The Borculo Storm Project

Creating a realistic Storm in VR

Student name: Anna Kemper

Student number: 451520

Graduation Supervisor: Yvens Rebouças Serpa

Company: Saxion XR Lab

Company Client: Matthijs van Veen

Abstract

The Borculo Storm Project aims to create a realistic VR-experience of a storm disaster. The Storm hit the Netherlands in 1925. The project also aims to raise awareness for natural disasters and climate change. It is supposed to be an exhibit of a future weather museum. Previously, the project has been worked on by students during the Smart Solution Semester in the Saxion XR-Lab. During that phase Unreal was used as a game engine and the students created a small scene in which a basic storm is simulated. The group now switched to Unity as it is more convenient to work with. The old project files consisted of a basic particle set up, which needed to be reworked. The particles appeared to be appropriately sized on screen but in VR the scaling increased and created an unrealistic look and feel. To convey the situation more appropriately, mesh destruction was used during this project to simulate the storm effect on buildings. Also, a realistic sound environment was added and the overall light in the scene was adjusted to create an immersive experience. Next to the visual and acoustic aspects, hyper VR was integrated, which means, the storm is not only going to be experienced visually over the VR goggles, but also physically, by using hardware such as industrial fans that are going to react to the stages of the storm.

For this graduation, the focus was laid on the recreation of the storm visuals and the realism and immersion of the scene itself. Realistic particles and lighting were the main subject.

This report covers the process of simulating and recreating a massive storm with particle Systems and lighting in Unity. Visuals and information about the Storm in Borculo and similar natural disasters were gathered through the study of reports, videos, and images. Multiple different particle setups were created in the game engine and tested with a broad audience. Feedback was collected using surveys to figure out the best and most immersive set up for a final version.

The final product consists of a dynamic storm system in Unity. The system consists of:

- Particles such as rain, dirt, and leaves
- Lighting
- Fog
- Animation of foliage
- A weather system to combine the elements mentioned above.

The product was tested twice by using videos and a build of the VR experience. In doing so the exact settings for the weather system were found and later polished.

Table of contents

1. Introduction

Extreme weather is one of the biggest growing threats of the current time. According to Roz Pidcock and Robert McSweeney, the increasing number of extreme weather events such as storms, floods, droughts, and heatwaves is a result of human activity (carbonbrief.org, 2021). 70% of 405 extreme weather events were made more severe and more likely by human activity and 92% of 122 studies have shown that climate change in general made the events more likely or severe.

The UNESCO states that it is an important response to climate change to educate about it. It helps people understand and encourages changes in attitude and behaviour. Without awareness and education, people see no need in reducing their emissions and stabilizing greenhouse gases and adapt to the changing environment instead of acting against the devastating effects (UNESCO, 2020).

Even though extreme weather events are usually associated with tropical countries, they become more and more common in Europe as well. The Climate Change post says that number of disastrous weather events is growing and 50% of those events are storms (*Damage and Fatalities Extreme Weather Events in Europe on the Rise*, 2019).

To address the issue of the rising threat of extreme weather, the Fire Department Museum in Borculo developed, together with the Saxion XR Lab, a project proposal about a realistic storm simulation. The goal of the project is to visualize the impact of extreme weather in virtual reality. The project is supposed to raise awareness for climate change.

2. Company Outline

The project is realised with the help of the Saxion XR Lab for the Fire Department Museum in Borculo and the Historical Society Borculo. The XR Lab is a non-profit facility at Saxion University that focusses on the development of extended reality and serious games. It was officially opened November 2019 and is organized by project leader Matthijs van Veen (Hogeschool Saxion, 2019).

The lab offers a space for students to research and work on projects as well as a variety of technology reaching from motion Capture suits over VR Headsets to high quality audio equipment. Furthermore, experienced teachers offer support and answer questions in order to provide a good learning experience (Saxion, n.d.).

The Fire Department Museum in Borculo is a museum focused on the history of firefighting and exhibits old fire trucks and pumps as well as methods of firefighting before cars became the main method of transportation. Apart from that the museum also has a storm department in cooperation with the historical society. It exhibits images and remnants of the storm disaster that took place in the area of Borculo in 1925 (Kistemaker, n.d.).

The Historical Society Borculo was found on 20th December of 1995. They aim to uncover history, in particular the history of Borculo. They do not only focus on the early times but the overall development of the city up until the recent past. Currently the society is managed by Peter Nieuwenhuis (*Bestuur | Historische Vereniging Borculo*, n.d.).

3. Objectives of the Client

Client Question

The overall objective set by the client is to elaborate on an already existing project and make it as believable and immersive as possible. Matthijs van Veen wants the group to enhance a project that has been worked on previously by students from the XR Lab. He wants a finished prototype so that it can be of great use for the storm disaster exhibition of the fire department museum in Borculo.

The Fire department museum has the objective to not only use the experience in the future as part of their own exhibition but also for a weather museum that is in the making. They want to have a realistic, immersive, as well as frightening experience to raise awareness about the dangerous effect that can be caused by climate change, such as a massive.

Problem Analysis

The client's and stakeholder's problem is drawing people to a museum. Museums depend on visitors and donations most of the time so attracting new visitors is always of importance. Next to that, the client also wants to raise awareness for climate change as it fits into the museum's theme.

To allow visitors of the museum to immerse themselves in the storm experience, realism and authenticity is key, especially when it comes to the representation and visualization of the storm itself and the effects it has on the environment. This creates a challenge because a VR experience needs a good performance to ensure wellbeing of users. If the experience has lagging problems the user can feel uncomfortable, nauseous, and sick. To allow realistic visuals, the decision was made to use the High-definition Render Pipeline, which is a plugin for Unity that makes highly realistic visuals possible. However, it is also performance intense and decreases the frame rate that is important for the usability of virtual reality.

Limitations

Another challenge are the limitations by the Covid situation, as the current circumstances make testing very hard. Usually, a larger group of unbiased people from the target group would be required in combination with multiple hardware solutions to get a scientifically valid result. However, the pandemic does not allow that. It is, for example, not possible to invite people to the XR lab and have them test the experience.

In order to do some testing despite these issues, peers have received videos and VR users were able to test the experience first-hand. However, those tests will not give the exact same result as using the hardware that is actually going to be used. The work also cannot be tested in the context of Hyper VR because it is not possible to visit the XR Lab at Saxion where the hyper VR environment is going to be set up.

4. Theory

4.1 The storm disaster in Borculo

The storm that hit Borculo on August 10th, 1925, is also known as "cycloon van Borculo" and it was not a regular storm. According to Weer.nl, the storm was caused by an area of low pressure and an area of high pressure that met above the Netherlands and caused a breeze to blow warm and humid air into the country. At the same time, cold air was coming from the south. According to the article, it is likely that the event caused supercells, a thunderstorm that has a strong rotating updraft (*weather.gov*, 2015), and thus caused a number of tornados to form (*Meteo Consult Weernieuws Archief*, 2005, l. 14-30).

Eyewitnesses describe that the event started with lightning and thunder, the sky became dark, and a heavy wind came up. From their house they could see a huge storm front form. Objects outside were carried away by the intensity of the wind, roof tiles were ripped from the roofs, windows were pushed open and smashed, houses were shaking, and rain and leaves were flying around fast and horizontally. The storm lasted for about ten minutes and disappeared almost as quickly as it appeared (*Meteo Consult Weernieuws Archief*, 2005, l. 38 -55).

Fig. 1 - F4 Tornado Damage Example (2005) and three adults sitting at a table next to their collapsed house (1925)

Experts think that a tornado with a diameter between one and two kilometers, F4 on the Fujita scale, has caused the damage in Borculo. ('Stormramp van 1925', 2020)

These facts and eyewitness accounts create a detailed image about what the storm looked like. By considering the Fujita scale, even physics can be considered for the recreation of the storm in the virtual reality experience. The Fujita scale is a scale that was developed to divide heavy storms like Tornados and similar into twelve categories according to the damage that they cause and the speed of the wind. Of the 12 categories, only the first five ever occurred naturally ('Fujita-Skala', 2020). Focusing on the storm that most likely destroyed Borculo and its score on the Fujita scale, the damage caused by it is described as "devastating damage"

Taking images of the disaster in Borculo into consideration, the description appears to be fitting.

Fig. 2 - Man in the wrecks of his house and top view of destroyed Borculo (1925)

The images were taken August 1925 in Borculo and coloured in 2009 by H. Hendriksen. The damage shown by the first image of Figure 1 is clearly comparable to the damage presented by the second image of Figure 1, and Figure 2. Analysing it, one can see that the damage in fact is devastating. The storm raged through the city and destroyed the majority of it. Some houses completely collapsed, some are missing their roof and wooden pillars, planks, and beams as well as bricks and bits of plaster are spread all over the area. All of these aspects and especially the objects that become projectiles can be integrated into the particle effects of the storm to further show the dangers posed by it. The fact that the Storm reached a scale of 4 from 5 emphasizes on the severity of the disaster. It is of importance to really display how dangerous the event was.

4.2 Unity

Unity is a platform created by the company Unity Technologies, which allows to create interactive real time content. It can be used to create games, animations, movies, and company software for example for architecture, engineering, and transportation.

Unity allows the use of Augmented Reality and Virtual Reality, offers simple solutions for advertisement, and allows integration of industrial data. The engine is free for students and beginners but also offers a more complete version for companies and professionals. It comes with an elaborated documentation which allows users to easily get into development and an asset store in which free and paid assets can be acquired and integrated into projects. Furthermore, Unity Technologies offers over 750 hours of learning material to advance skills in multiple fields of work such as level design or game development (Unity Technologies, n.d.-i).

4.3 Particle Effects

To display the Borculo storm properly, leaves, rain, small debris, and fog should be present as mentioned by the eyewitnesses in the previous chapter. In order to visualize weather in a game engine, such as Unity, particle systems are of great help.

Particle systems are a large number of small meshes or images that together are able to display different effects such as smoke, magic spells, or rain. Particles can be animated or even be influenced by physics, depending on the engine that is used (Shiffman, 2012).

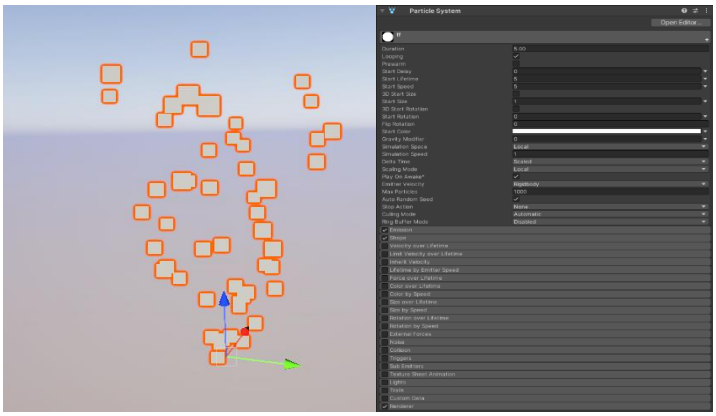

Fig. 3 - Standard Particle Effect in Unity (Screenshot Unity 2021)

When using Unity, particle systems and the particles themselves are dynamic and can be adjusted to fit a variety of purposes and looks (Unity Technologies, n.d.-b). Variables such as the rate of emitted particles over a specific time, the time until a particle disappears, velocity, gravity, and colour can be adjusted as seen in Figure 3. Moreover, particles can not only consist of images but of Meshes as well, which means that for example a flying roof tile could be incorporated into a particle system (Unity Technologies, n.d.-c).

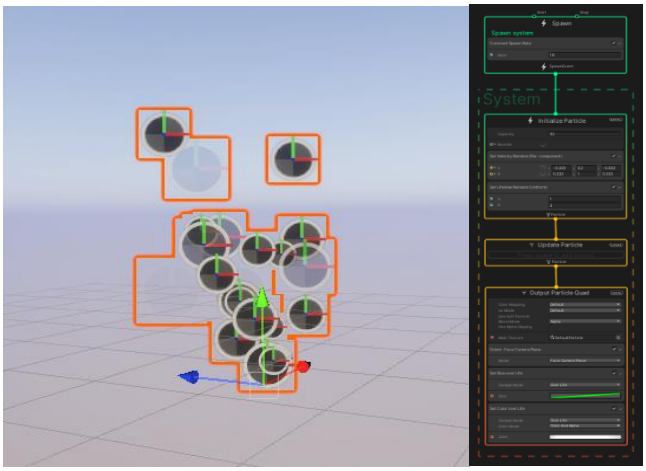

Fig. 4 - Visual Effect Graph in Unity (Screenshot Unity 2021)

In addition, Unity also offers the Visual Effect Graph which is an alternative to the standard particle System. It offers a node-based particle system as shown in Figure 4 with similar options but renders the particles over the graphics card (Unity Technologies, n.d.-d). This allows more particles with better performance but also makes the system harder to work with. The node-based system offers a variety of nodes and in order to figure out what each of them does, experimentation is needed.

4.4 VR Performance

According to the Oculus Developer Guide (*Guidelines for VR Performance Optimization | Oculus Developers*, n.d.) the big difference between PC Software and VR Software is that for the latter everything has to be rendered twice, one image for each eye. This means that the rendering of meshes and textures as well as real time lighting and shadows has to be well adjusted for good performance.

The Guide lists multiple possibilities on how to ensure or improve performance such as a limitation of triangles and vertices in the meshes of models, a limited script execution time, a limited number of textures, and simplified code. In other words, code should be efficient, and models should be game-ready; optimized so that they do not influence a game negatively.

Google's developer Guide (n.d.) suggests exchanging some high-cost effects with post processing and also references the Unity Profiler. The profiler evaluates Data while testing a game and gives detailed information about performance as can be seen in Figure 5.

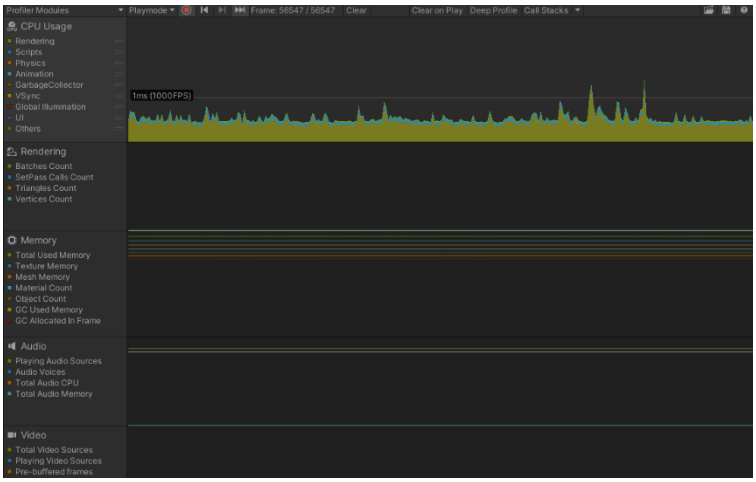

Fig. 5 - Unity profiler (Screenshot Unity 2021)

Performance, or in other words, a high frame rate is very important for any VR application in general.

According to IrisVR (*The Importance of Frame Rates*, n.d.), a framerate below 90 already interferes with the user's immersion and makes a VR experience less believable. A framerate below 60 can cause discomfort, nausea, and motion sickness. This shows that a good performance is of importance.

4.5 Motion Sickness

Motion sickness can be a typical side-effect of some VR Experiences and sometimes even daily life. Some people experience Motion Sickness during plane flights, car drives, or while on sea. Dan Brennan from WebMD (2016) states that motion sickness is caused by conflicts among senses. The body is unable to cope with mixed signals, which causes dizziness and sickness. Those mixed signals are present in many VR experiences, depending on the types of motion that a user can experience. If the user for example can experience flying like a bird with smooth motions, the eyes and the brain experience movement while the rest of the body does not. This disparity can be too much and while some people do not experience any side effects other get motion sickness.

Since many users have problems with motion sickness, it is of importance to take that into account and develop VR Software according to it. To prevent motion sickness, movement independent from the body can be completely disabled or teleportation can be used, so that the user disappears at one point and appears at another in the virtual world without a smooth movement.

4.6 High Definition Render Pipeline

The **H**igh **D**efinition **R**ender **P**ipeline (HDRP) is a Scriptable Render Pipeline. It makes use of physics-based lighting techniques, and allows users to create games, animations, and similar software with high visual fidelity.

According to the Unity Manual (n.d.-a), the HDRP has on its own many differences to the standard built-in pipeline. It for example allows:

- **H**igh **D**ynamic **R**ange **R**endering (**HDRR**):
  HDRR is a rendering algorithm which calculates lighting and 3D Space in a large dynamic range. The technique saves processing time, storage costs and waiting times (*Understanding High Dynamic Range Rendering - Steve's Digicams*, n.d.).
- Physically Based Sky:
  The physically based sky simulates an actual planet with a dynamic atmosphere and simulated air and aerosol particles (Unity Technologies, n.d.-e).
- Volumetric Lighting:
  Volumetric Lighting is a physical lighting model that manifests as visible light beams rather than flat lighting. It adds realism and depth (Sutton, 2020).
- Shader Graph as an alternative to scripted shaders:
  Shader Graph is a Unity plug-in that allows users to create a shader with nodes instead of code while also being able to see changes in real time. It is a complex tool with many options (Unity Technologies, n.d.-f).
- GPU Particles with the VFX Graph:
  As mentioned above, the VFX Graph a node-based particle system which renders the particles over the graphics card. It allows more particles than the standard Unity system.

5. Problem definition

The client wants to raise awareness about extreme weather and in particular the storm event in Borculo through a simulation. However, simulations need to be realistic, which is hard to achieve if you have limited memory and processing power due to the render technique of VR. The information gathered previously can be used as a guideline for a better and more educational simulation of the storm.

Subjects that have been researched in order to create the final product are:

- Which workflow influences the performance of the experience positively?
- How high can the visual fidelity be without negatively influencing the performance?
- Which type of particles need to be present for a realistic appearance?
- Which set up of effects offers the highest degree immersion?

6. Main and Sub Questions

In which way can realism in the VR experience be achieved in order to recreate the Borculo Storm Disaster in an immersive manner?

Sub-Questions:

Which elements of the visualization are most important when it comes to environment and particles?

Showing the impact and severity of the storm which hit Borculo is an important aspect of the project's realism. While destruction of buildings can show the intensity of the storm, in order to really show the effects of it, debris and objects can be added as well, such as flying signs, ripped flags, flowerpots, leaves, dirt, and similar debris. It needs to be evaluated which objects were influenced by the storm due to their construction, weight, and size, and which were not.

In order to define the objects that could enhance the experience, experiments with wind were considered, as well as desk research with a focus on the Fujita Scale because it also describes which stages of storm have an impact on objects of different sizes and weights. That way it became clear which objects effectively show the impact of the storm.

How does the weather affect lighting in the environment?

Especially in extreme weather situations, the sky can get very dark and the brightness of for example daytime is influenced immensely. For the VR experience, the storm might be simulated from start to end so that the light can change drastically in the scene.

In order to show that realistically, research was needed to be research to figure out exact day times and how much influence the upcoming storm had on the brightness. Therefore, videos were considered, as well as photography and articles about similar events.

Which workflow for realism is more efficient in terms of rendering performance?

To convey realism, interactive elements such as moving leaves, falling trees, collapsing houses and such are of importance. Adding those elements will add realism to the experience but they can also affect the framerate in a negative way, which is, especially for virtual reality, problematic.

To avoid a low framerate that makes the experience unusable, testing is important. Each approach regarding particles and animations was tested in VR and monitored in Unity to figure out what works best.

7. Development

The following contains relevant Information about the assignment and the work that had to be done. The project started as an Unreal project. Therefore, it is necessary to analyse the previous files so that problems can be defined. Another aspect that needs attention is the engine choice that affected the workflow. The decision was made to switch from Unreal to Unity so that a simpler workflow was possible without learning the use of a new engine. Finally, the focus will lay on the creation of particles, lighting, and foliage animations. These elements are most important for the immersive and realistic visualization of a storm.

7.1 The Unreal project

The Borculo Storm project started as an internship project for a group of students at the XR-lab of Saxion. Therefore, work has already been done in the past. The files delivered contained a historically accurate designed scene with a light setup and particles. The scene itself displayed a church square, a church, and houses surrounding it as well as low detail props in the background, basic props, and some foliage.

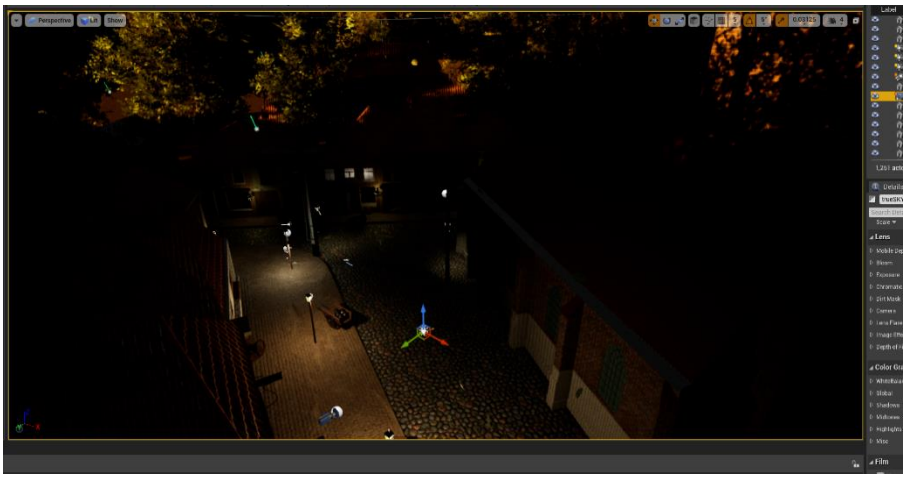

Fig. 6 - Borculo Storm Project (Unreal 2021)

Some problems of the old project were clear from the start. Especially the particle system did not convey a heavy storm. It did not contain enough particle types and particles in general. While the system looked good on screen, in VR the particles were at least ten times as big as they needed to be.

According to eyewitnesses (*Meteo Consult Weernieuws Archief*, 2005) the storm started to develop at half past six in the evening with lightning and thunder coming from the northwest. At seven the storm arrived, and it was raining heavily, everything was flying horizontally. Debris, dirt, rain, fog, and leaves were carried away by the storm.

Fig. 7 – Tropical Storm Harvey (Ralston, 2017)

The old project came with a leaf particle system that was moving rapidly, but not fast enough according to the description of eyewitnesses as it was not moving fast enough to fly horizontally. It was also lacking all types of particles such as small debris, fog, and rain. Only leaves were present.

During research, the lighting became a problem as well. The lighting in the provided scene was too dark. The storm started in the evening in august. At that time of the year, the sun starts to set after 9 o'clock in the evening which means, during the storm it was still as bright as day, not considering the clouds covering the sky (*Sunrise Sunset | Borculo*, n.d.). Even with clouds covering the sky at daytime, the atmospheric lighting never turns too dark to see, due to scattering and reflection of light. The provided scene came almost without daylight. Everything except for areas around streetlamps was pitch black and hard to see.

Considering the results of the analysis, the tasks for the graduation assignment are clear. It is of importance to develop a weather system that provides correct lighting, particles, and wind speed.

7.2 Choosing Unity and setting up the scene.

A few weeks into the project, the decision was made to switch the game engine, from Unreal to Unity. This decision was made by the whole project group with the intention to have an easier workflow. Most group members are familiar with Unity but not Unreal and had complex tasks that did not allow to also learn a new and complex workflow for another game engine. The choice of engine also made it easier to set up the necessary visual elements.

In order to allow good and realistic visuals, the decision was made to use the high-definition render pipeline. It is concepted for good lighting and realistic graphics, a multitude of post processing effects and runs well on modern hardware (Unity Technologies, 2020).

One of the first steps for this project was to get the content from the old project files into Unity. Not everything could be imported, some aspects needed to be recreated. In Unreal the trees were animated by a shader. Unity also comes with shaders for foliage and trees and even a built-in tree creator. However, neither of those features is compatible with the HDRP. For the project it was important to compensate those features because animated trees are necessary to visualize a storm.

7.2.1 VR-ready foliage

To set up the project properly, trees were needed. Because the tree from the previous project created a repetitive look and was too large in size, the SpeedTree Software was chosen. SpeedTree, which also is available as a plugin for Unity, allows users to create foliage in every shape and form. It is not only available as a plugin but also as an external software. Due to the compatibility issues, the external software was used. The Unity plugin uses a specific shader set up which is not compatible with the HDRP and as such did not allow to create good looking foliage without a time-consuming workaround (*The State of the SpeedTree/Unity HDRP Pipeline – SpeedTree*, n.d.).

Speed Tree is available as a software specifically made for games and as a software for movies (*SpeedTree – 3D Vegetation Modeling and Middleware*, n.d.). Both versions have advantages and disadvantages. While SpeedTree Cinema allows to create trees with animations for wind, growth and different seasons, the trees created have a high polycount and are not game ready. In comparison, SpeedTree Games creates game ready trees with an automatically created level of detail. However, this version does not allow to create realistic animations, but it also creates a rig for each tree so that the tree can be animated by hand if necessary.

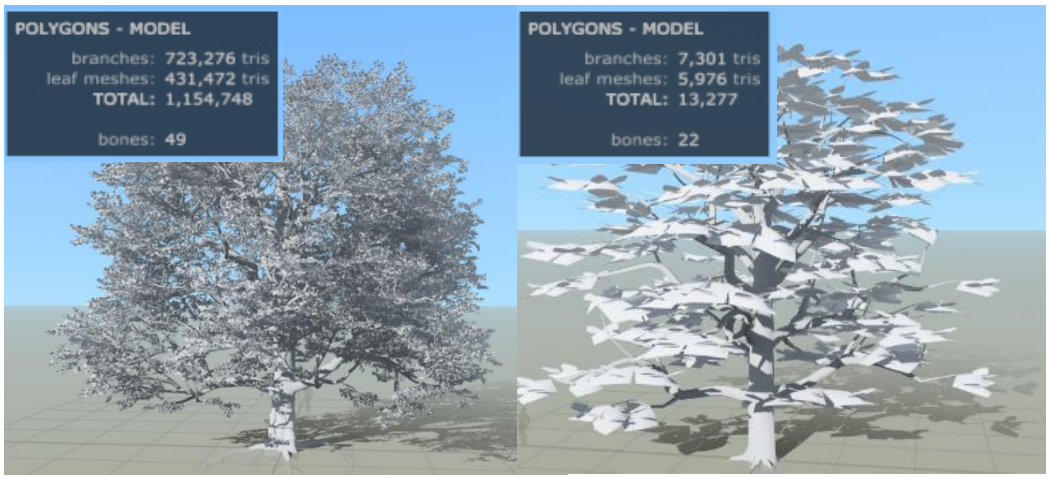

Fig. 8 - Comparison of Trees created with SpeedTree Cinema and SpeedTree Games (Screenshot SpeedTree 2021)

While experimenting with both software versions, the decision was made to use SpeedTree Games. Even though the cinematic version allows a lot more realism, for Virtual Reality projects such as this one, game ready assets are important to keep a good frame rate to ensure the user's wellbeing.

When trying to find a good workflow to animate the tree, it was possible to switch between the two SpeedTree versions. The low poly tree could be animated in SpeedTree Cinema which first appeared to be a good solution. However, after importing the tree into Unity the animation appeared to be gone. While looking into the animations created by SpeedTree, it became clear that Unity is unable to show those animations.

SpeedTree does not create animations by rigging an object and creating keyframes. It uses point cache animations. A point cache is a cache of raw data. It usually stores various simulation data such as cloth or physics simulations. Many 3D Programs such as Maya, Blender and 3DsMax can read point cache data but Unity cannot.

After spending time trying figure out how to read point cache data in Unity, the decision was made to use a shader or handmade keyframe animations instead.

7.2.2 Shader approach for wind animations

The quickest approach to animate the trees for the prototype was to create a foliage shader. To achieve that, Shader Graph was used as it offers, as mentioned before, the easiest workflow. To gather information about how to work with Shader Graph, especially for foliage, the Boat Attack Tutorial from Unity was of great help (Unity Technologies, 2020b). The purpose and use of different nodes were explained and aided in the creation of a shader that simulates different strength of wind on a tree.

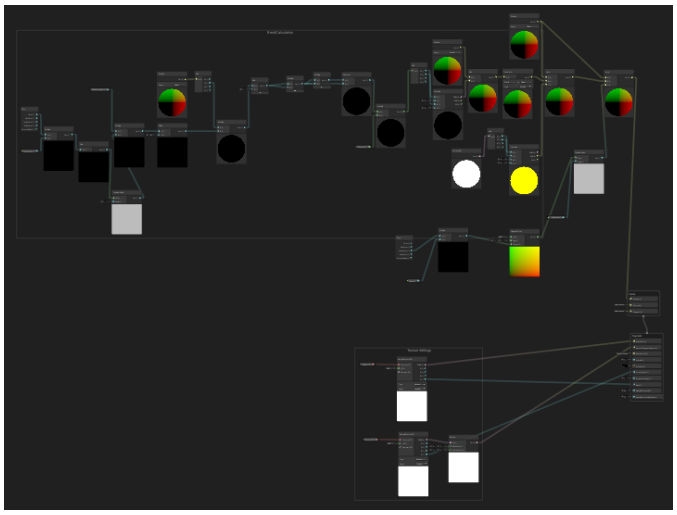

Fig. 9 - Unity Shader Graph (Screenshot Unity 2021)

The Shader uses Vertex displacement and gradient noises to simulate wind. The displacement direction can be set so that the animation can be adjusted to every direction that wind comes from. Next to the direction, noise scale, speed, strength, and bend can be adjusted. Using vertex paint the intensity of movement in the tree can be influenced.

Fig. 10 - Iterations of Tree-Shader in Unity (Screenshot Unity 2021)

Finding the correct shader values and settings was challenging so that multiple iterations of the shader were created, as seen in Figure 10, until the appearance was correct.

7.2.3 Keyframe approach for wind animations

Next to creating a shader, keyframe animations were made to symbolize heavy wind colliding with foliage. A reason for considering the workflow is the high degree of freedom. When rigged and weight painted, every branch of a tree can be moved and that way wind directions as well as wind strength can be shown. Moreover, the animation system of unity consists of Blend Trees (Unity Technologies, n.d.-g) which allow to blend smoothly from one to another animation.

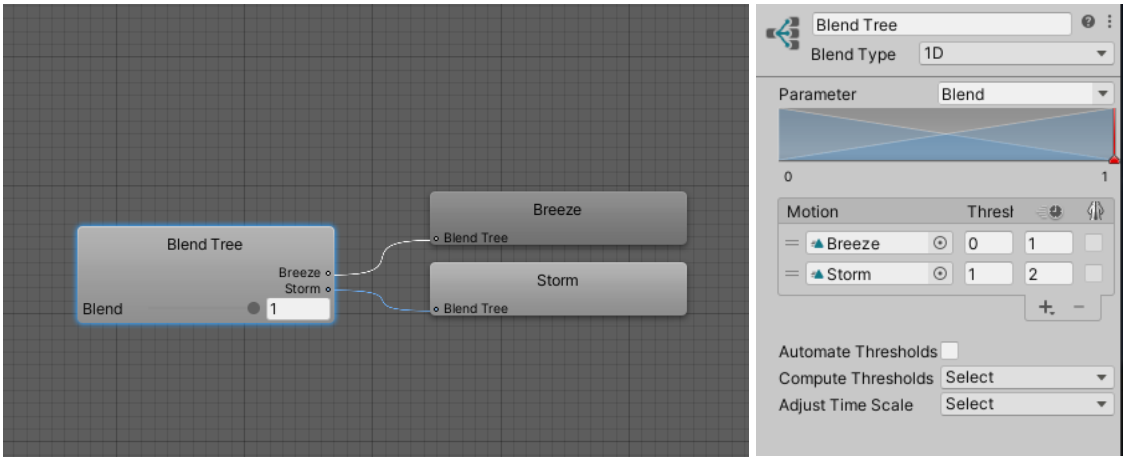

Fig. 11 – Blend Tree (Screenshot Unity 2021)

Using Blend Trees, the gradual development of the storm can be visualized in a realistic manner by increasing a value connected to the blend function as seen in figure 11.

To allow a realistic visualization, two animations were created with the Animations System in Unity as seen in figure 12. One animation for a small breeze with lightly moving branches and a second animation for the heavy storm with heavily shaking branches and leaves. Later a blend tree was created to gradually blend from the first to the second.

Fig. 12 – Tree Animation for Breeze and Storm (Screenshot Unity 2021)

7.2.4 Standard Particles

Particles were the next important aspect of visualizing the storm. The priority at the start was to recreate what already had been done in Unreal, but in Unity. Therefore, a basic particle system was created. As stated previously, besides flying leaves, all aspects of a storm particle system were missing so they were added at this point.

Leaves

When creating the leaf particles, the first focus was to make sure that the scaling is correct, in relation to the player and trees. This was achieved by changing the scaling multiple times until it appeared to fit on screen as well as in VR.

Fig. 13 – Leaf Particles (Screenshot Unity 2021)

In order to simulate and tweak the leaf movement correctly, visual research helped. While videos were evaluated, it became clear that leaves do not only rotate and flip while falling but also do not fall down straight and not with the same speed depending on the shape and size of the individual leaf falling (Relax Sleep ASMR, 2019). To achieve that visual, random velocity and rotation over lifetime was added to the particles as well as varying gravity modifiers. To enable movement based on physics, particle speed was completely disabled.

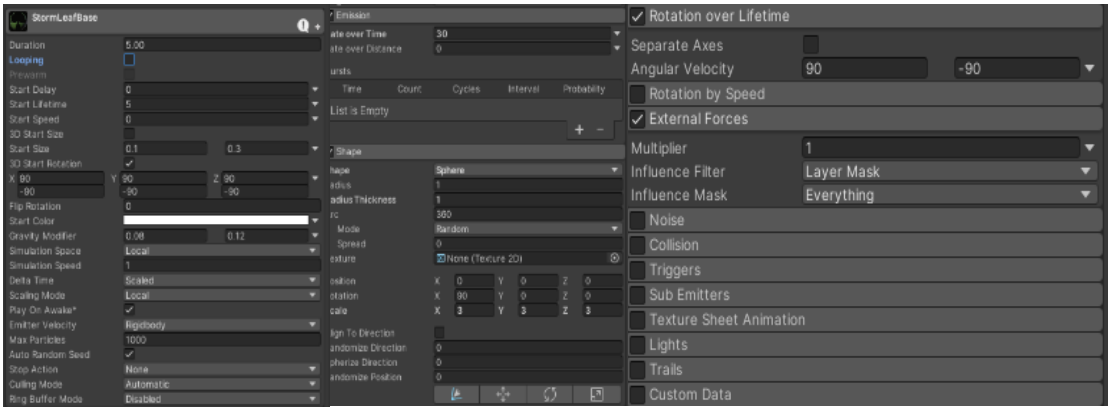

Fig. 14 – Leaf Particle Setup (Screenshot Unity 2021)

Rain

The challenge when creating rain particles was to make them move according to the strength of wind. When rain is observed, every raindrop appears to have an elongated shape due to the speed that it is falling with and not all raindrops are falling into the exact same direction but are influenced by turbulences when wind comes into play (AMANDA, 2020). When wind gets very strong, the raindrops fly almost horizontally. In the beginning, when the strength of the wind zone was increased, the rain particles did not rotate into the correct position but stayed in a vertical position (Figure 15).

Fig. 15 – Wind Strength 70; Render Alignment World (Screenshot Unity 2021)

The issue could be solved by changing the render alignment from “World” to “Velocity”. That way the particles rotate towards their moving direction (Figure 16).

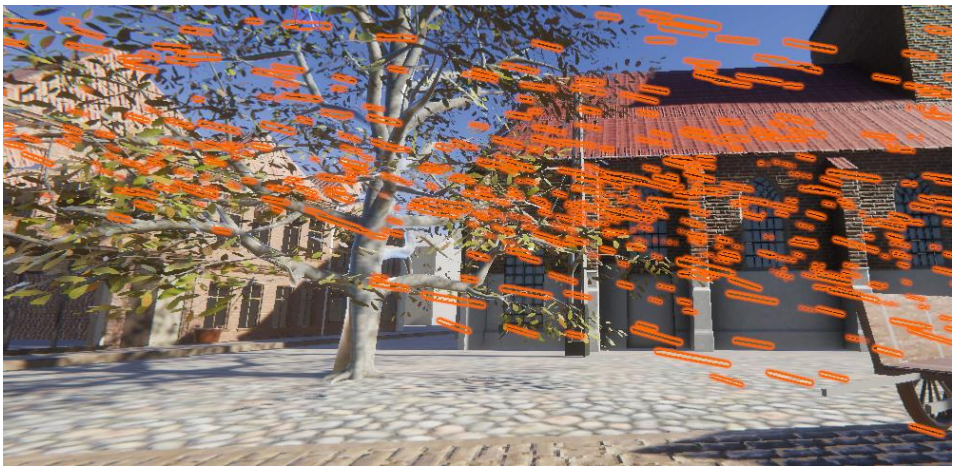

Fig. 16 – Wind Strength 70; Render Alignment Velocity (Screenshot Unity 2021)

At this point the rain particles fall according to the strength of wind and react based on physics rather than settings. This allows a lot more variety when it comes to wind strength and wind direction.

Dirt

The dirt particles are a supporting element to add more depth into the visualisation. Visual research showed that the atmosphere during a heavy storm appears to be very foggy with some grain to it. The dirt particles were created with the intention to replicate that.

Fig. 17 – Examples for grainy Fog (The Guardian 2015; Newsweek 2017)

In order to simulate the appearance, the particle system creates billboard particles with a grainy texture that also move based on physics and external forces. The billboards rotate randomly to minimize the risk of a repetitive appearance.

Fig. 18 – Dirt Particles in Scene (Screenshot Unity 2021)

The decision was made to use texture sheets because that way the number of particles could be decreased. The visual effect was not impacted negatively due to the speed of the particles when the wind starts. Thus, a good performance could be ensured.

Fog

Creating the fog was a challenge as well. As presented before, a heavy Storm appears to create dense fog by moving many particles at a high speed. When rain is involved, those particles create an appearance similar to mist.

When the first prototype was created, a particle system was used which was problematic from the start. Using a billboard renderer was the only possibility to get a semi volumetric appearance with the standard particle system but depending on the camera angle the particles cut into buildings and created visual edges that disrupted the appearance rather than supporting it.

Fig. 19 – Billboards cutting into walls and roofs of houses. (Screenshot Unity 2021)

Decreasing the size of the billboards could have prevented that. However, decreasing the size meant to also increase the number of particles to keep the foggy appearance which in the end would cost performance while not improving the visuals enough.

The conclusion was to try a different approach in order to create the desired look. Instead of using particles, the decision was made to use Volumetric together with standard lighting settings from the HDRP.

Fig. 20 – Particles and Fog combined. (Screenshot Unity 2021)

7.2.5 VFX Graph particles

As mentioned in previous chapters, VFX graph is an alternative to the standard Unity particle System for the HDRP. As such it creates particle effects with a node-based system. To evaluate which particle System creates the best looks while at the same time offers best performance, the VFX Graph was a second approach for the project. Using the graph, three particle systems were created: leaves, rain, and dirt.

Leaves

Fig. 21 – VFX Leaf Particles. (Screenshot Unity 2021)

The leaf particles created with the VFX shader are emitted in a five-by-five-meter radius (Figure 22). This was achieved by using the Set Position Random (Per-Component) node. Hereby each the position of each particle is set randomly within the set bounds.

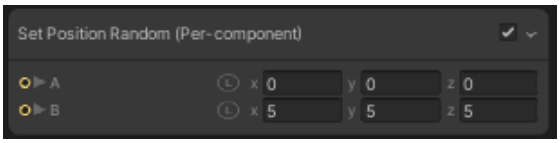

Fig. 22 – Set Position Random node for leaves. (Screenshot Unity 2021)

The Set Angle.XYZ Random (Figure 23) node then also rotates the leaves in a random position to make them look more natural. Leaves grow in many directions on a tree, so it is important to make a leave fall not only from different positions but also in different angles.

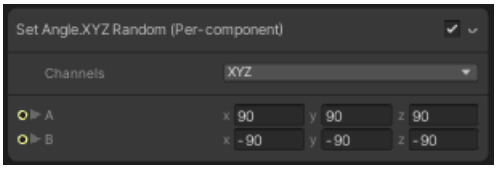

Fig. 23 – Set Angle.XYZ node for leaves. (Screenshot Unity 2021)

Because leaves do not fly straight but have a rotating motion while falling or being carried away by wind, in the Update Particle Section Add Angle.XYZ was added (Figure 24). The node adds 0.3 degree of rotation towards every angle of each particle to simulate the swirling and rotating during the particle's lifetime.

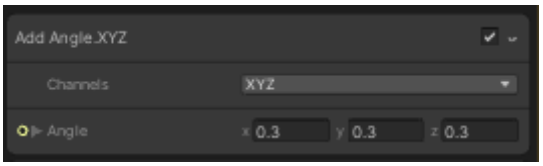

Fig. 24 – Add Angle.XYZ node for leaves. (Screenshot Unity 2021)

Next turbulence was added (Figure 25). Turbulence influences every particle's velocity with a noise pattern and is thus able to simulate natural movement. The type of noise can be set as well as position, angle, and scale. The intensity value defines by which factor the velocity is multiplied and frequency influences the sample period. Octaves allow a more varied look by layering the noises while also being resource intense, and roughness sets the octaves scale. Lastly Lacunarity defines the rate in which the frequency of an octave is changed.

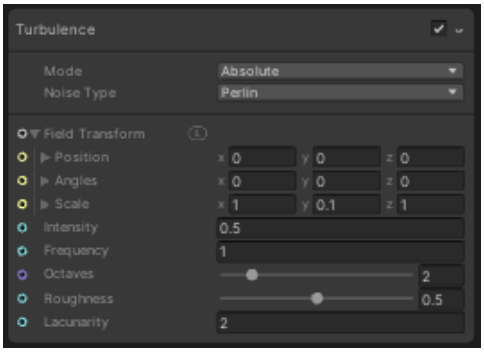

Fig. 25– Turbulence node for leaves. (Screenshot Unity 2021)

Standard Velocity was added as well (Figure 26). The VFX Graph is not compatible with wind zones. To compensate that, the wind strength is added as velocity to the x axis of the particle emitter. By using an exposed float, it is possible to change the wind strength and look of the particles with the script. This technique was used for all VFX particle effects.

Fig. 26– Velocity setup for all particle system created. (Screenshot Unity 2021)

Lastly, the orientation of all particles is rotated towards a position in the scene to make them independent from the camera and allow the leaves to appear realistic (Figure 27). By default, the orientation of the particles is set towards the position of the camera. This would cause all leaves to rotate dependent to the camera and thus break immersion.

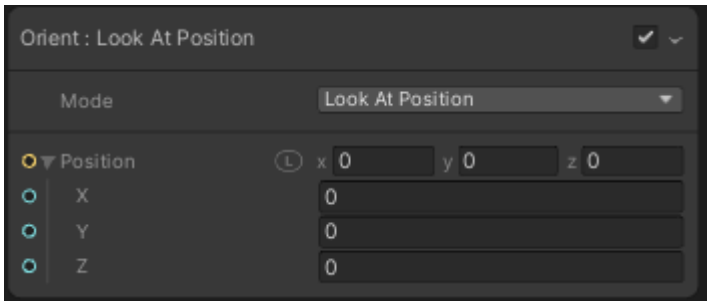

Fig. 27– Look At Position node for leaves. (Screenshot Unity 2021)

Rain

Fig. 28 – VFX Rain Particles at run time and highlighted (Screenshot Unity 2021)

During the initialisation of the rain particles, the velocity is set to a random value between -24 and -30 on the Y axis (Figure 29). Because the VFX Graph in comparison with the standard particle system does not have a gravity option, velocity is used to compensate that. Doing so, the values have to be negative so that the rain falls downwards and not upwards. The velocity is set to a random value because not all rain drops fall at the same speed since they also do not have the same size.

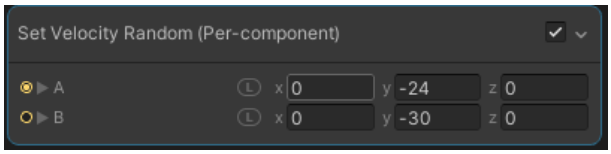

Fig. 29 – Set Velocity Random node for rain. (Screenshot Unity 2021)

Other than for the emission of the leaves the rain particles are emitted by a hundred-by hundred-meter plane (Figure 30). This was done to simulate a large area of effect and to eliminate the need to place multiple particle effects in the scene.

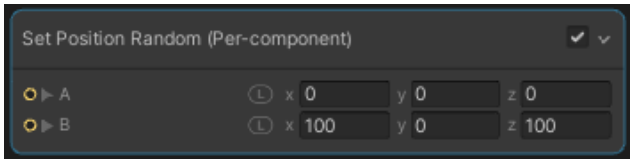

Fig. 30 – Set Position Random node for rain. (Screenshot Unity 2021)

Setting the position of each rain particle was an approach to achieve a natural appearance and overall size for each raindrop (Figure 31). A similar technique was used for the standard particle effect. While the particles on the X and Z axis are set to a minimum width and depth of 0.1 and 0.2, the length on the Y axis was set to a value between one and ten. That way the particles are always comparably thin but vary a lot in length just as with real rain.

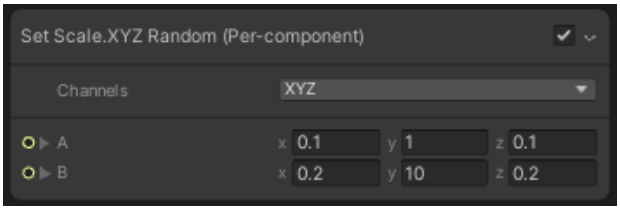

Fig. 31 – Set Scale.XYZ Random node for rain. (Screenshot Unity 2021)

The rain particles required some turbulence as well because with their travel distance rain drops often fall in slightly varying directions (Figure 32). However, compared to the leaf particle system the intensity of the influence on velocity was much higher so that a visible difference could be achieved in combination with the velocity that was also higher for the rain particles.

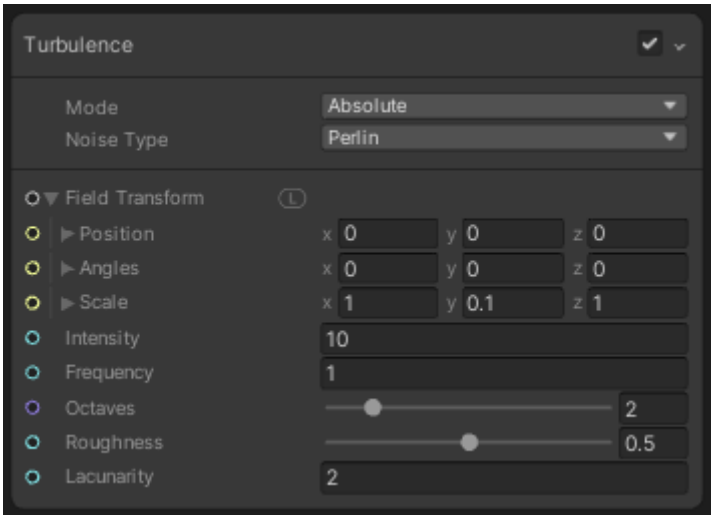

Fig. 32 – Turbulence node for rain. (Screenshot Unity 2021)

Similar to the rain system that was created with the standard particle system, the render alignment was given a dependency to the velocity of each particle (Figure 33). That way, the VFX rain particles also turn into the direction they are moving and have a natural appearance.

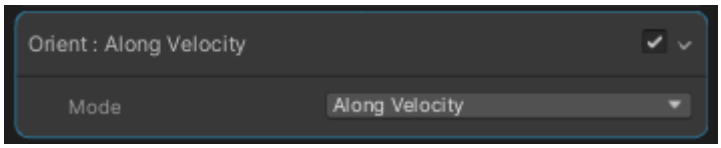

Fig. 33 - Orient : Along Velocity node for rain. (Screenshot Unity 2021)

Dirt

Fig. 34 - VFX Rain Particles at run time and highlighted (Screenshot Unity 2021)

Different than for the dirt particle system that was created previously, the system created with VFX graph does not render the dirt particles on a big plain with a sprite but each clump of dirt individually. This was

possible because of the different render technique that is a lot less resource intense. The dirt is shown by a total of 8000 individual particles that were defined in the spawn section of graph (Figure 35).

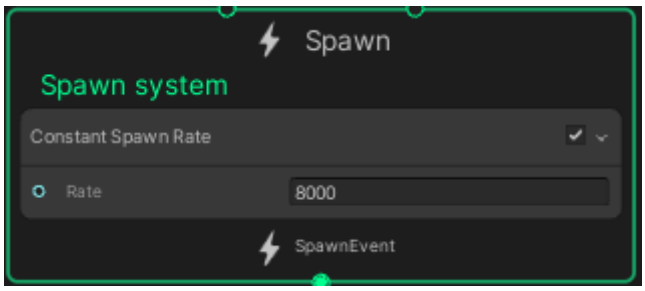

Fig. 35 – High spawn rate of dirt particles (Screenshot Unity 2021)

A difference from the other two VFX graphs is that the dirt particles spawn without an inherited velocity. The reason for this is that the dirt particles only appear when the wind gets very strong. In the beginning, no dirt particles should fly through the scene because there is no wind that picks them up. However, the particles are influenced by turbulence for the same reason as previously mentioned: More natural movement.

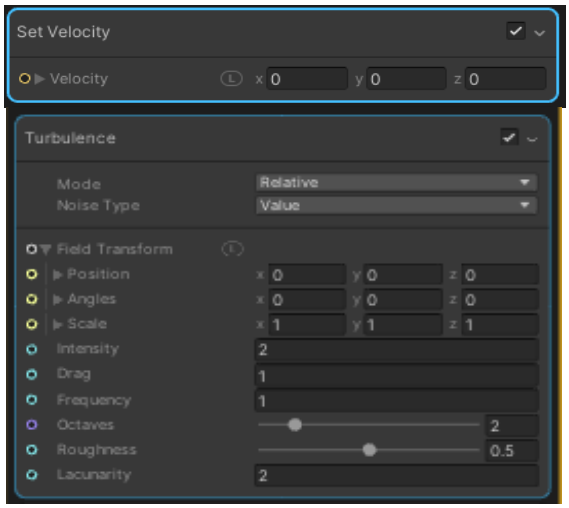

Fig. 36 – Set velocity and Turbulence node for dirt (Screenshot Unity 2021)

Similar to the rain system, the dirt particles also have a random scale between two values (Figure 37). Thus, every particle has a different shape and as such as much diversity as possible.

Fig. 37 – Set Scale.XYZ node for dirt (Screenshot Unity 2021)

Another difference to the other particle system is that a sphere was used as an emission shape (Figure 38). So instead of spawning from a loose range, the particles can spawn on top of or within a sphere. This shape was chosen because it was more convenient to hide inside of props so that the particles can move from the source over the screen depending on the wind strength.

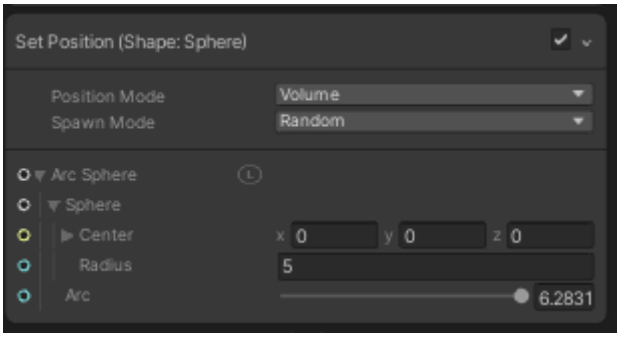

Fig. 38 – Set Position Sphere node for dirt (Screenshot Unity 2021)

7.2.6 Weather System

To accommodate the client's wishes, it was of importance to not only show a storm but also the origins of it. To achieve that, a Weather System was programmed which is able to control all variables of the storm simulation and allows the recreation of different type and strengths of the storm. It also offers the possibility of changing values as dynamic as possible so that the system can be adjusted according to results of research.

In order to convey the origins of a storm, different variables were needed because not all effects of a storm start at the same time and strength. Examples for those variables can be seen in figure 39.

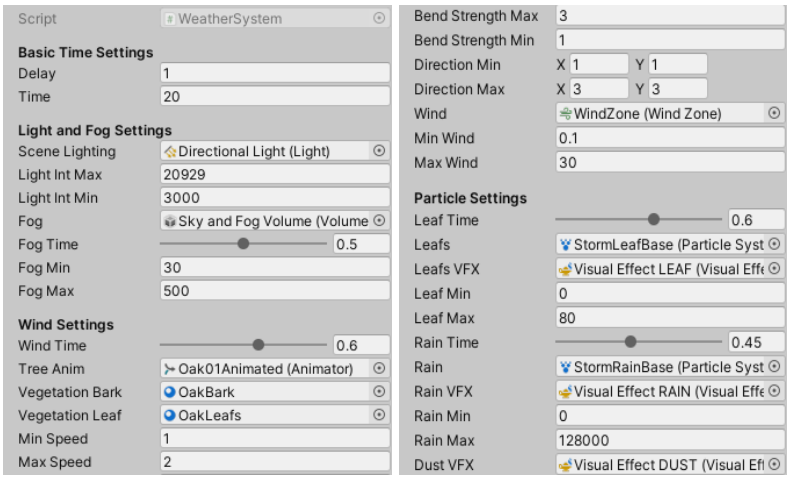

Fig. 39 – Weather System (Screenshot Unity 2021)

The time in which the storm should take place needed a dynamic and interchangeable value because the experience is meant for a museum. Visitors of a store, park or museum do not want to spend a long time in a queue to wait for something, in that case the VR experience, the duration of the experience needs to be fully adjustable to accommodate that. The average person accepts to wait between 5 and 10 minutes in a queue or a bit more if the perceived waiting time is reduced (Strange, 2012). To set the time precisely, the Delay and Time variable was implemented. While Delay sets a timespan from the start of the experience until the weather starts, the Time variable defines the exact length from when the storm starts until it reaches its peak.

The different effects creating the visuals needed to be dynamic too. Values were needed to change their starting time as well as intensity. Especially the particles should not be at the same strength throughout the whole experience. Rain starts with a few drops und gets heavier over time. Leaves are not ripped from a green tree during a soft summer breeze. Also, light and fog had to interchange over time. Therefore, minimum and maximum value variables were implemented such as:

- lightIntMin/lightIntMax which were connected to the light intensity of the Directional light
- lightTempMin/lightTempMax which were connected to the temperature of the Directional light
- fogMin/fogMax which were connected to the sky and fog volume
- min/maxSpeed, bendStrengthMin/max and directionMin/max which were connected to the foliage shader
- windMin/windMax which was connected to the Wind Zone as well as the Blend Tree of the animated tree
- leafMin/leafMax and rainMin/rainMax which were connected to the specific standard particle systems as well as the VXF particle systems.

To set at which time each of the effects should start to change from their respective minimum value to a maximum value, variables controlled by a slider for a starting point were implemented. The slider defines at which percentage of the overall time during which the weather changes the respective effect should start to blend in. Those variables are:

- fogTime which defines the starting point of the fog increasing in density
- windTime which defines the starting point of the wind increasing in strength
- leafTime which defines the starting point of leaves increasing in number
- rainTime which defines the starting point of rain increasing in strength

The sliders worked with values between 0 and 1 so that their value could be used with a Lerp function. The Lerp function linearly interpolates a value between a and b by t while t is clamped to a value between 0 and 1. By gradually increasing the value of t from 0 to 1 a smooth blend from value a to value b is achieved (Unity Technologies, n.d.-h).

8. Testing

In order to distinct which visualization of the storm appears most realistic to users, a survey was created (Appendix A). The Questionnaire was meant to show how to set up the weather system in order to make users believe that they are in the middle of a massive storm.

For the survey, seven video samples were created with different particle and lighting settings as well as different versions of particles and trees. Each sample offered three or four statements that had to be rated by using a Likert Scale. The Likert scale offers better evaluation of surveys because it does not expect a yes/no answer but more an opinion about something (McLeod, 2019). Users read a statement and rate the statement according to their own opinion

In order to prevent bias, the videos were not ordered chronological, and the statements did not carry any opinion-based formulation.

Participants were found by sharing the survey on Discord servers that were attended by many people of different ages and genders. A total of 17 answers were obtained over a cause of one week.

From the answers gathered, it became clear that the majority of participants found the most intense samples most convincing. While the heaviest rain created with the VFX shader was most popular, the leaves created with the standard particle system were liked more. Overall, the participants thought that more leaves and debris from the trees should be visualized. Focusing on fog, most participants thought that the strongest sample regarding grain and density was too heavy while the second strongest appeared to be fitting. Lastly, the darkest light settings were found to be most convincing while both versions of trees did score similar points. The average score for each statement can be seen in figure 40.

Rain emissio	Leaf emission	Fog softness	Light intesity	Sample	YT/Surves order	Rain	Leaf	Fog	Light
2000	20	100	9000	1	Sample004	more rain needed	more leafs needed	denser fog needed	too light
4000	30	80	8000	2	Sample002	more rain needed	more leafs needed	denser fog needed	too light
8000	40	70	7000	3	Sample006	more rain needed	more leafs needed	denser fog needed	too light
16000	50	60	6000	4	Sample007	more rain needed	unsure	denser fog needed	too light
32000	60	50	5000	5	Sample001	amount seems okay	enough leafs	fog seems okay	light seems okay
64000	70	40	4000	6	Sample003	amount seems okay	unsure	enough fog	light seems okay
128000	80	30	3000	7	Sample005	enough rain	unsure	fog is too dense	dark enough
						Best was 7	Best was 5	Best was 6	Best was 7

Fig. 40 – Survey Answers average.

According to the answers, the experience was adjusted with a focus on each individual particle effect and the influence on performance. For testing each particle setup was placed individually in the scene, without any other effects present.

Leaves

The leaf particle systems were both tested with 60 particles as it resulted from the survey. The difference in performance was significant, the VFX System saved a total of 40 FPS as it can be seen in figure 41. Even though the standard particle shader was liked more by participants of the survey, the difference shows that it makes much more sense to use the VFX version and adjust the movement to accommodate good visuals and better performance. The decision was made to use the VFX graph leaves.

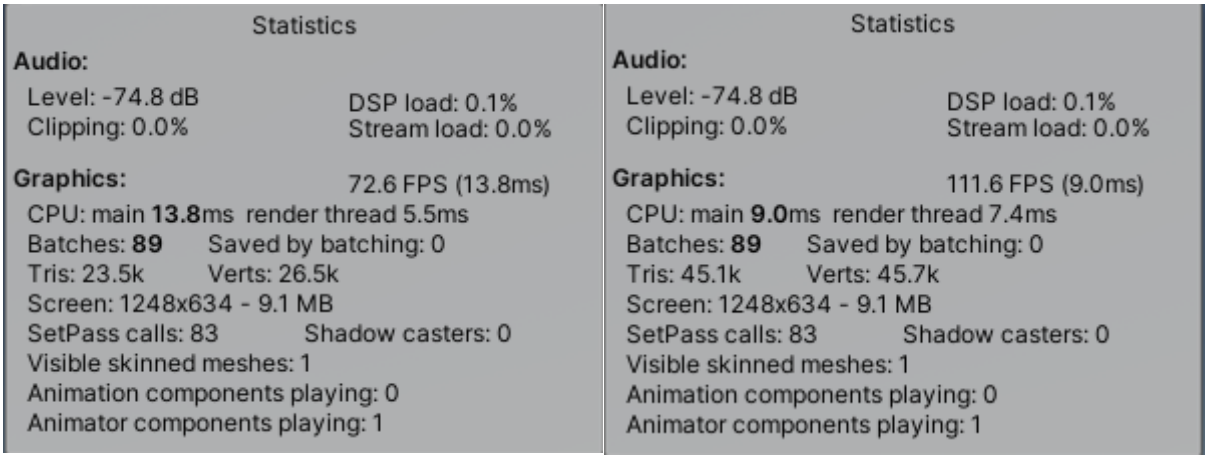

Fig. 41 – Comparison of FPS changes for standard leaf particles and VFX leaf particles (Screenshot Unity 2021)

Rain

The rain system was tested with the number of particles resulting from the survey as well; 128000. The difference was even more significant. While the experience was no longer playable and hit a rate of 3 FPS when the standard particle system was used, the VFX particle system allowed the simulation to still run at 60 FPS as shown in figure 42. While 60 FPS still was an acceptable rate, the decision was made to lower the number of rain particles a bit to improve performance further and also enable Frustum culling for the particle system so that particles which are not in the user's view will not be rendered (Gerlits, 2017). Again, the VFX particles were chosen.

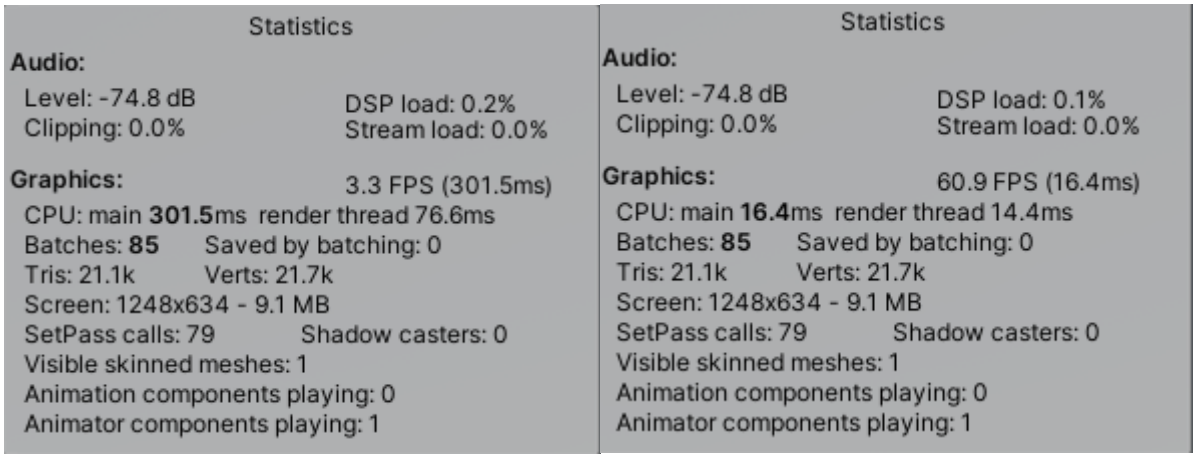

Fig. 42 - Comparison of FPS changes for standard rain particles and VFX rain particles (Screenshot Unity 2021)

Fog

The dirt and grain effect for the foggy atmosphere was tested with an emission of 40 particles for the standard system and 8000 particles for the VFX system. Again, the VFX system scored a lot better compared to the standard particle shader when it comes to performance even though the number of particles in the scene was significantly higher as presented by figure 43. The VFX system again was chosen, and the visuals tweaked to accommodate the results of the survey.

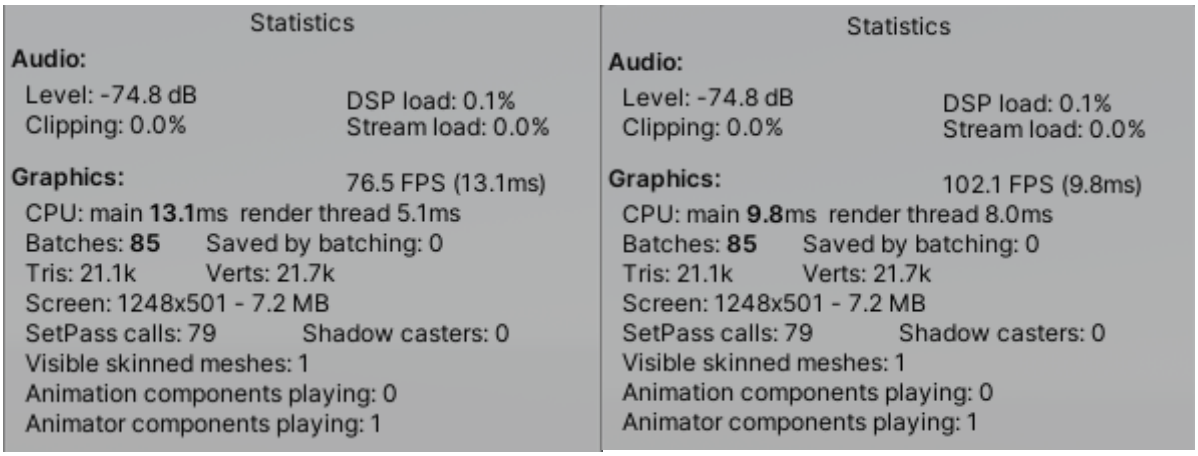

Fig. 43 - Comparison of FPS changes for standard grain particles and VFX grain particles (Screenshot Unity 2021)

VR Test

In order to test and verify the adjusted setup, a build was created and shared with VR-Users who have access to a VR-Headset in a safe environment which accommodates the hygienic measurements during the Covid Pandemic. In total, three participants were found. Because of the small number of participants, the survey was not representative, but it gave an idea about what could be further improved.

The participants received a build of the simulation and the recent adjustments and a survey link (Appendix B) in which they could list improvements to the different particle effects and lighting. The survey was formulated as a request for help and a demand to at least fill in one improvement was added as well. That way it could be avoided that out of sympathy no improvements were suggested or that the participant did not analyse enough and think about what could be improved.

While no one had anything to complain about the rain particles, suggestions were made for leaves, fog, and lighting. All participants thought that the lighting had to be darker because in VR space the experience appeared to be lighter then on screen. One participant suggested that twigs and branches could be added to the leaf particles and two mentioned that the fog density could be increased by a little. Another suggestion was to add more dirt to the particle system.

As a result, the experience was adjusted a second time and branches were added to the particle system as shown in figure 44.

Fig. 44 – Visualization before and after adjustments (Screenshot Unity 2021)

The minimum lighting intensity in the scene was decreased to half of the strength, from an intensity of 3000 Lux to 1500 Lux. The fog density was increased by lowering the attenuation distance from a minimum of 40 to 15. The amount of dirt particles was increased from 8000 to 10000 and the sizing was randomized so that a few particles are better to see while others are more subtle.

A branch mesh was created using Speed Tree so that branches in the scene could be visualized (Figure 45a). The mesh was used to create a new particle system in the VFX graph. The particle system then was incorporated into the weather system. The branch particle system itself was based upon the leaf particles but had one major difference: Instead of a 2D output, the VFX graph used a Lit Mesh output. That way a particle system with meshes and physic-based lighting could be created (Figure 45b).

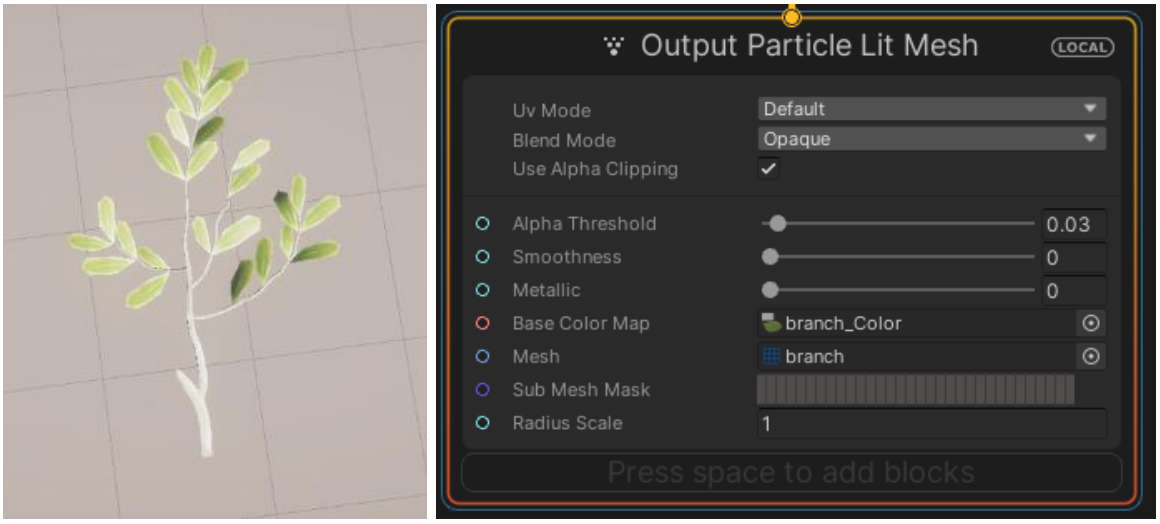

Fig. 45 – 3D Branch and Lit Mesh Output (Screenshot Unity 2021)

9. Results

The knowledge required to solve the client's problem was gathered by answering the sub questions. Elements of visualization, effect on lighting and rendering performance were important topics when it comes to answering the main research subject.

1. **Which elements of visualization are most important when it comes to environment and particles?**

This question was answered by doing research on the type of storm that hit Borculo in 1925. It was a stage 4 storm according to the Fujita scale and was likely a downburst or a large tornado. With that information it was possible to find images and video material of storms with a similar strength. Those materials gave an idea of what needed to be visualized: rain, dirt, debris from trees as well as interactive foliage and a foggy atmosphere.

The particle elements were realized with two types of particle effects as well as the volume settings of the HDRP and then used for the creation of multiple videos with different particle settings. A survey was created in order to distinct the best setup for the effects. The changes made to the prototype based on the answers collected by the survey ensured that the visualization is linked to the appearance of an actual storm. To create prototype foliage, a tree was created in SpeedTree and animated twice, by using a shader and keyframes. Participants of the surveys found that both trees helped the visualization even though the animation quality could be improved by a professional animator.

The final version of the visualization contains more adjustments focusing on the final VR perspective such as more fog, and more debris from trees.

2. How does the weather affect lighting in the environment?

While analysing the prototype in Unreal it became clear that the effects of the extreme weather on environmental lighting was not visualized properly. The scene had no environmental lighting. While researching the time of day in combination with the month in which the disaster took place it was discovered that it was still light as day when the storm happened. Due to scattering and reflection some light still lights up the scene. Light settings were included into the prototype which changed throughout the simulated storm development and were included in the videos for the survey. The survey participants found the darkest light to be more realistic which was then applied to the prototype. During the VR test, it was suggested to decrease the environmental lighting more because it appeared lighter in virtual space then on screen.

The final version consists of lighting with the strength of 1500 Lux at the peak of the storm.

3. Which workflow for realism is efficient in terms of rendering performance?

To find an efficient workflow for good VR performance, research was conducted focusing on the setup of the VR application as well as particles. For the VR setup the render technique was changed from multi pass to single pass instancing as recommended by the VR guide.

After the first survey, the decision was made to use VFX particles because the performance costs were lower, even with a large number of particles. While the standard particle system only ran at 4 FPS with 128000 particles, the VFX graph still allowed 60 FPS. To further improve the final performance setup, occlusion culling was enabled for the particle system as well.

In which way can realism in the VR experience be achieved in order to recreate the Borculo Storm Disaster in an immersive manner?

The main question was answered by researching and solving the sub questions. A realistic and immersive storm can be achieved by visualizing elements of an actual storm and by meeting the user's expectations of extreme weather. For immersion in a virtual space, performance is important as well. To ensure that, the setup of the simulation had to be adapted by taking cost reducing measurements such as using alternative types of particles, culling, changing render techniques and stylizing elements to a limited extend.

10. Conclusion

The purpose of the project was to improve the visualization of extreme weather for the Borculo storm project so that more visitors are drawn to the museum and awareness about climate change can be raised. Those improvements meant to add upon the already existing prototype without influencing the performance much in a negative way. Ideally, once the prototype is improved users can immerse in the situation and get educated about extreme weather.

By conducting research and surveys with people from the broad target audience it was possible to create multiple prototypes for the simulation and adjust them based on the outcome of the survey. By ensuring good performance and adjusting to criticism, immersion and realism on screen could be achieved. However, adjusting the experience to VR space was difficult. Due to the Covid situation, only a small number of users were able to test the experience and no scientific representation could be achieved. Besides that, some valuable hints and ideas were given and implemented into the experience.

Eight particle effects in total have been created and adjusted repetitively. A tree has been created which was animated once by a shader and once with keyframe animations. A weather system has been programmed and tailored towards the different types of particles and lighting and the overall setup was modified focusing on performance.

The client also wished to draw more people to a museum by showcasing modern technology and increase awareness about extreme weather. The success of this task will only become clear over time.

11. References

- A.F.P. (2015, April 6). Storm in Bangladesh [Photography]. The Guardian. https://guardian.ng/news/37-killed-in-bangladesh-storms/
- AMANDA. (2020, November 10). Fall Asleep with Heavy Rain, Howling Wind, Great Thunderstorm Sounds - Real Rain Sounds for Sleeping [Video]. YouTube. https://www.youtube.com/watch?v=rCJSBwLHYyY
- Amarillo, TX Weather Forecast Office. (2015). What is a Supercell? National Weather Service. https://www.weather.gov/ama/supercell
- Baez/Reuters. (2017, July 9). Police patrol the area as Hurricane Irma slams across islands in the northern Caribbean on Wednesday, in San Juan, Puerto Rico September 6, 2017. [Photography]. Newsweek. https://www.newsweek.com/irma-remain-category-4-hurricane-next-four-days-660776
- Bestuur | Historische Vereniging Borculo. (n.d.). Historische Vereniging Borculo. Retrieved 26 May 2021, from https://hisvebo.nl/wie-zijn-wij/bestuur-2/
- Brennan, D. (2016, October 6). Why Do I Get Motion Sickness? WebMD. https://www.webmd.com/cold-and-flu/ear-infection/motion-sickness
- Damage and fatalities extreme weather events in Europe on the rise. (2019, August 26). Climatechangepost.Com. https://www.climatechangepost.com/news/2019/8/26/damage-and-fatalities-extreme-weather-events-europ/
- F4 tornado damage example. (2005, November 19). [Photography]. https://upload.wikimedia.org/wikipedia/commons/4/44/F4_tornado_damage_example.jpg
- Fujita-Skala. (2020, November 1). In Wikipedia. https://de.wikipedia.org/wiki/Fujita-Skala#:%7E:text=Hyperschall%2DWindkanal%20erzeugt.-,Enhanced%20Fujita%20Scale,der%20Fujita%2DSkala%20sehr%20verbreitet.
- Gerlits, A. (2017, February 17). Frustum Culling. GameDev.Net. https://www.gamedev.net/tutorials/programming/general-and-gameplay-programming/frustum-culling-r4613/
- Google. (n.d.). VR Performance best practices | Google VR |. Google Developers. Retrieved 27 May 2021, from https://developers.google.com/vr/develop/best-practices/perf-best-practices
- Guidelines for VR Performance Optimization | Oculus Developers. (n.d.). Oculus for Developers. Retrieved 27 May 2021, from https://developer.oculus.com/documentation/native/pc/dg-performance-guidelines/?device=RIFT
- Hendriksen, H. H. (2009, November 6). Three adults sit at a table next to their collapsed house [Photography]. Flickr. https://www.flickr.com/photos/hanshendriksen/5793337282/in/photostream/
- Hendriksen, H. H. (2009a, November 6). Top view of destroyed Borculo [Photography]. Flickr. https://www.flickr.com/photos/hanshendriksen/5793336384/in/photostream/
- Hendriksen, H. H. (2009b, November 6). Man in the wrecks of his house [Photography]. Flickr. https://www.flickr.com/photos/hanshendriksen/5793335922/in/photostream/
- Hogeschool Saxion. (2019, November 28). Saxion opent Extended Reality Lab in Enschede [Video]. YouTube. https://www.youtube.com/watch?v=qJiHJ5HZZQo
- The Importance of Frame Rates. (n.d.). IrisVR. Retrieved 27 May 2021, from https://help.irisvr.com/hc/en-us/articles/215884547-The-Importance-of-Frame-Rates

- Kistemaker, H. K. (n.d.). Willkommen auf der Homepage des Feuerwehrmuseums Borculo. Brandweermuseum Borculo. Retrieved 11 March 2021, from https://brandweermuseumborculo.nl/de/
- McLeod, S. A. (2019, August 03). Likert scale. Simply Psychology. https://www.simplypsychology.org/likert-scale.html
- Meteo Consult Weernieuws Archief. (2005, August 10). Weer.Nl. https://web.archive.org/web/20170303191045/http://archief.weer.nl/?menu=view&cat=tornado&item=cycloonborculo10082005.wn
- Pidcock, R., & McSweeney, R. (2021, April 26). Mapped: How climate change affects extreme weather around the world. Carbon Brief. https://www.carbonbrief.org/mapped-how-climate-change-affects-extreme-weather-around-the-world
- Ralston, M. (2017, August 25). Tropical Storm Harvey [Photography]. New York Magazine. https://nymag.com/intelligencer/2017/08/tropical-storm-harvey-is-strengthening-as-it-nears-texas.html
- Relax Sleep ASMR. (2019, December 1). Detailed Sounds of Rustling Leaves Falling and Wind Blowing through the Forest Trees in Late Autumn [Video]. YouTube. https://www.youtube.com/watch?v=aT66uumZ0Zo
- Saxion. (n.d.). Extended Reality Lab | Saxion. Hogeschool Saxion. Retrieved 11 March 2021, from https://www.saxion.edu/business-and-research/labs/extended-reality-lab
- Shiffman, D. (2012). The Nature of Code: Simulating Natural Systems with Processing (1st ed.). The Nature of Code.
- SpeedTree – 3D Vegetation Modeling and Middleware. (n.d.). Speedtree. Retrieved 17 May 2021, from https://store.speedtree.com/
- The State of the SpeedTree/Unity HDRP Pipeline – SpeedTree. (n.d.). SpeedTree. Retrieved 26 May 2021, from https://store.speedtree.com/the-state-of-the-speedtree-unity-hdrp-pipeline/
- Stormramp van 1925. (2020, November 5). In Wikipedia. https://nl.wikipedia.org/wiki/Stormramp_van_1925
- Strange, R. (2012, September 27). How long will retail customers wait (and what you can do to help)? Irisys. https://www.irisys.net/blog/how-long-will-retail-customers-wait-and-what-you-can-do-to-help
- Sunrise Sunset | Borculo. (n.d.). Meteogram.Org. Retrieved 17 May 2021, from https://meteogram.org/sun/netherlands/borculo/
- Sutton, J. (2020, February 1). What do Volumetric Lighting and Volumetric Fog do - Graphics Settings Explained Written by Jon Sutton on 01 February 2020 at 16:07. Game-Debate. https://www.game-debate.com/news/28294/what-do-volumetric-lighting-and-volumetric-fog-do-graphics-settings-explained
- Understanding High Dynamic Range Rendering - Steve's Digicams. (n.d.). Steves Digicam. Retrieved 28 May 2021, from http://www.steves-digicams.com/knowledge-center/how-tos/photography-techniques/understanding-high-dynamic-range-rendering.html
- UNESCO. (2020, January 7). Why we urgently need to teach and learn about climate change. https://en.unesco.org/news/why-we-urgently-need-teach-and-learn-about-climate-change
- Unity Technologies. (2020). High-Definition Render Pipeline (HDRP) for high graphics quality. Unity. https://unity.com/de/srp/High-Definition-Render-Pipeline
- Unity Technologies. (n.d.-a). High Definition Render Pipeline overview | High Definition RP | 11.0.0. Unity Manual. Retrieved 28 May 2021, from

https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@11.0/manual/index.html
- Unity Technologies. (n.d.). Unity - Manual: Occlusion culling. Unity Documentation. Retrieved 26 May 2021, from https://docs.unity3d.com/Manual/OcclusionCulling.html
- Unity Technologies. (n.d.-b). Unity - Manual: What is a Particle System? Unity Manual. Retrieved 26 May 2021, from https://docs.unity3d.com/2018.3/Documentation/Manual/PartSysWhatIs.html
- Unity Technologies. (n.d.-c). Unity - Scripting API: ParticleSystemRenderer.mesh. Unity Documentation. Retrieved 26 May 2021, from https://docs.unity3d.com/ScriptReference/ParticleSystemRenderer-mesh.html
- Unity Technologies. (n.d.-d). Visual Effect Graph. Unity. Retrieved 27 May 2021, from https://unity.com/de/visual-effect-graph
- Unity Technologies. (n.d.-e). Physically Based Sky | High Definition RP | 7.1.8. Unity Manual. Retrieved 28 May 2021, from https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@7.1/manual/Override-Physically-Based-Sky.html
- Unity Technologies. (n.d.-f). Introduction to ShaderGraph. Unity Learn. Retrieved 28 May 2021, from https://learn.unity.com/tutorial/introduction-to-shader-graph#5f500900edbc2a0022843fb2
- Unity Technologies. (n.d.-g). Unity - Manual: Blend Trees. Unity Documentation. Retrieved 31 May 2021, from https://docs.unity3d.com/Manual/class-BlendTree.html
- Unity Technologies. (n.d.-h). Unity - Scripting API: Mathf.Lerp. Unity Documentations. Retrieved 1 June 2021, from https://docs.huihoo.com/unity/5.4/Documentation/en/ScriptReference/Mathf.Lerp.html
- Unity Technologies. (n.d.-i). Unity.com. Unity. Retrieved 7 June 2021, from https://unity.com/
- Unity Technologies. (2020). Shader-Graph | Build your shaders visually. Unity. https://unity.com/de/shader-graph
- Unity Technologies. (2020b, July 17). Simulating Wind in URP (Shader Graph Tutorial) [Video]. YouTube. https://www.youtube.com/watch?v=ZsoqrHHtg4I

12. Appendix

A. First survey

Link to survey: https://forms.gle/VEkK9Uv7up6SZr3e7

Detailed results

The rain is heavy	The atmosphere is as fo	Enough leafes are flying	The lighting in the scene	The light conditions fit a	Too many leafes are ripp	The fog should be alot m	It is as rainy as it should
2	4	3	5	1	1	3	2
4	3	1	2	1	2	2	3
2	2	2	2	2	2	3	3
3	2	1	4	2	1	1	5
3	4	2	3	4	3	4	4
3	3	2	3	1	2	5	1
2	5	1	2	1	1	2	1
5	1	1	3	2	3	4	3
2	2	1	1	2	1	3	3
3	2	1	2	2	1	3	2
5	4	2	2	3	2	4	4
3	3	1	2	2	2	4	3
3	4	1	3	4	1	3	4
3	2	1	2	2	2	2	4
4	4	3	3	4	4	4	4
4	1	1	3	2	1	1	4
3	4	1	2	1	3	5	4
5	3	3	4	1	1	5	3
3,176470588	2,941176471	1,470588235	2,588235294	2,117647059	1,882352941	3,117647059	3,176470588

The amount of debris tha	The scene should be alol	There is enough rain pre	More fog would fit the stc	The scene should be dar	Enough leaves are flying	Less rain would fit a heal	The scene is as foggy as
2	2	3	1	5	1	1	2
1	2	4	2	5	2	1	3
2	2	1	4	3	2	2	1
1	1	5	3	5	1	1	1
3	1	5	5	5	2	2	2
1	1	1	4	5	2	1	1
2	2	1	2	5	3	1	1
2	1	4	3	5	4	3	3
1	1	3	3	4	1	1	2
1	1	2	4	4	1	1	1
4	1	4	5	5	2	2	2
2	2	3	3	5	1	1	2
3	1	4	2	5	1	2	1
3	1	4	2	5	3	2	3
4	2	4	4	4	4	4	2
1	1	4	2	5	1	1	5
3	1	5	4	5	5	2	4
2	1	4	2	3	3	1	1
2,117647059	1,352941176	3,375	3,117647059	4,705882353	2,117647059	1,647058824	2,117647059

A more dense atmosphe	The lighting is about right	Enough rain is present	The amount of leaves do	The lighting has to be da	A more dense atmosphe	The amount of leaves ne	More rain should be in th
2	5	3	2	4	2	2	4
2	4	4	2	4	3	4	2
3	4	4	3	4	3	3	4
1	5	5	2	5	5	5	1
5	4	4	2	5	5	2	4
5	5	4	4	5	5	2	2
1	5	5	5	5	3	1	1
1	3	5	5	5	4	4	2
2	4	4	2	4	4	1	4
3	4	4	3	4	4	1	3
1	4	5	3	5	5	4	1
1	5	2	4	4	4	4	3
1	5	5	1	4	4	1	4
1	5	5	3	4	4	2	4
2	5	5	4	4	3	4	2
3	4	5	5	4	4	3	1
1	5	5	5	4	4	1	1
1	5	5	5	5	3	1	3
2,058823529	4,470588235	4,352941176	3,235294118	4,352941176	3,882352941	2,588235294	2,625

Less fog would improve t	The rain is as heavy as it	A lighter atmosphere wo	The amount of leaves ne	The tree looks realistic	The movement before th	The movement after the	The tree looks realistic	The movement before th	The movement after the
1	1	1	4	5	3	4	4	2	3
2	2	2	3	2	1	2	2	2	4
1	2	2	3	1	2	5	3	5	2
1	2	1	1	1	3	1	3	1	3
1	1	1	4	4	5	3	5	5	5
1	1	1	1	4	4	2	4	5	1
1	1	1	4	3	5	5	5	1	4
2	2	2	4	4	5	3	4	3	3
1	1	1	3	2	2	2	3	1	3
1	1	1	4	4	4	1	4	4	4
1	3	1	1	3	5	2	4	2	5
2	2	2	4	2	1	3	3	2	2
1	1	1	5	5	5	4	4	1	3
2	3	2	4	4	4	2	5	2	5
4	2	2	2	5	5	4	4	1	5
1	2	2	5	4	5	3	3	5	3
1	1	1	4	4	2	5	4	2	4
4	4	3	1	5	4	1	5	1	3
1,411764706	1,647058824	1,411764706	3,294117647	3,352941176	3,588235294	3	3,764705882	2,588235294	3,470588235

B. VR Testing

Link to survey: https://forms.gle/G6ntzHNYYQmHt5Qg7

Detailed results

What would you change about the rain in order to make the simulation look more accurate?

3 Antworten

- Nothing, the rain was good
- the rain is fine
- The rain looks good.

What would you change about the Fog in order to make the simulation look more accurate?

3 Antworten

- More Fog at the end
- the fog looks okay, maybe stronger
- The can be fog thicker

What would you change about the leafs ripped of the tree in order to make the simulation look more accurate?

3 Antworten

- maybe some of them could land ontop of the balkony
- some more tree details. twigs can fall off to
- The leafs look also good like the rain.

What would you change about the lighting in the scene in order to make the simulation look more accurate?

3 Antworten

- it should be darker in the end
- the scene is too light
- I think the scene could be a little darker

What is missing when it comes to particles?

3 Antworten

- maybe some branchs/twigs in the air?
- i think its fine
- More dirt could be blown to make it more realistic