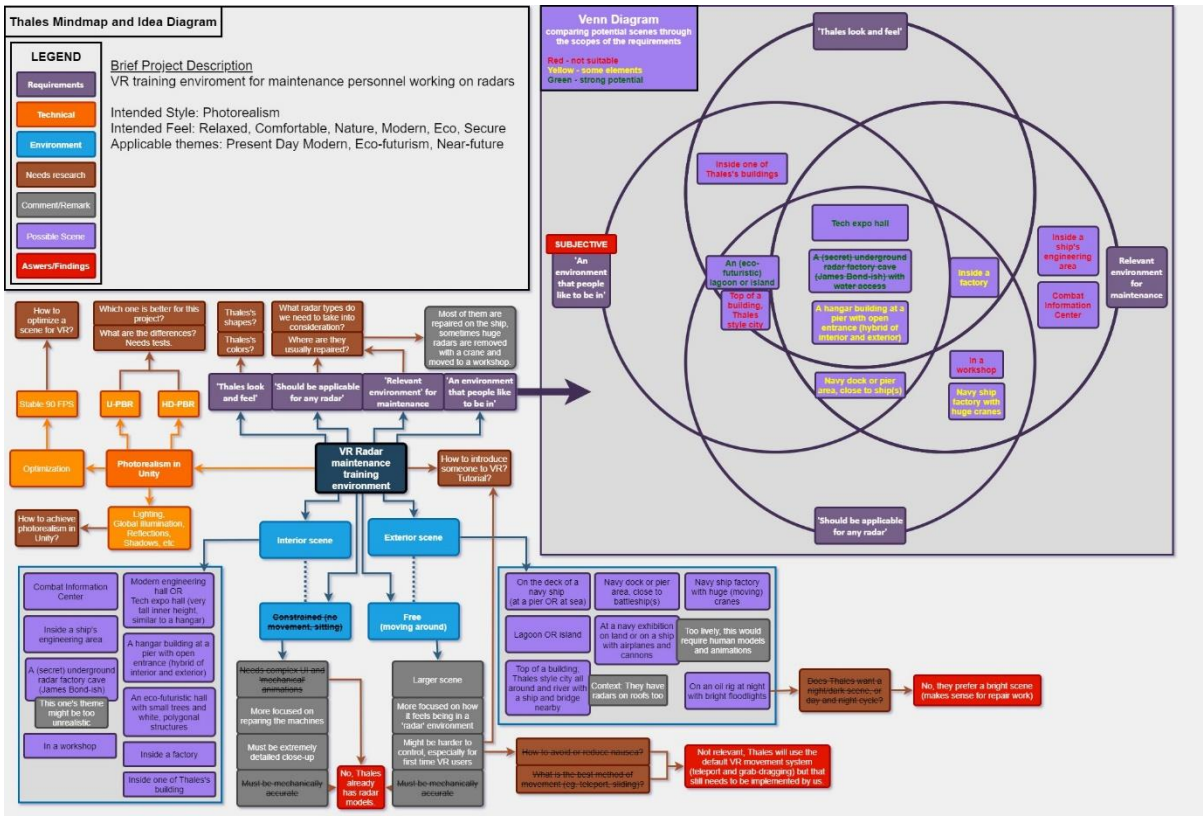

Figure 18: The Graph with the Diagram of the Potential Environments

Return

Appendix C: The Secret Cave Idea's Graybox

Graybox models of the secret cave concept were assembled by the student, shown in Figure 19. The idea was to have a somewhat overgrown secret cave with water access, interesting rock formations such as stalagmites, and a clean, bright modern area to create a fascinating thematic contrast. The idea of the diegetic radar selection, mentioned in Appendix J, was also born in the student during this. This concept was discarded because it felt too dark for the client, and it opposed the later defined direction of a bright scenery.

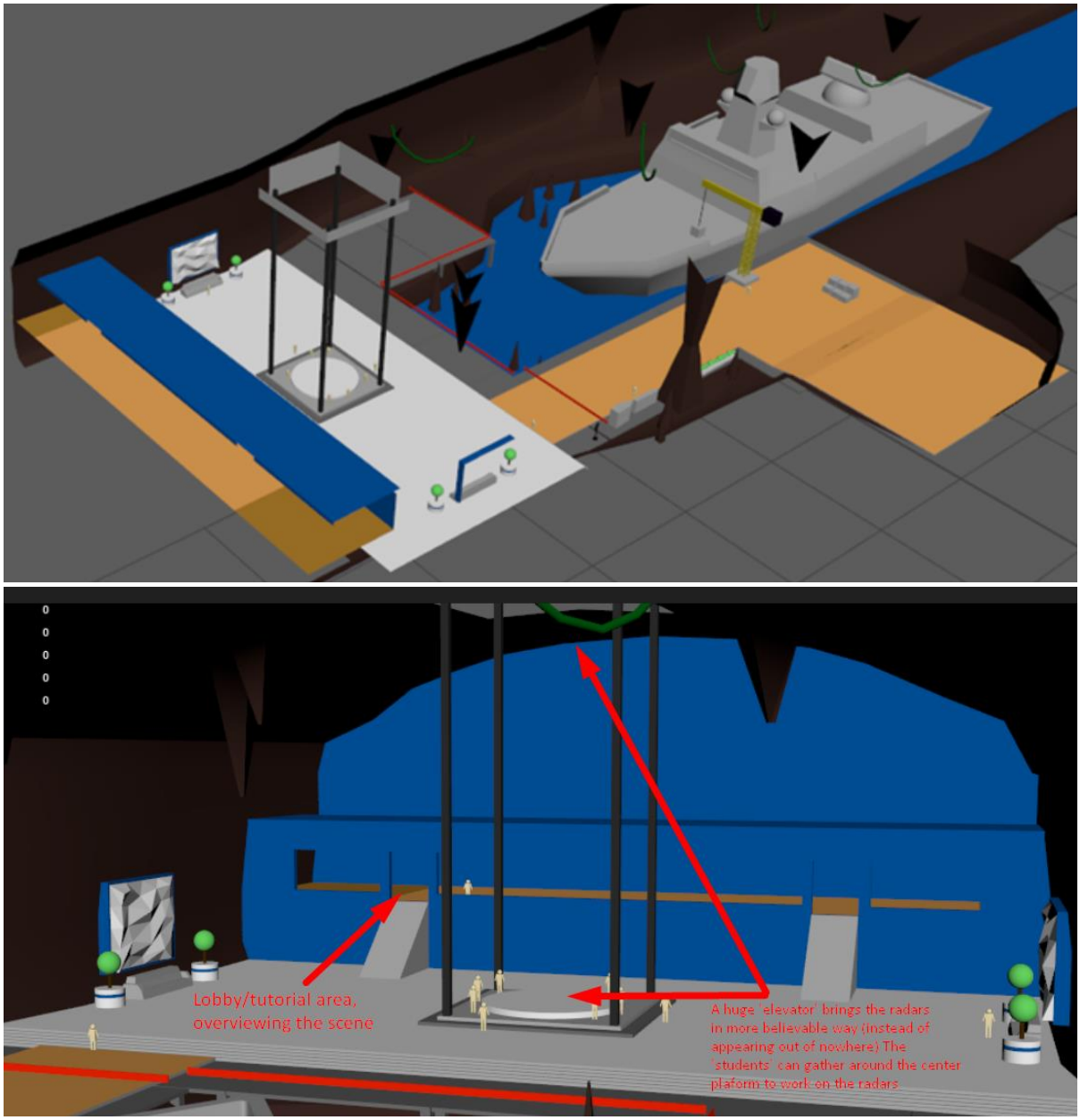

Figure 19: The secret cave graybox

Return

Appendix D: The Exhibition Hall Idea's Graybox

Graybox models of the exhibition hall concept were assembled by the student, shown in Figure 20. The idea was to have a huge exhibition hall with access to the sea, a sophisticated ceiling design with large windows for natural lighting, and a navy ship docked in the middle. An exhibition of radars or other navy items would have been placed around the scene, and plants to contrast the structure. The gameplay would have taken place on the ship. This concept was discarded because it felt too industrial for the client, and it opposed the later defined direction of natural scenery.

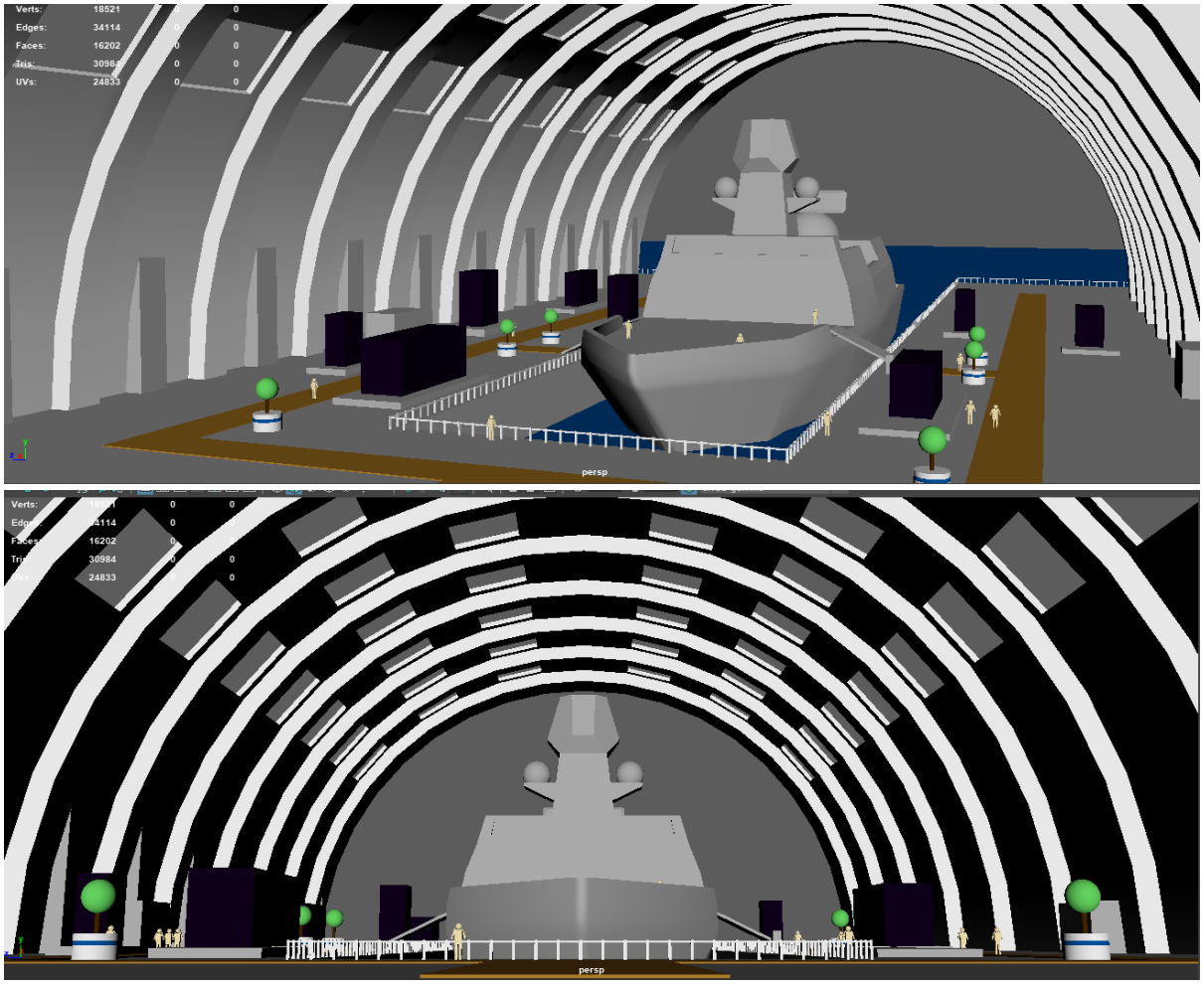

Figure 20: The exhibition hall graybox

Return

Appendix E: Rendering Passes Explained by Unity

"*Because Google VR has two Textures, one for each eye, Unity needs to render a Scene twice. With multi-pass rendering, Unity attempts to avoid duplicating work required for each eye, such as shadows, which don't need to be rendered twice. Multi-pass still renders most objects twice, but renders the Scene graph once. This creates more accurate lighting, but comes at a computational cost, since the two renderings do not share GPU work across Textures. This is the least efficient rendering path, but works on most devices.*
Single-pass, on the other hand, packs the two Textures into one big Texture (known as a double-wide Texture). It goes through the Scene graph only once, so it's much faster on the CPU. However, it requires a lot of extra GPU state changes to accomplish this.
Single-pass instancing allows for even simpler integration and better performance. It reduces the CPU overhead, just like single-pass (lowering the number of draw calls), but also reduces the GPU overhead more than single-pass. The GPU is able to more efficiently process the draws, and you minimize state updates by not having to change the viewport between draws unlike with traditional single-pass." (Unity, 2020)

Return

Appendix F: The RTX 2080 Ti's Performance Chart

"*The RTX 2080 Ti easily manages Ultra settings with 184.62 unconstrained FPS, and it neither drops nor requires any frames to be synthesized. It handles Ultra settings plus Level 8 Fidelity setting, delivering 131.37 unconstrained FPS, while dropping only 1 frame or requiring it to be synthesized. If the game engine manages the framerates, occasionally the Fidelity level will drop a notch on the most demanding scenes.*" (Poppin, 2020). The chart is shown below in Figure 21.

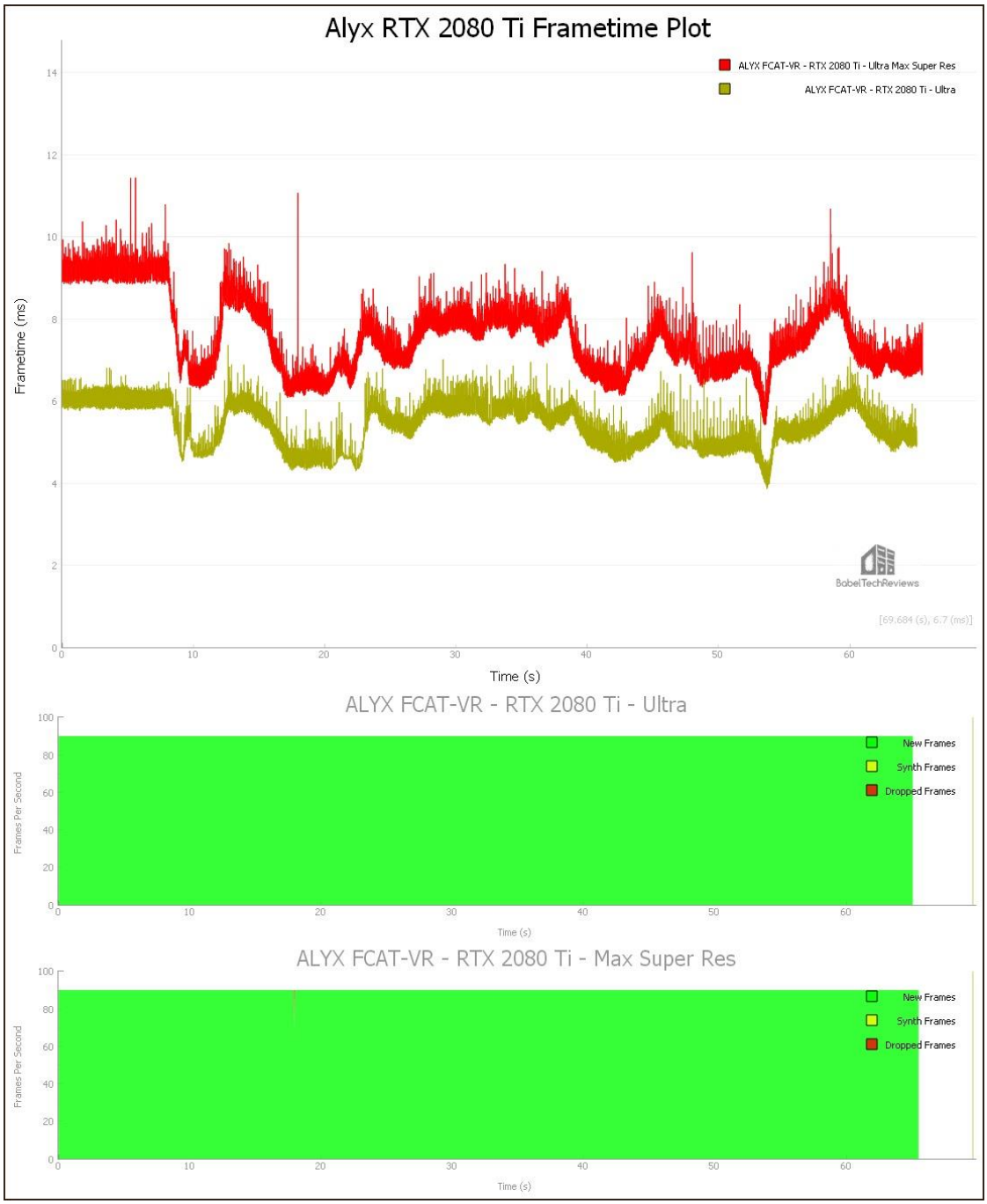

Figure 21: The RTX 2080 Ti's performance chart

Return

Appendix G: The Design Thinking Model

"Design Thinking is a design methodology that provides a solution-based approach to solving problems. It's extremely useful in tackling complex problems that are ill-defined or unknown, by understanding the human needs involved, by re-framing the problem in human-centric ways, by creating many ideas in brainstorming sessions, and by adopting a hands-on approach in prototyping and testing.[…] The five stages of Design Thinking, […] are as follows: Empathise, Define (the problem), Ideate, Prototype, and Test." (Dam & Siang, 2020).

A chart of the 'Design Thinking Model' is shown in Figure 22.

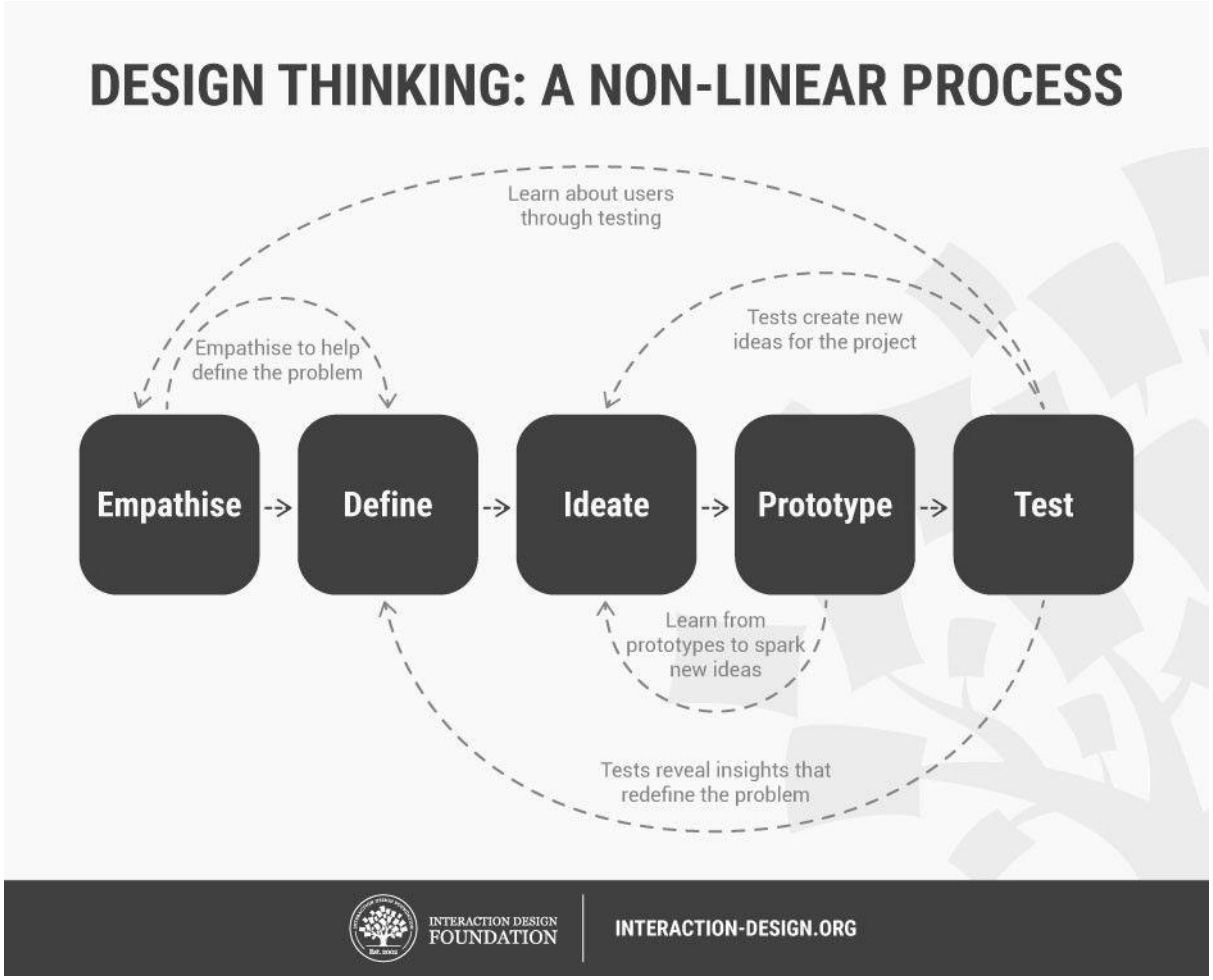

Figure 22: The Design Thinking Model

Return

Appendix H: Vincent Callebaut's Architecture

In Vincent Callebaut's ecological architecture, the focus is on natural curves and harmony with nature. White dominates many buildings. Some examples are shown below in Figures 23, 24, and 25.

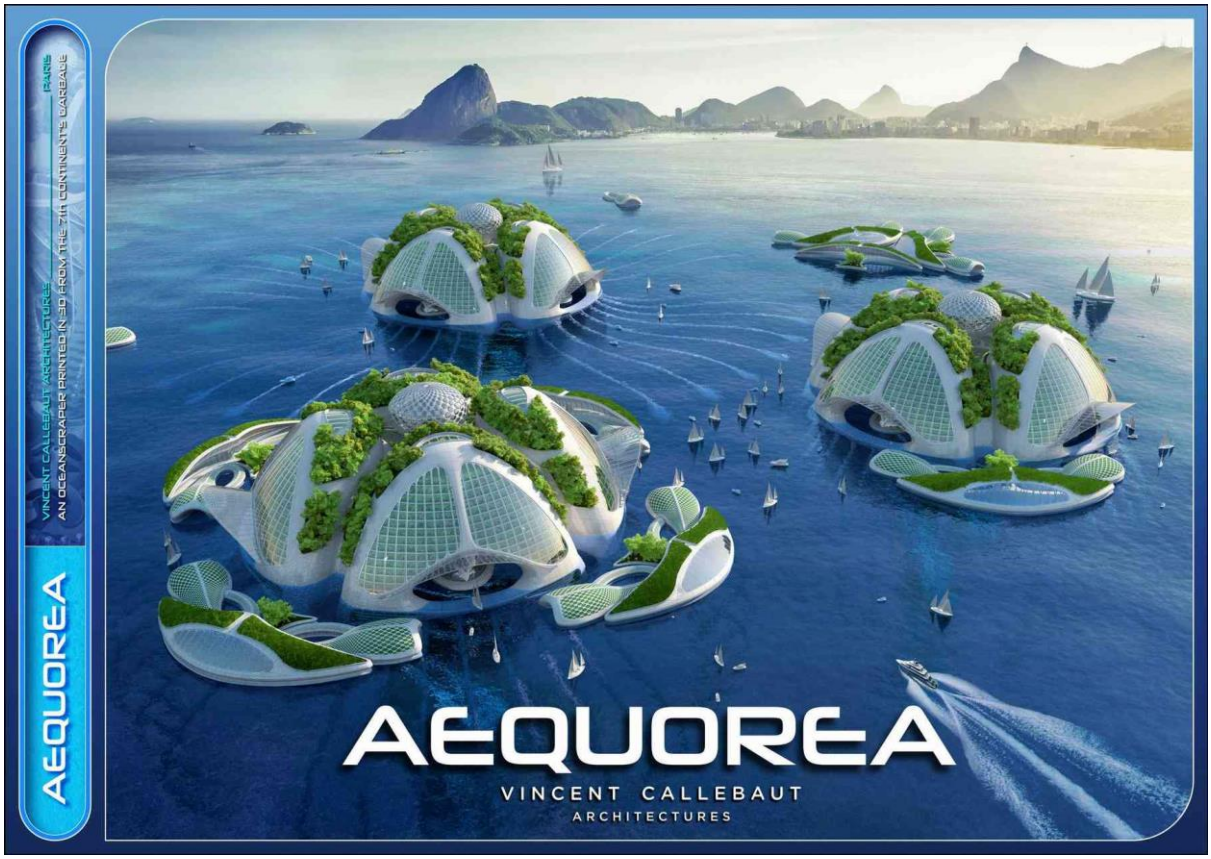

Figure 23: AEQUOREA by Vincent Callebaut, image 1

Figure 24: AEQUOREA by Vincent Callebaut, image 2

Figure 25: PHYSALIA by Vincent Callebaut

Return

Appendix I: Graduation Schedule 2021

Start of Graduation	February 8
Submission of the Preliminary Implementation Plan	March 2
Company Visit	March 2 – March 16
Submission of the Final Implementation Plan	March 16
Preparation Phase	Until March 31
Production Phase	From April 1
Submission of the Draft of the Graduation Report	May 11
Submission of the Final Graduation Report	June 15, 12:00
Graduation Presentation	June 28 – July 2
Diploma Ceremony	July 12

Return

Appendix J: Ideas Regarding the Platform

Because of the long distance between the observatory and the platform, a simple teleportation method was necessary. The student's idea was to have teleportation stands with holographic orbs. These teleports would have only activated if the player used the grab button on the orb while standing on the teleporter, as shown in Figure 26. This idea would have served as an introduction for the grabbing mechanic. Thales preferred to teleport between the buildings by simply pointing the controllers and pressing the teleport button, so the concept was updated.

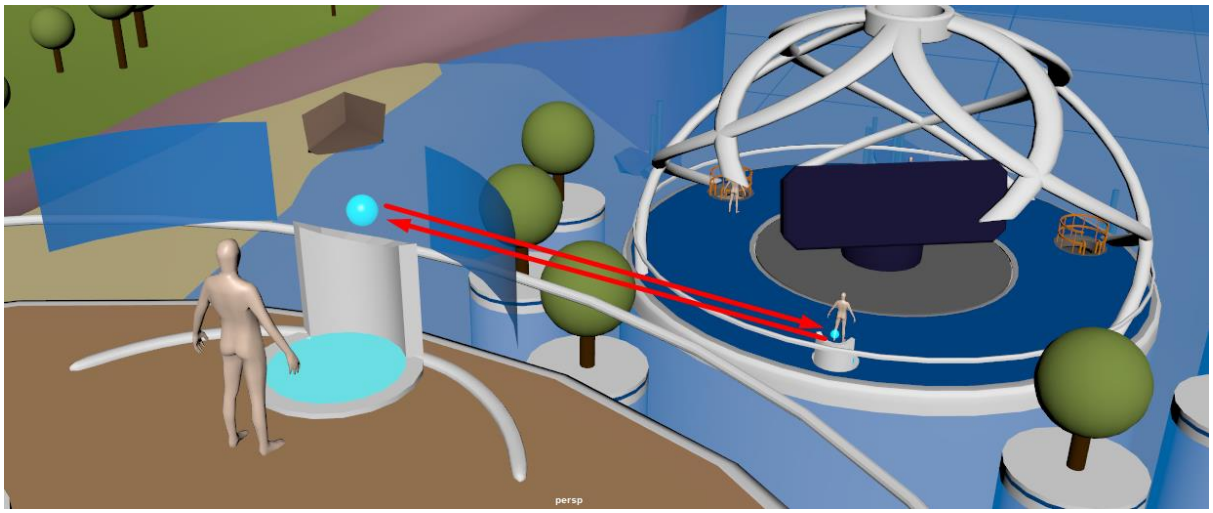

Figure 26: The long-distance teleport concept

The idea of including whiteboards for making notes, as seen in Half-Life: Alyx (2020), was proposed to Thales. Thales liked the idea, though the team could not implement it because of this feature's highly technical nature.

Another idea was suggested by the student regarding the selection of radars. They could have appeared in a diegetic way by selecting one on a screen on the platform. A circular door would have opened, and the corresponding radar would have been elevated, as shown in Figures 27 and 28 below. Thales liked the idea, but they dismissed it because, according to them, only one radar is active during a training session, and they preferred to select the radars before entering the scene.

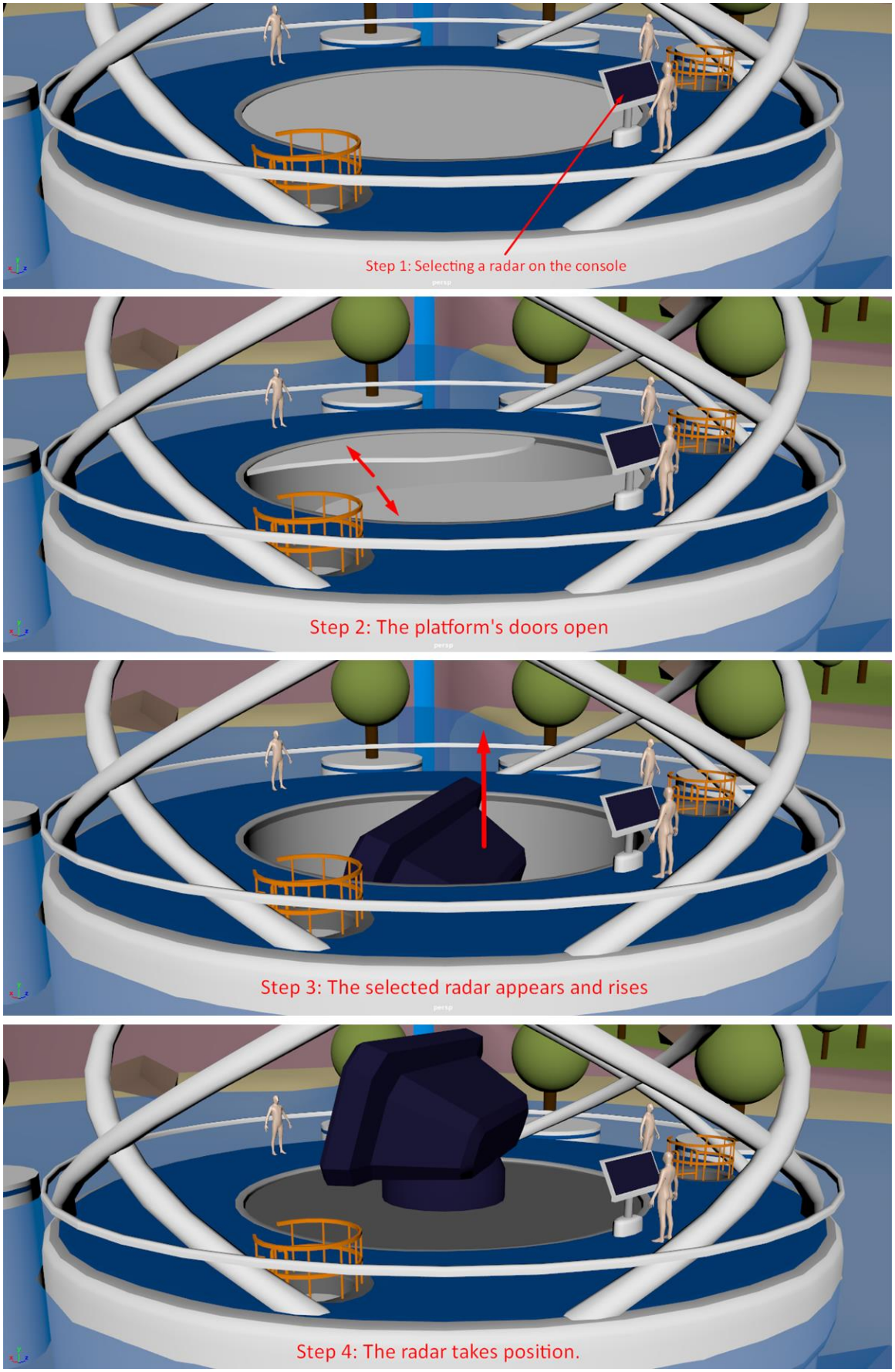

Figure 27: The radar selection explained

Alternative door animation

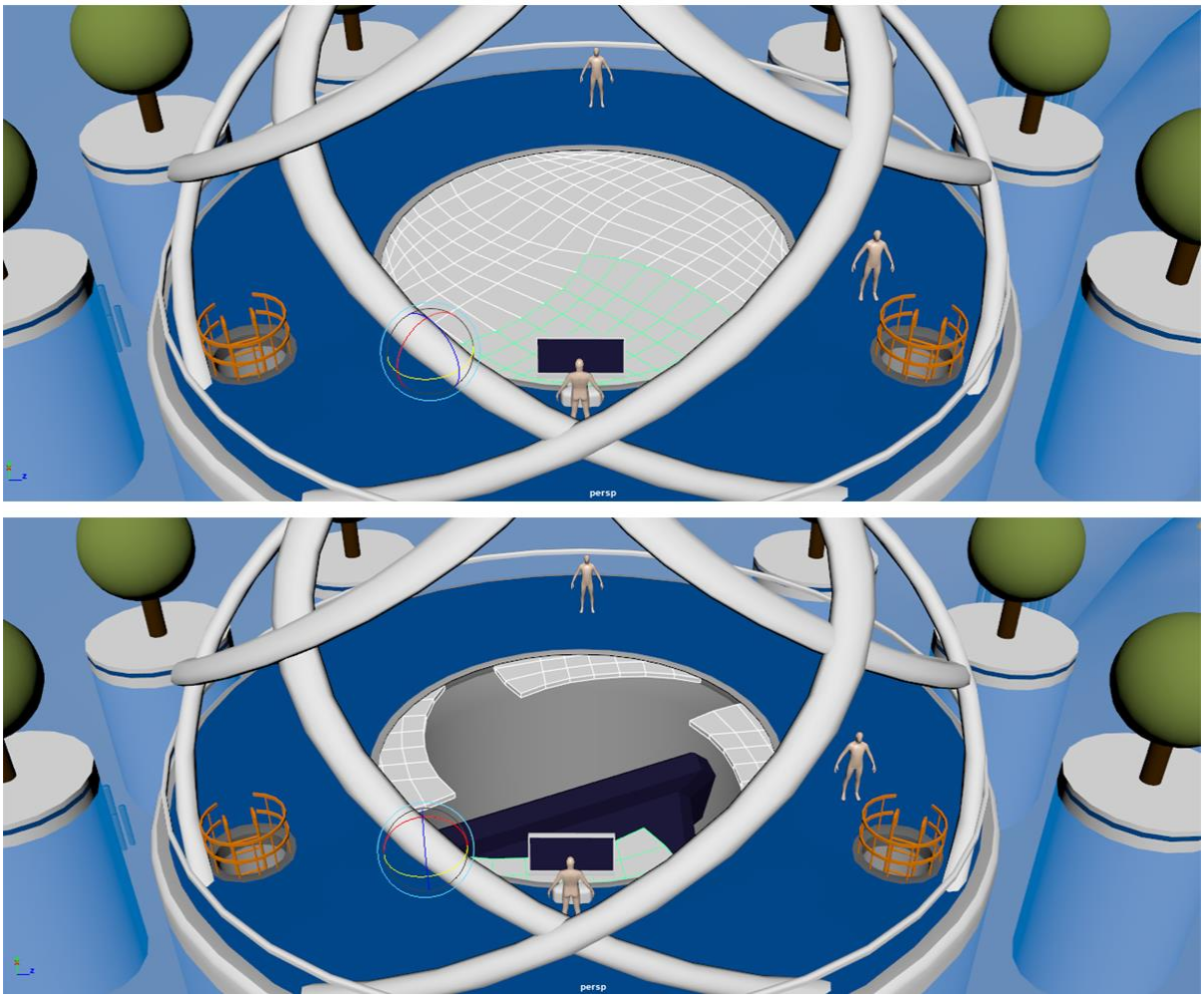

Figure 28: Alternative door animation

Return

Return

Appendix K: The Tutorial

As shown in Figure 29, many popular VR games, such as Half-Life: Alyx (2020), The Lab (2016), Arizona Sunshine (2016), use accurate models of the controllers in VR and simply prompt what button to press on them to show the controls. The first prompt is always the movement.

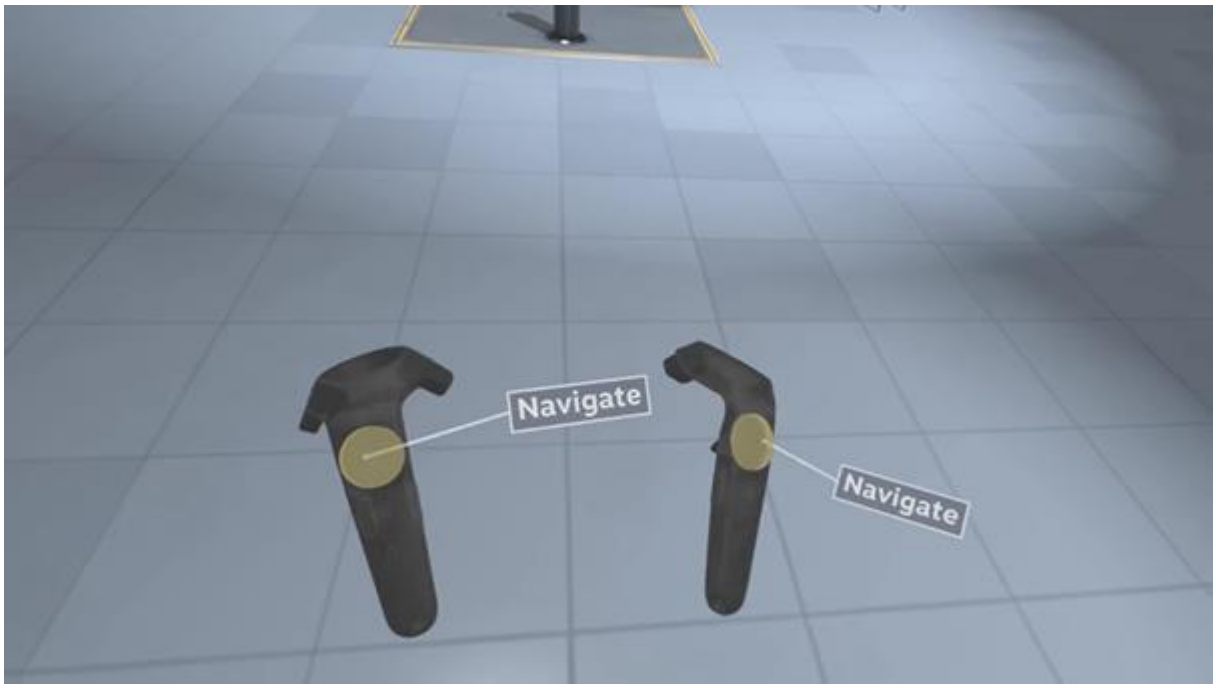

Figure 29: The controls shown on the handles in The Lab

The team intended to implement a similar minimalist tutorial and control system. The student wrote a script of the tutorial events.
‘The screen fades in, the player is spawned in the observatory, near the large portal. The observatory is overviewing the scene. A button prompt for movement appears almost immediately. The text could say 'Press to aim, release to teleport', which is clearer than 'Press to move' or 'Navigate'. There would be a screen with an illustration explaining which button to press and what it does without words for even more clarity. There should be an arch or laser pointer whenever the teleport button is held down. Once they have moved around a bit, the movement prompt will disappear, and they are directed to aim at the platform. The pointer should turn another color when connecting with a giant invisible collider on the platform to indicate a successful connection. When the button is released, the player appears on the platform in a fixed position. This long-range teleportation works both ways. ‘

Return

Appendix L: The Different Leaf Models

Five different leaf meshes were tested to find the most aesthetic and performant one, shown in Figures 30, 31, 32, 33, and 34.

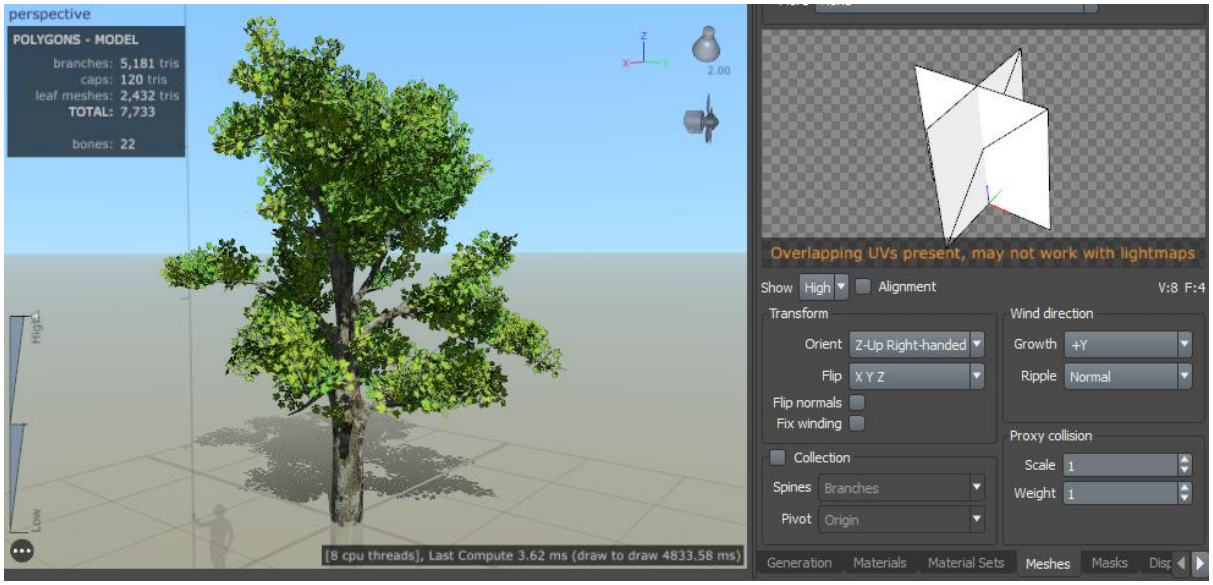

Figure 30: Cross leaves

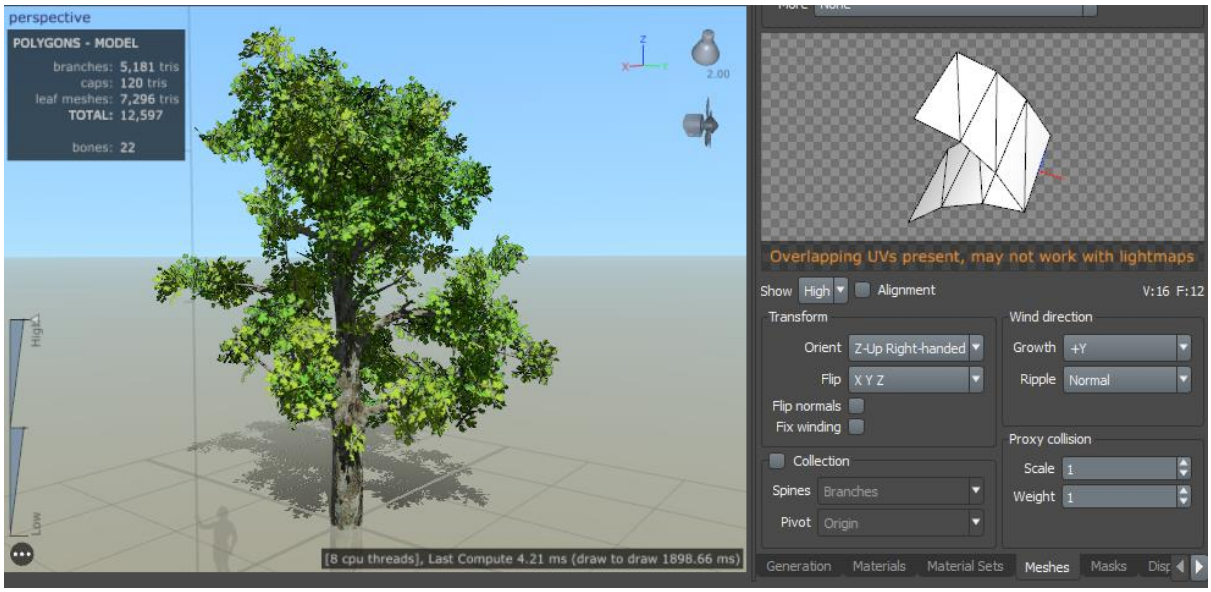

Figure 31: 'Fan' leaves

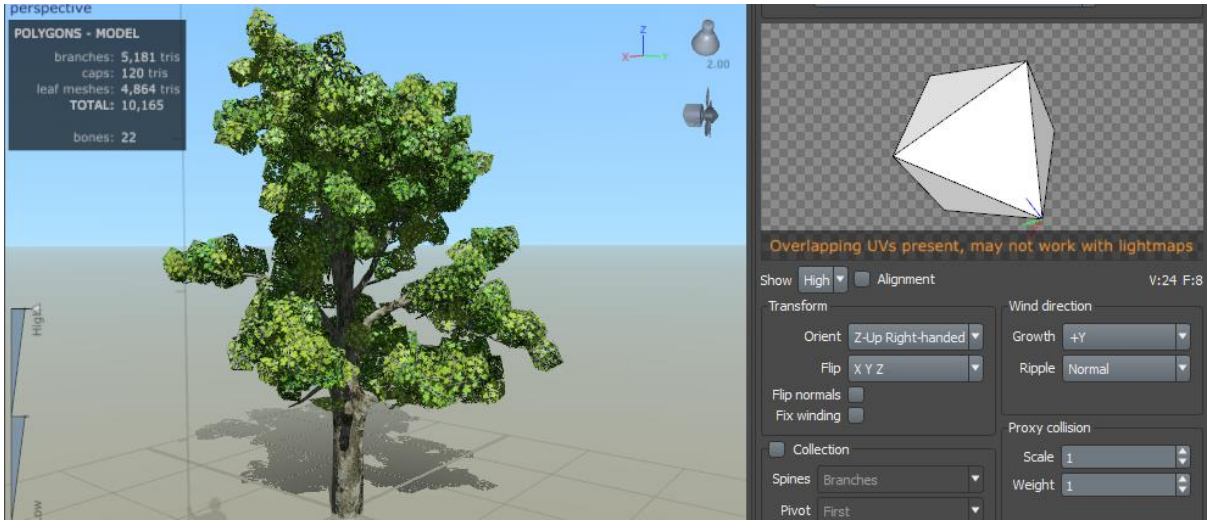

Figure 32: Octahedron leaves

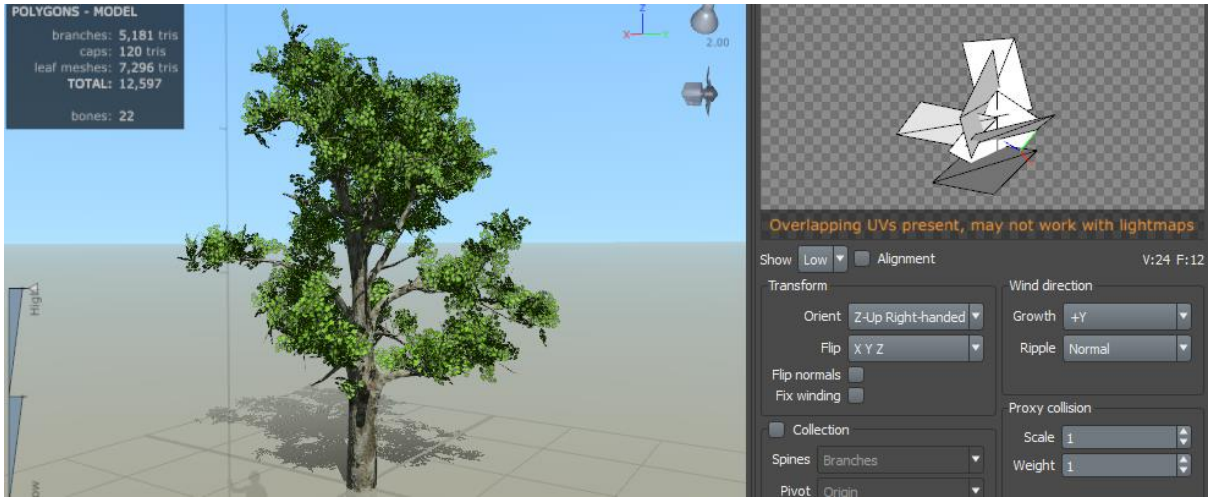

Figure 33: Poly plane leaves

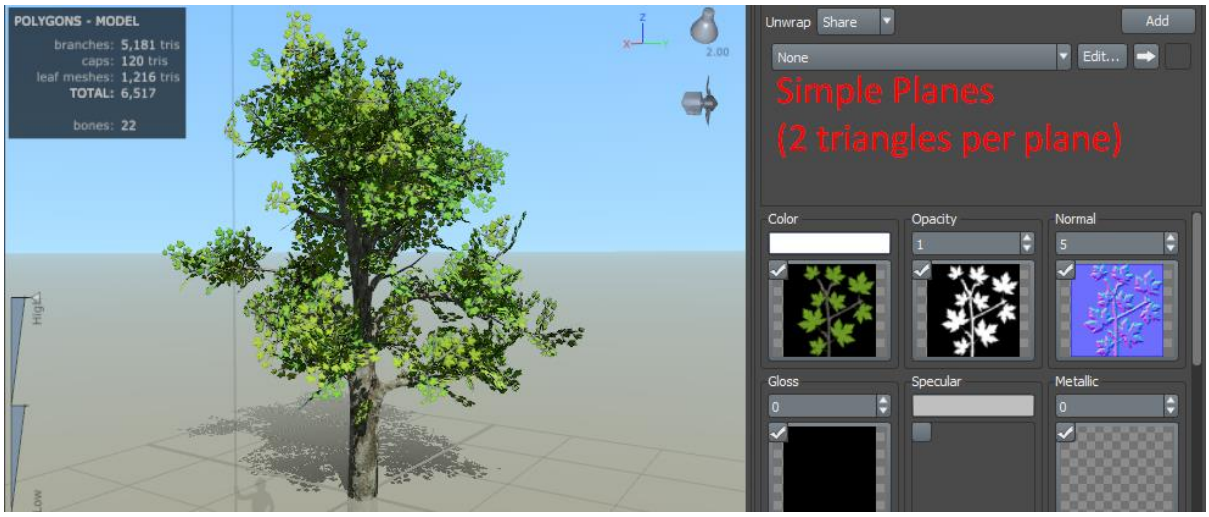

Figure 34: Simple plane leaves

Return to chapter VIII.3.
Return to Appendix N

Appendix M: Testing the Draw Calls of the Trees

A new, empty scene was created to test if the trees were causing too many draw calls. One hundred Unity's standard trees were compared to one hundred custom trees, as shown in Figures 35, 36, and 37. Unity's trees caused around 564 draw calls while the custom ones only 18. With both tree stacks enabled, the scene had 576 draw calls. This test proved that Unity's standard trees were not batching, even though they were set to static.

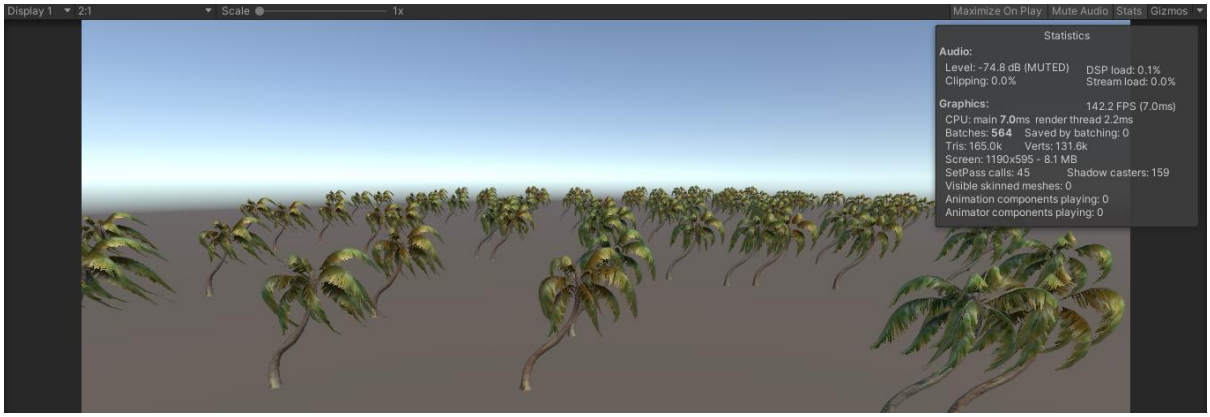

Figure 35: One hundred standard Unity trees, 564 draw calls

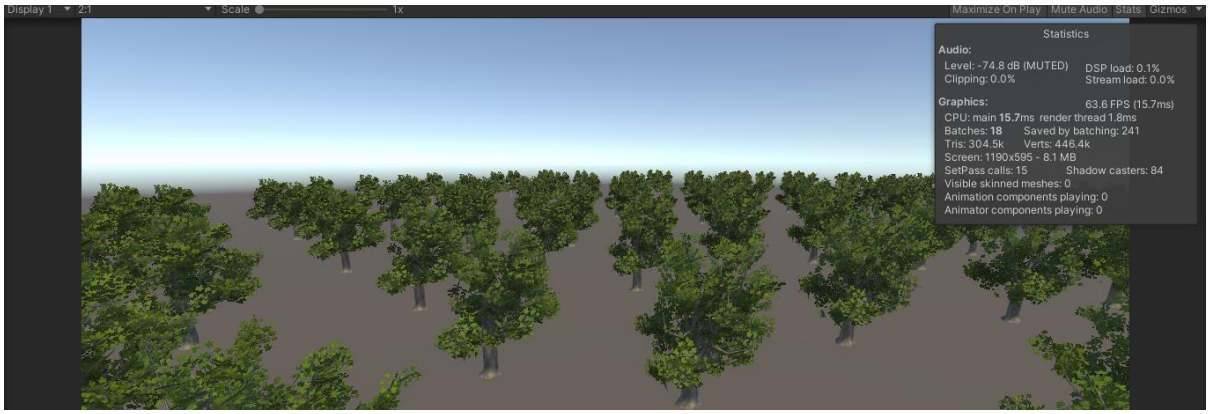

Figure 36: One hundred custom trees, 18 draw calls

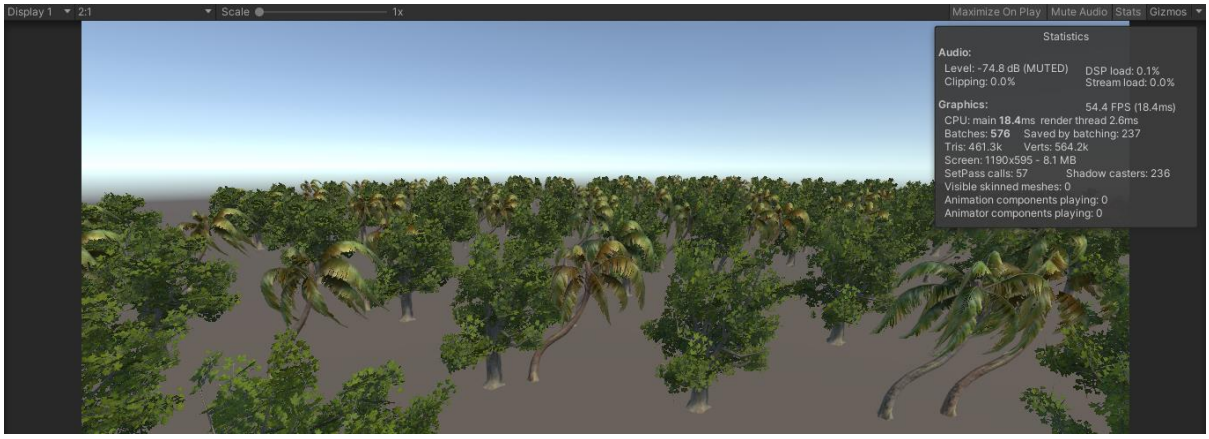

Figure 37: Both tree stacks enabled, 576 draw calls

Return

Appendix N: Researching SpeedTree

The student knew about SpeedTree before the project, and it seemed to be the perfect tool for it. Many triple-A game studio's games used SpeedTree in their pipeline, for example, The Witcher 3: Wild Hunt (2015), Detroit: Become Human (2018), Ghost of Tsushima (2020), and Resident Evil 8: Village (2021). The student decided to learn SpeedTree because it was an industry-standard tool for foliage creation.

Online tutorials and blogs were visited for learning, including the official SpeedTree website and YouTube channel. Many hours were spent practicing. It turned out to be a simple-to-use generator-based tool, meaning that changing specific values influenced every part of the mesh. After learning the basics, it was quick and easy to create certain types of trees. The challenge was keeping the triangle count low while maintaining a visually pleasing, lush tree.

The trees were shaped based on the team's research about tree preferences mentioned in chapter **VIII.1.A.**. Short trunks and wide, lush, thick crowns were the key aspects. For the best results, it seemed necessary to create a custom twig texture. The student did this by assembling one in Photoshop made of realistic leaf and branch images. Large maple leaves were used to achieve a thicker-looking crown. Conifer evergreens were purposefully avoided because wider, lusher trees were generally preferred by most. (Gerstenberg & Hofmann, 2015) (Nelson, Johnson, Strong, & Rudakewich, 2002). Once the layout of the twig was satisfactory, different texture maps were made. A color map, an opacity map, a roughness map, and a height map were exported. The heightmap was later converted into a normal map using an online tool called 'NormalMap Online' by Christian Petry.

Unfortunately, an unidentified issue caused the custom trees and their shadows to flicker in Unity. The team made many attempts to fix this error, including adjusting shadow resolutions, disabling LODs, and changing quality settings, but none of these resolved it. Finally, it was solved by replacing the trees with another model made by the student. This tree had slightly more triangles per LOD, but the student preferred how it looked. Replacing the maples with this new tree solved the complication for an unknown reason, but the error seemed to have correlated with the custom twig textures made by the student, so the final trees used SpeedTree's built-in leaves instead.

The student learned that the size of the individual leaves and the shape of the leaf mesh also had a fundamental impact on the trees' overall aesthetic and crown density. Many model shapes were tested to find the most suitable one. In **Appendix L**, different leaf models are showcased. In the end, simple planes were used for the final trees because they offered fine aesthetics while also having the least triangles.

The student believes that one of the strongest features of SpeedTree is built-in LODs, including billboards. The LOD system helped with performance in particular. The number of LODs and the amount of triangle reduction could be controlled via graphs. The LOD aspect was especially beneficial because the manual creation of different density models is time-consuming.

In the final build, two tree models were present, one sapling and one adult tree. The sapling model was a mostly unchanged standard SpeedTree asset. Both trees had three LODs, plus one nine-sided billboard asset for the farthest distance.

Return to chapter VIII.3.A.

Appendix O: What climate, season, and colors are generally considered relaxing?

Shinrin-yoku is a Japanese expression meaning ‘forest bathing’, referring to the positive effects of walking in a forest and taking in the atmosphere. Field research was conducted by Park et al. (2010) in twenty-four forests across Japan to prove that shinrin-yoku has meditative effects. Volunteers were split into two groups and walked in either a city or a forest. On the second day, the roles were reversed. Many physiological tests were used on the volunteers before and after the walk. The results strongly suggested that ‘forest bathing’ indeed helped people become relaxed. This evidence is lower pulse rate, blood pressure, cortisol concentrations, and greater parasympathetic nerve activity.

In an experiment by Kweon et al. (2008), two groups of students were tasked to work in office environments. One office was decorated with abstract paintings, while the other with nature paintings. The volunteers had to perform a mildly annoying task on a computer. Results indicated that the difference in decoration had a significant impact on the stress levels of male volunteers. Females were mostly unaffected. This data suggests that nature has a stress-relieving effect, even as illustrations.

According to Beute and Kort (2013), viewing natural environments positively affects stress, even when the scenery is digital. They also claim that people prefer nature over the city, bright over dark, and sunny over shaded environments. Johansson (2018) suggests that incorporating small plants in an office environment may significantly impact the workers’ mood. A study conducted by Lee et al. (2015) shows that viewing a city scene with green roofs boost sustained attention and cognitive functions compared to viewing bare concrete roofs.
Based on the research of Kaufman and Lohr (2008), plants are generally relaxing to look at, but green-colored canopy trees are especially calming and realistic looking. It is noted that different tree colors may evoke various physiological responses.

A study by Yang and Brown (1992) observed the visual preferences for landscapes of different cultures. A selection of international volunteers was involved. Korean, Japanese, and Western natural sights were presented. The results showed that most groups preferred Japanese landscape and water elements. Korean tourists had a general preference towards Western landscapes, while Western tourists mainly chose Korean sceneries. Water and vegetation were generally favored over rocky terrains. Another study also suggests that people prefer images of forest and water more than rock and sand. (Hardcastle, 2018). Personal experiences and societal values may also influence landscape preferences.

To summarize, several studies confirm that natural environments have positive effects on humans, including lower heart rates, more focus, and less stress. People have an innate preference toward natural scenery over urban. The color green reduces stress, even if it is merely a visual representation of nature, such as paintings or digital art.

Return to chapter VIII.2.A.

:P