Wiebke Spieker

Utilizing and Optimizing 3D Photogrammetry Assets for VR Experiences

Case: Pop-Up VR Museum on Meta Quest 2

Metropolia University of Applied Sciences

Bachelor of Culture and Arts

Media

Bachelor's Thesis

28 November 2023

Tiivistelmä

Tekijä(t):	Wiebke Spieker
Otsikko:	Fotogrammetristen 3D mallien käyttö ja optimointi VR kokemuksissa
Sivumäärä:	35 sivua + 2 liitettä
Aika:	28.11.2023
Tutkinto:	Medianomi (AMK)
Tutkinto-ohjelma:	Viestintä
Suuntautumisvaihtoehto:	3D-animaatio ja visualisointi
Ohjaaja(t):	Lehtori Kristian Simolin

Opinnäytetyön tavoite on analysoida fotogrammetristen mallien käyttöä mobiili-VR-kokemuksien 3D-grafiikan kehityksessä. Työn keskipiste on olennaisimmissa grafiikan optimoinneissa, joissa käytetään esimerkkitapauksena Pop-Up VR Museum -projektia. Projektissa esiintyy esimerkkejä optimointitekniikoista, ja työ sisältää kuvauksia lopullisten tulosten luonnista ja haasteista.

Tämän opinnäytetyön tavoitteena on selvittää, miten 3D-fotogrammetriamalleja voi käyttää luomaan todenmukaisia replikoita alkuperäisistä esineistä ja miten niitä voidaan soveltaa VR-kokemukseen aiheuttamatta suorituskykyongelmia.

Avainsanat: 3D, mallinnus, fotogrammetria, virtuaalitodellisuus, optimointi, reaaliaikainen

Abstract

Author(s):	Wiebke Spieker
Title:	Utilizing and optimizing 3D photogrammetry assets for VR experiences
Number of Pages:	35 pages + 2 appendices
Date:	28 November 2023
Degree:	Bachelor of Culture and Arts
Degree Programme:	Media
Specialization option:	3D Animation and Visualization
Instructor(s):	Kristian Simolin, Senior Lecturer

This thesis aims to analyze the utilization of photogrammetry assets to develop 3D graphics for mobile virtual reality experiences. There is a heavy focus on the most important aspects of graphics optimization using the case study Pop-Up VR Museum as an example. The case study showcases examples and explanations on how the finished results were achieved and what challenges were encountered.

My goal for this thesis is for it to serve as a reference on how 3D photogrammetry models can be utilized to recreate close replicas to their original counterparts and be implemented into a VR experience with little to no impact.

Keywords: 3D, mobile, photogrammetry, virtual reality, optimization, real-time

Contents

1 Introduction

This thesis covers the various aspects and challenges of optimizing and utilizing 3D photogrammetry assets to develop virtual reality (VR) experiences with a heavy focus on mobile device development. I share different approaches to utilizing raw object files scanned through photogrammetry and how to make use of them during the modelling and texturing process. I am using Blender, Substance Painter and Unity as my primary programs throughout the process but will not focus on any program specific tools but rather provide a general overview of what can be achieved. Using the Meta Quest 2 device as a baseline serves as the primary constraint and provides challenges similar to the restrictions of mobile phones or devices. It is advised to have a basic understanding of the process of making 3D models prior to reading this thesis.

I am tackling the question of what a 3D artist should take into consideration while planning and developing a virtual reality experience and how to minimize possible discomfort and motion sickness for the user due to unoptimized environments. As virtual reality experiences become more widely available for the public, it is important for 3D artists to make themselves familiar with the topic of how mobile VR development differs from the usual 3D workflow aimed at higher-end machines.

I am providing pictures and examples throughout the thesis which are mostly derived from the case study, Pop-Up VR Museum, developed at the Media Lab of Aalto University in cooperation with Design Museum Helsinki as part of the SPICE (Social cohesion, Participation, and Inclusion to promote Cultural Engagement) research project. The pictures I am using are of models and textures I have created during the development of the experience with a few exceptions of similar projects to be used as a comparison. All finalized models are included in the experience and the examples are either from a work in progress or the final product.

2 Abbreviations and terms

Alpha channel	–	Channel of the texture that defines the transparency of the image
High-poly	–	Polygonal mesh that consists of a high number of polygons
Low-poly	–	Polygonal mesh that consists of a relatively small number of polygons
Material	–	Optical properties of a 3D object’s surface, such as color, shininess, metallic and bumpiness
Polygon	–	A face in 3D space consisting of three or more connected line segments
Shader	–	Function that alters the appearance of a 3D model’s material properties
Texel	–	Texture pixel, smallest unit of information in a texture
Triangulation	–	The process of converting polygons into triangles
UV	–	Process of projecting a 2D image onto a 3D model; U and V stand for the 2D axes of the texture
Vertex	–	A point in 3D space

2.1 VR – Virtual Reality

Virtual reality (VR) is defined as a simulated 3D environment achieved through computer technologies which is being viewed through a head mounted display (HMD). It offers a fully immersive experience while obscuring the natural world and creates a synthetic environment through sensory stimuli, such as sight and sound. (Bardi 2022). Virtual reality relies on the natural occurrence of the brain fusing two identical views of the eyes into one stereoscopic display with a sense of depth. Similarly, the headset displays two almost identical images with a slight shift in perspective which our brains fuse to create full immersion.

While virtual reality is mostly associated with games, the variety of its use cases has dramatically expanded since it has been made available for the public. Many museums and art exhibitions have realized the potential of extending their showcases to digital environments to offer visitors a different perspective on their artworks or a time travel to experience past times entirely. A major reason for the expansion is the rapid advancement in technology. Meta, formerly Oculus, has been publishing affordable and portable devices that can easily be carried between separate exhibitions. Portable, or standalone headsets, are defined as wireless devices that are equipped with integrated pieces of hardware which allows them to function without an external computer.

2.2 Photogrammetry

Photogrammetry is the process of capturing images of an object or space from various perspectives to obtain 3D representations that can be used to extract geometric information via triangulation. This is not to be confused with polygonal triangulation, as in photogrammetry it is the process of capturing overlapping images of the target and identifying similar geometrical points to determine where these points are in a three-dimensional space. (Mubanga 2022). Triangulation relies on sharp and high contrast images of the target or the surrounding area to stitch the images to achieve a good result. Any messy,

unclear, or dark setups can cause inaccuracy and may even result in the process failing.

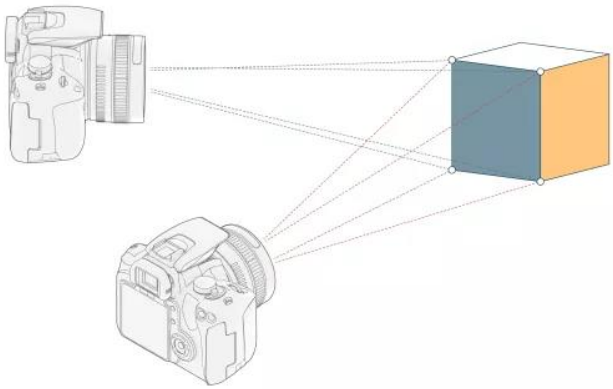

Figure 1. Once the geometrical points of interest have been determined, it can be mathematically calculated where these points intersect. (Mubanga 2022)

Photogrammetry was originally used to obtain reliable measurements from photographs of the surface of the earth. With technological advancements however, we can use computer processes to construct a virtual object from given images resulting in a model with a simple albedo texture. During the scanning process it is advised to provide an even and neutral lighting setup to avoid uneven colors and sharp shadows on the scanned textures. Furthermore, it is crucial to ensure that the neighboring images overlap by at least 30% or more to capture as many viewpoints and as much geometric information as possible (Ylinen 2021).

Figure 2. Visualization of the image stitching process (Mubanga, 2022)

Since the scanning process relies on the images provided of the object, it captures not only the shape but also the color and wear. Many small details can be easily reconstructed digitally which allows us to preserve the object in its current state. From a 3D artist's standpoint, photogrammetry is a great tool to assist in the creation of photorealistic models and 1:1 reconstructions of desired artifacts.

3 Case study: Pop-Up VR Museum

Pop-Up VR Museum is an experimental project aiming at the preservation of cultural heritage from the viewpoints of co-design and design. During the development of the project, I was provided with approximately 50 different photogrammetric scans of a variety of artifacts on display in Design Museum Helsinki. I then processed these scans into VR-ready 3D assets that were imported into the virtual reality project. To provide a realistic and immersive experience, the 3D models were to be kept the same virtual size as their physical counterparts displayed at the museum. As the experience offers something akin to a portable museum, it was decided to use a consumer level mobile VR device with the advantage of it being lightweight, portable, and wireless. The objective was to bring the museum to people, with a heavy focus on studying senior citizens, that otherwise would not be able to visit on-site.

3.1 Challenges and resources

Using a mobile VR device also comes with a handful of challenges. Meta Quest 2 is hardly any more powerful than a mobile phone, which is why it requires similar optimization methods as used in mobile game development. It was important to maintain a stable framerate to reduce motion sickness given that the experience is going to be used by a wide range of people, including some users that have never tried VR before. I was often faced with the challenge that framerate drops whenever the scene contained an abundance of 3D objects, particularly ones that are made of glass. Furthermore, many techniques that are commonly used in mobile game development to improve performance could not quite be applied, as the VR experience relied on many unique models that do not share materials. The target audience being senior citizens proved to be another challenge, as disorientation was a frequent occurrence once they entered the virtual world. Similarly, the lack of experience with technology and physical handicaps restricted the movement inside Pop-Up VR Museum for some users. Therefore, it was important to create an accessible experience with simple UI and navigation.

The scanned photogrammetry models worked as a baseline to recreate the virtual artifacts and help to maintain their original size and shape. However, 3D scanners still struggle to capture highly reflective, dark, or transparent surfaces and produce incomplete meshes and inaccurate textures, resulting in the final shape of the scanned model to require significant touch-ups for it to be considered of acceptable quality. In addition, should lighting conditions change during the scanning process, the exported albedo map features uneven reflections and shadows which require de-lighting. To provide an immersive experience for the user, the goal was to recreate accurate virtual replicas of the artifacts. During the duration of my work, I was provided with all the scanned objects and requested reference photos that were taken at Design Museum Helsinki. I also frequently visited the museum on-site to inspect and analyze the objects on display to fully be able to reconstruct realistic materials for the final objects. The reference photos also provide additional data that can be used

during the process and proved useful to project details back onto the model. To ensure that more detailed variants of each 3D model are displayed when closely observed by the user, three game-ready polycount versions of each artifact were to be implemented using a LOD (Level of Detail) system.

3.2 Practical implementation

To further clarify, Pop-Up VR Museum is a prototype VR experience that allows the users to interact with virtual interpretations of artifacts, immerse themselves into personal stories and experience the objects from perspectives that would otherwise be impossible. This allowed for the creation of immersion scenes, where objects up to ten times their original size can be displayed around the surroundings, or the user may be transported into an oversized pot. These simple, yet effective illusions were especially entertaining for younger and older generations, that could not use similar types of media before and helped to engage the audience with the experience (Vishwanath 2023).

The VR experience currently features two scenes, an introduction that includes a short questionnaire with a 2D animation to showcase how the user interface is used, and the main scene where users can interact with the artifacts. The introduction animation was done through vector art, which was rigged and animated in a 3D program and exported into the engine. This helps to ground the introduction into the scene and avoids large prerendered videos. Pressing and holding buttons on both controllers proved to be difficult for first-time VR users, thus an interface with a single controller that does not require button interaction was found to be more intuitive, especially for seniors. The application was also designed around minimal amount of user interaction to simplify the interface as much as possible. To aid motion sickness and disorientation, a virtual table and chair was integrated into the VR space to replicate the physical table on site that the user would be sat on during user testing (Figure 3). This also proved to be essential for grounding the experience and assisted people with limited mobility. During the development, several workshops at different museums and care homes for senior citizens were organized to engage with the

audience directly and collect feedback on their interaction with the application. Adjustments and improvements could then be easily made according to the collected data.

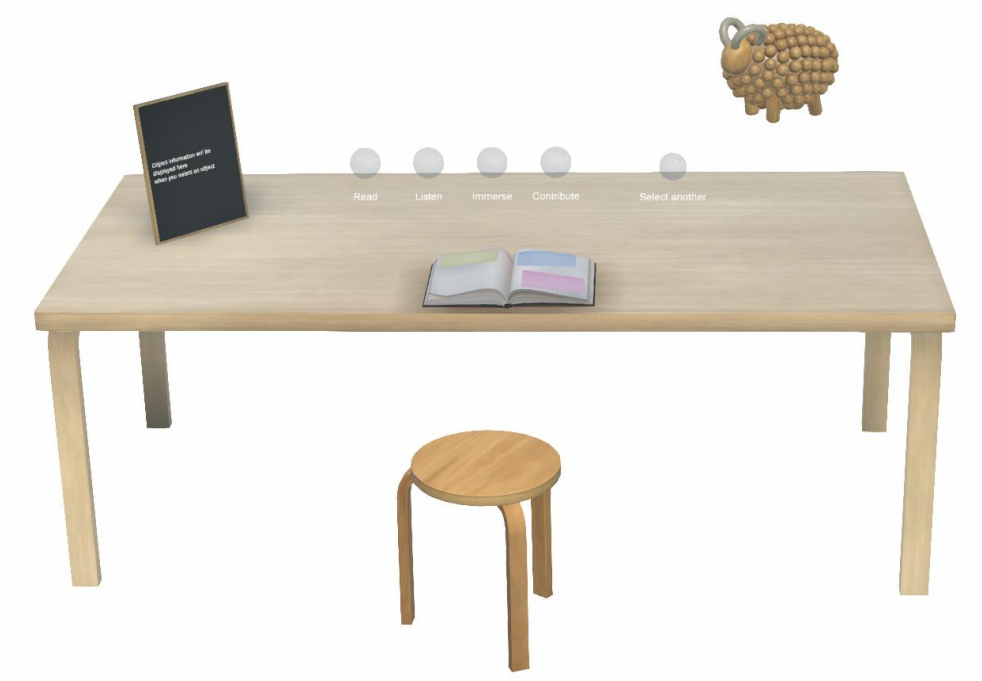

Figure 3. Table set up in the main scene in Pop-Up VR Museum

Pop-Up VR Museum was also featured in several “Design Evenings” at the Design Museum Helsinki throughout the year 2022 to interact with a wider range of people. Part of the artifacts that are included in the VR experience were presented to visitors at the workshops to encourage them to leave either written or audio recorded memories and stories associated with the objects. (Vishwanath 2023). These stories were then implemented into Pop-Up VR Museum through text and sound files which could either be viewed in a virtual book that is placed in front of the user or played through sound. Meta Quest 2 is equipped with an internal speaker system, which made playing the sound files possible.

4 Theoretical background: Mobile VR performance considerations and optimizations

Optimizing VR performance while considering the limitations of the device, user comfort and experience is arguably one of the most critical aspects of VR development. Keeping the experience rendering at around 60 fps (frames per second) while avoiding lag or framerate slowdown can help to tackle serious motion discomfort and nausea (Google VR 2018). Mobile devices are getting increasingly more powerful though, which removes some of these limitations and makes it possible to create more complex and realistic scenes.

4.1 TBDR - Tile-Based Deferred Rendering

Most mobile devices use a rendering technique referred to as Tile-Based Deferred Rendering that optimizes performance and battery life by breaking up the render region into a grid of small tiles. The GPU processes the tiles individually, rendering all contained primitives before writing them back to the memory. For the GPU to know what geometry must be rendered in a tile, a render pass is first computed in a process called binning that calculates which triangles are part of the tile. After this process, opaque geometry is rendered front-to-back which means that any occluded triangles are discarded automatically and not unnecessarily rendered, also referred to as hidden surface removal (Arm Developer 2023. Tile-based GPUs, Apple Developer 2023). Tile-Based Deferred Rendering comes with a set of up- and downsides. Processing the tiles in the cache reduces bandwidth enormously, but dense geometry and long triangles, or slivers, take up a lot of resources due to the binning process, that can slow down the performance. Furthermore, many full screen effects are comparatively more expensive than on PC as they cannot take advantage of TBDR effectively. This is caused by neighboring tiles not being able to share information with each other which would be required in post processing effects such as motion blur and bloom. In terms of anti-aliasing, a technique to reduce jagged edges and flickering caused by aliasing, MSAA (Multisample Anti-Aliasing) is highly efficient on mobile devices due to TBDR.

Aliasing is a common issue in computer graphics caused by the picture rasterization into pixels when the resolution and number of pixels cannot compromise for the details that should be rendered. The large benefit of using 4xMSAA on VR devices outweighs the minimal performance loss and should therefore always be used. MSAA being so light on the performance is due to it being applied during the tile rendering instead of after the full picture has been rendered. (Arm Developer 2023. Multisample Anti-Aliasing, LinkedIn 2023).

From a 3D artist's perspective, Tile-Based Deferred Rendering comes with a set of limitations that need to be taken into consideration. The next few chapters will go through optimization methods that are relevant to a 3D artists' job in more detail, while keeping TBDR in mind. Such optimizations include polygon reduction to avoid dense geometry, avoiding alpha blended materials and relying on MSAA as an anti-aliasing tool.

4.2 Polycount and topology

Polycount is defined as the total number of triangles a 3D object consists of when the model is triangulated in the game engine. Reducing polycount is an essential optimization and should be considered during the modeling process. Excessive polycount on models that are either small or only viewed at from a distance can impact performance drastically, as the GPU must draw unnecessary geometry. That is why every model should not be treated equally and it is important to understand the context of how the 3D model is going to be used in the final software. An efficient model has high visual quality while retaining a low poly count and it is essential to keep balance between these two aspects. However, having high polycount does not automatically result in good visual quality. Through careful considerations and knowledge of topology, an artist has the possibility to create visually pleasing models without drastically impacting performance. (Stepp 2020).

When working with low-poly models, it is important to keep a readable silhouette in mind, especially in VR experiences where the user has the

possibility to inspect models from each side. Polygons should be prioritized on rounded edges to give them a smoother look while unnecessary edge loops around flat surfaces can be reduced. Figure 4 shows an example of polycount reduction while prioritizing the silhouette of the outer rim by collapsing vertices that add unnecessary geometry to the inner edge. Other optimizations include deleting backfaces that are not visible to the user. Models that are only viewed from a greater distance should prioritize using textures and baked details instead of geometry which will preserve visual quality while ultimately reducing rendering costs.

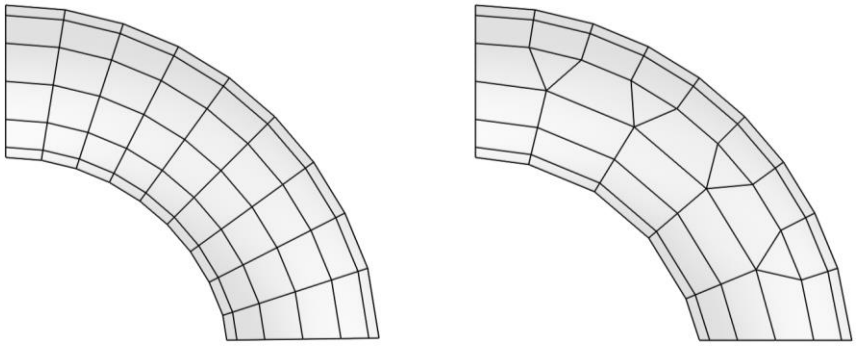

Figure 4. Topology optimization with the silhouette in mind

Many shaders, especially ones that use PBR (physically based rendering) shading models rely on normal maps to add details to flat geometry. Baking normal maps before triangulation can cause inaccurate results when the textures are applied in the engine. This is caused by the interpolated tangent space not matching between the baking software and engine due to the triangulation differences. Tangent space is a coordinate system that specifies the orientation of the UV map surface on each point of the model. Tangent space normal maps rely on this information to be displayed correctly. When a quad is split into triangles, it can be done in two different ways and which diagonal split is chosen depends on the triangulation method a program is using. When baking the normal map, the software automatically accounts for the polygon split and compensates for the tangent space change. Therefore, if the engine is using a different triangulation method, the orientation of the

normals in the map will not match with the tangent space and will cause shading artifacts. That is why it is recommended to triangulate models before baking or importing them to other software to ensure consistency. (Polycount Wiki 2021, Racoon Artworks 2019, Marmoset Toolbag Documentation 2023).

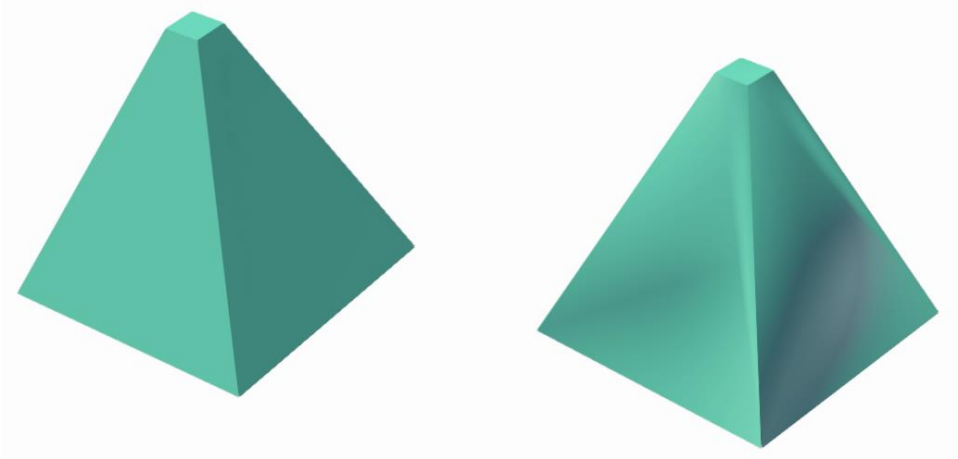

Figure 5. Tangent space mismatch of the normal map on the right

While polycount is an important factor to keep in mind and should be tracked during the process of creating assets, it is usually not the most impactful reason causing performance drops. Alpha-textures and real-time lighting are often the main causes in real-time rendering which is why it is often advised to use geometry instead of transparent materials.

4.3 Texture resolution and compression

Determining the optimal texture resolution for each individual texture is crucial to achieve the best possible result and performance. This can vary a lot, as the art style and the size of the objects play a big role in how large textures are required to display all the details that the artist intends the user to see. This is especially true with VR experiences, as objects are often observed from up close. The challenge is to balance the quality of the picture and size of the texture (Palmi, 2013). Lower resolution textures reduce the needed amount of memory bandwidth but also cause blurry results. As a rule of thumb, all textures should be aligned to the power of two ratio, for example 1024x1024, as most

engines rely on that size format to further process the images. The images do not need to be perfectly square though, as long as they follow the same ratio. One in-engine process is mipmapping, an optimization method where the original texture is stored in a chain of several lower resolution pre-filtered versions of the texture. The mipmaps are automatically swapped in the engine based on the distance of the camera from the viewer, using the lower resolution textures for objects which are further away. This method saves a lot of memory bandwidth, as the lower resolution textures are significantly smaller in memory. Engines can generate and apply mipmaps automatically when the texture is loaded into the program. Mipmapping can be compared to the equivalent of Level of Detail that is typically done for geometric details. (Arm Developer 2023, Texture mipmapping). Mipmapping also improves the general picture quality of the rendered screen, as it reduces texture aliasing, an issue where there is several smaller texels covering one pixel on the screen which will cause noise and flickering.

Another important optimization is texture compression, that is directly supported in most engines. Compression reduces the size of the texture resources and helps lighten the memory bandwidth required to display the graphics on the screen. Especially on mobile devices, minimizing memory bandwidth is important for performance and battery life. These compression schemes are decoded in real-time and behave very differently than more traditional types such as changing the file format of the images. The image is compressed in blocks, often 4x4 pixels, which can be individually decompressed on demand in the area of the image that is used by the shader at a given time (Arm Developer 2023, Adaptive Scalable Texture Compression Developer Guide). These blocks only have a set number of colors, most commonly 2–4, where each pixel inside the block can blend between these given colors. The compression format, however, heavily depends on the platform the software is going to be released on and should be considered accordingly. Mobile devices for instance most commonly use ETC (Ericsson Texture Compression) or ASTC (Adaptive Scalable Texture Compression) while PC builds rely on methods such as BC1–7 (Block Compression).

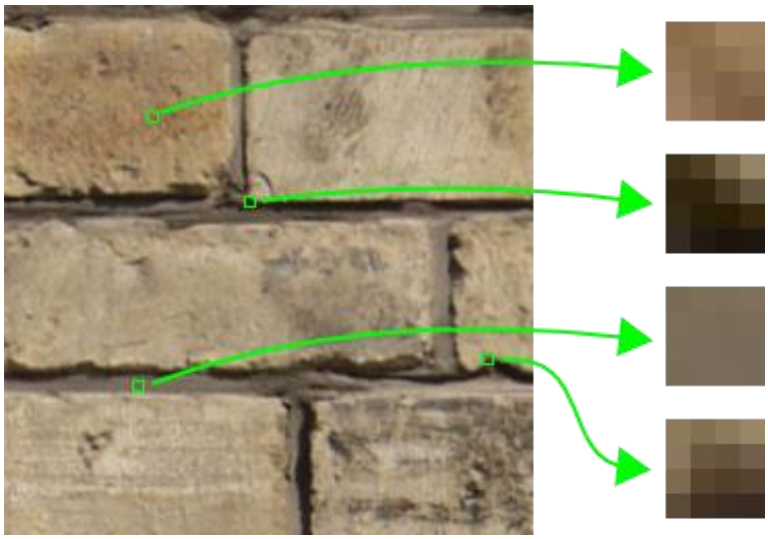

Figure 6. Texture compression done in 4x4 blocks (Reed, 2012)

When comparing the mobile compression formats, ETC and ASTC, it becomes quickly apparent that they behave differently. ETC is especially limited as it only supports two colors per block and loses a lot of the original information. ASTC on the other hand handles color compression far better, as it can use more colors, but comes with the downside of using more bandwidth as it requires twice as much memory for ASTC 4x4.

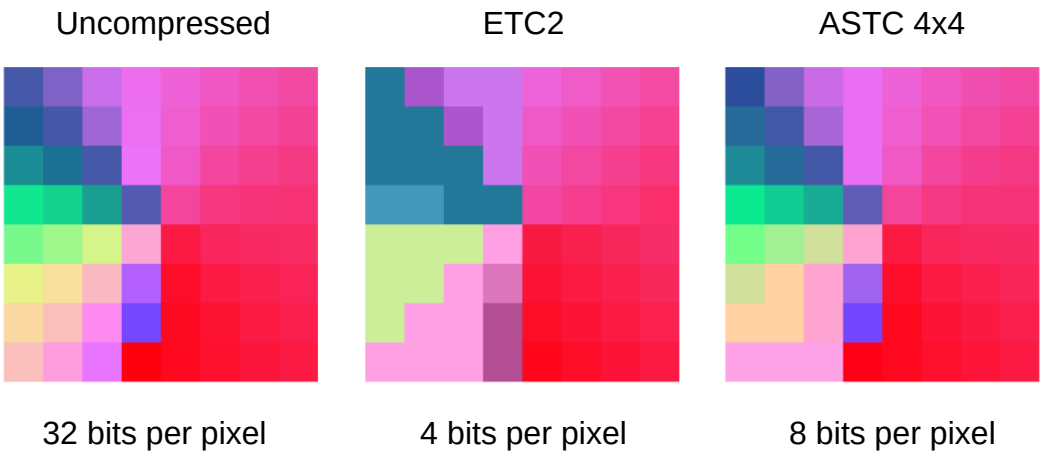

Figure 7. Comparison between ETC2 and ASTC

Another additional compression method is RDO supercompression such as crunch compression, which is used to reduce file sizes while compromising

texture quality. This can be especially useful for software that is available for download or in the browser to keep the file size small. This method will not have any effect on performance, as the crunch is decompressed before loading the textures into the GPU.

4.4 Alpha channels and translucency

Alpha blended or tested materials in Tile-Based Deferred Rendering can lead to immense performance drops, as they force geometry to be rendered back-to-front, instead of the more performant front-to-back. This order ultimately means the GPU computes a lot of unnecessary pixels, even when occluded by the opaque parts of the transparent material and defeats the purpose of the hidden surface removal in Tile-Based Deferred Rendering. This is known as overdraw, meaning the pixels on the screen are forced to be shaded more than once through multiple draw calls. To tackle unnecessary pixel overlap and overdraw, the transparent geometry should be cut off around the alpha clipped object to reduce the amount of fully transparent area as much as possible. If the transparent texture includes large opaque areas, it is also advised to separate those parts to its own submesh and render it with an opaque material to reduce overdraw.

The most common ways to implement transparency are through alpha blending or alpha testing. Alpha blending allows the material to have a range of transparency, while alpha testing can only be either fully opaque or transparent, determined by an alpha mask. Alpha testing removes some optimizations features within the GPU though and should be used in moderation. It is therefore recommended to rely solely on alpha blended materials on mobile devices. (Arm Developer 2023, Profile and compare transparency implementations). When rendering overlapping alpha blended materials in the scene, the engine sorts them by depth. This process determines in which order the objects need to be rendered, but especially in VR experiences where the viewer can move and overlap the objects at will, depth sorting issues can occur. One way of solving this issue would be using some sort of order independent

transparency, or OIT for short, but it is currently too heavy for virtual experiences. For models that include overlapping transparent materials such as in glass cups, this issue can be addressed by separating the inner and outer faces of the geometry and forcing a rendering order through the material settings. These issues do not occur on alpha tested materials though because they benefit from the depth buffer. They are rendered with solid objects, as alpha tested materials are either fully opaque or transparent and do not require the same soft blending as alpha blended materials do.

4.5 LOD – Level of Detail

Level of Detail, or LOD, is a technique where the engine dynamically switches a 3D object between lower and higher resolution versions of itself based on its distance from the camera. It reduces the number of triangles drawn in a scene, as the models in the background will be replaced by low resolution versions, occasionally even background billboards. Objects directly in front of the user on the other hand are rendered in their high-poly counterpart. This ensures that the models present in the scene will not be as heavy to draw and ultimately improves rendering performance. (Unity Documentation 2023).

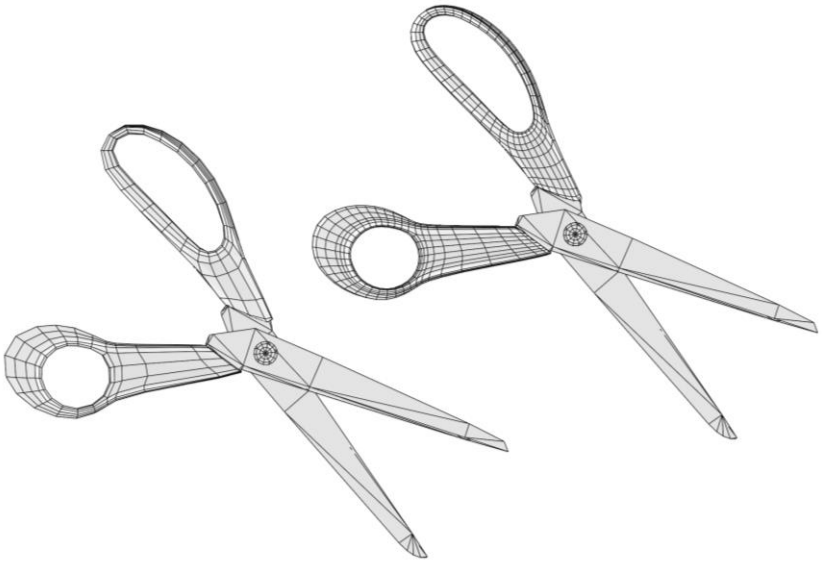

Figure 8. Two LOD levels of the same 3D object, edge loops were only added to places where needed to make the shapes smoother.

Especially in VR games this can be immensely useful as there are often 3D objects that can be picked up and observed up close. In these cases, objects are displayed in isolation and will not have as much of an impact on the performance as if every object was always rendered in high resolution. A downside of transitioning between the different model resolutions is LOD popping, which becomes far more apparent when the model is close to the user. The popping is caused by significant geometric differences between the models of the LOD levels. To avoid this issue, higher resolution models should not differ too much, or a dithering effect should be implemented, that is supported by most engines.

4.6 Draw calls

Draw call is an instruction from the CPU to the GPU, determining objects to render with a given shader. A draw call is required for each unique material and object, which can have a huge impact on the performance if the scene includes a lot of different models. To reduce the amount of draw calls, it is advised to combine smaller submeshes into one and reuse materials when possible. For example, texturing methods such as trim sheets are particularly popular on environmental assets as they can share the same material between a large set of models. Static batching is another method in which the engine combines several meshes to one and renders them at the same time. These objects must be set to be static, which means they will not move during the duration of the draw call. Furthermore, the meshes must also share the same material as one draw call can only render one material at a time.

4.7 Lighting and shadows

Lighting helps to ground the models in the scene and can create unique moods when applied correctly. Dynamic lighting is great for adding another layer to the experience but is unfortunately a very intensive and heavy process, especially on mobile devices. This will cause serious performance issues, which is why baking the lighting into textures called lightmaps is preferable. This can come

with a great advantage as prebaked lighting gives the opportunity to create a realistic lighting model that includes Global Illumination (Android Developers 2023). Baking lightmaps produces a set of textures with lighting information that is applied to the models in the scene. Like other textures, lightmaps also require non-overlapping UV coordinates. These can be generated directly in the engine into a secondary UV set, that will not affect the original UV unwrap. On the downside, the generated textures can quickly become very large and compression methods should be applied to the textures to keep the scene light.

Similarly, dynamic, or real-time shadows are expensive on mobile device and often need to be disabled for most of the objects in the scene. Instead, different approaches to fake shadows are taken, such as adding blob shadows to characters or baking a shadowmask. Shadowmasks allow for combined real-time and baked shadows in the scene at runtime by making use of Light Probes.

5 Pop-Up VR Museum: Creating 3D assets from photogrammetry

Using photogrammetry for Pop-Up VR Museum was chosen to produce accurate digital replicas of the artifacts from the collection at Design Museum Helsinki. The provided models were scanned utilizing two different handheld 3D scanners, Artec Leo and Scanniverse on an iPhone 12 Pro, that output very different results. Artec Leo produced high fidelity models with very accurate details, whereas Scanniverse tended to export 3D objects with missing mesh data and redundant geometry (Figure 9). However, Scanniverse was more adapted to scan large objects or spaces and could be used for objects too large to capture using Artec Leo, such as a bicycle. Despite the drawbacks, photogrammetry provided a great baseline for the modeling process, as the artifacts' shapes and scale were captured in a three-dimensional space that could be directly used as reference.

5.1 Remeshing and texture planning

For VR experiences it is advised to retain the real-life measurements of the object in the virtual counterpart to not only give people a sense of scale but to also add immersion. This is immensely useful for VR application development, as the 3D models present the same measurements as the original artifacts and further scaling is not necessary when imported into the engine. Photogrammetry devices take the scale of each object into account during the scanning process by referencing the surrounding environment. As a result, the imported scans most often have the correct scale, but slight errors might always occur and should be assessed whenever possible.

Figure 9. Comparison of the output quality from Artec Leo and Scanniverse

Scanned models were treated like high-poly meshes that would typically be exported from sculpting software. They were very large in polycount and their topology was highly unoptimized, especially for mobile devices. To tackle this issue, the 3D objects were remeshed, a process where a clean low-poly mesh is laid on top of the high-poly model. This method is most commonly done through manual means, but automatic remeshing tools are becoming increasingly more advanced and can be considered to save time on organic and complex models. The scans, however, were frequently missing mesh data on hard to process surfaces as can be seen in Figure 10. These errors were to be either reconstructed directly in the photogram mesh or approximated during the

remeshing phase. Similarly, capturing the bottom faces of an object rendered to be impossible through Scanniverse, as it does not support 360 degrees scanning. In these cases, detailed all-around reference photos of the artifacts ensured the correct reconstruction of the missing data.

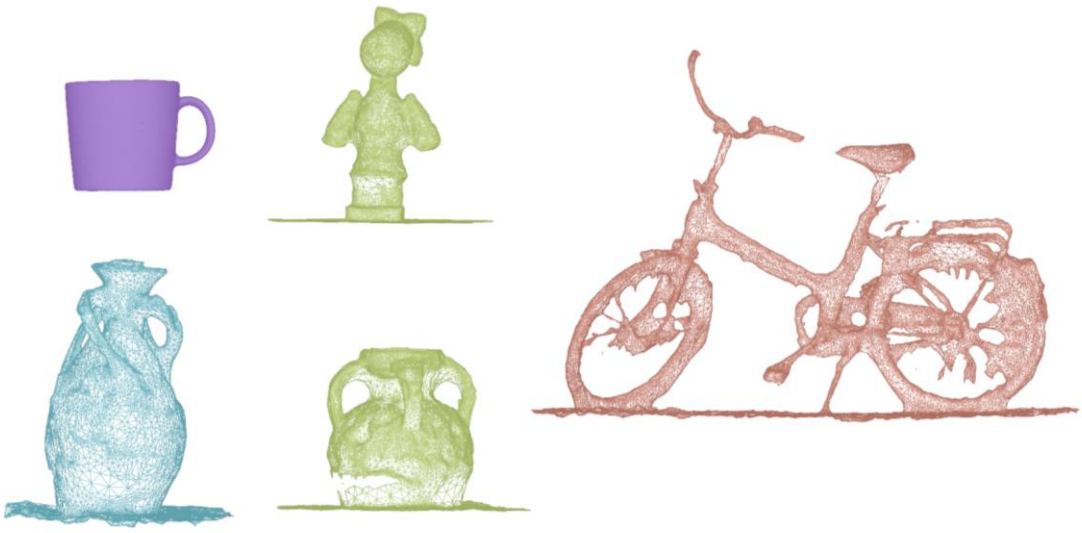

Figure 10. Quality and complexity differences of various photogrammetry meshes

During the remeshing process, the polycount was aimed to be kept at a maximum of 1000-1500 triangles for simple objects, ideally less. This ensured that the models would not impact the performance of the experience, even with an abundance of them present on the screen. However, the artifacts in Pop-Up VR Museum included anything from simple mugs to motorbikes, which proved keeping this goal challenging. The complexity of some artifacts demanded more modeled details raising the polycount. Furthermore, some artifacts, such as the Jopo bike proved to be rather difficult to scan compared to otherwise simple mugs and had to be reconstructed from scratch. Objects made of glass were also impossible to capture and had to be modeled from reference pictures. Fortunately, many manufacturers such as Artek also provide CAD (computer-aided design) files and blueprints from their products on their websites, which could be utilized as reference. This aided in reconstruction of the artifacts enormously.

To guarantee the best possible result, most parts of a 3D model were kept at a consistent texel density during the UV unwrapping process to avoid loss of quality as the model was going to be being viewed from all sides. Parts that contained small details in the texture, such as labels, were given more texture space to compensate for the texture compression in the engine. Some materials such as wood were also taken into closer consideration, as wood grain must follow the object even when bent. This could be achieved by straightening the UV shells after the initial unwrapping process (Figure 11). Planning out the change and flow of materials before texturing was immensely important for achieving the desired outcome. UV seams and shell layout were therefore also placed with the materials in mind. As Pop-Up VR Museum relied on a lot of unique materials that were not procedurally produced, each object received its own set of textures that in hindsight could have been further optimized and should be considered in future iterations.

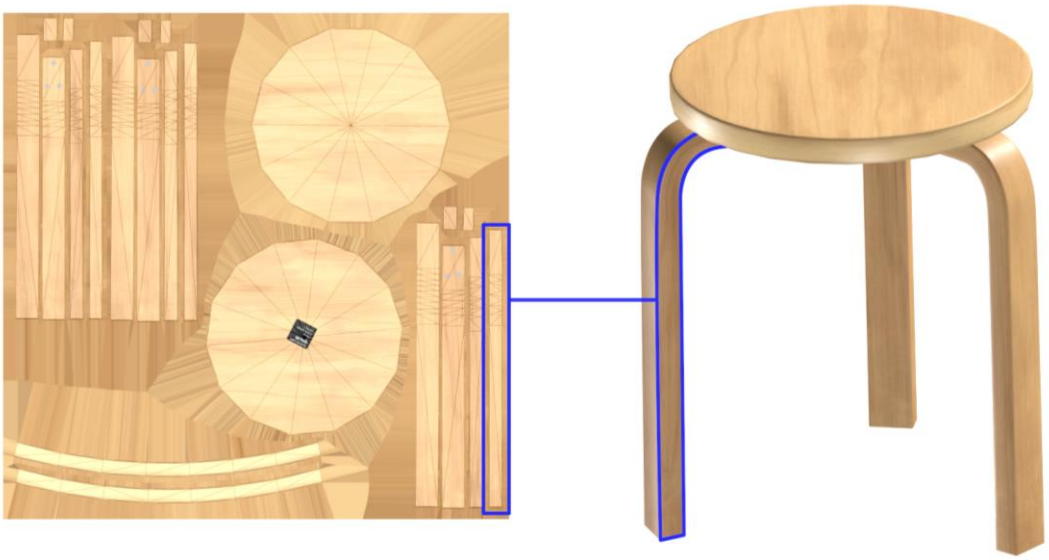

Figure 11. UV coordinates with material flow in mind

As the objects were viewed from different angles and distances, an LOD system with a maximum of three levels for each 3D model was implemented. More detailed variants of each model were displayed in its highest LOD version when closely observed by the user. The LOD levels were hand crafted by increasing the polycount through partial subdivision and adding details after the remeshed

low-poly object has been created. This ensured that the three levels are reusing the same UV coordinates and therefore, each LOD level of one object could utilize the same material. While this was viewed as the most ideal outcome, exceptions had to be made on more complex models that required different baked textures for LOD levels. Highly complex objects also did not receive LOD levels, as their base polycount target was highly exceeded and would have impacted the application's performance too drastically due to the dense polymesh. To further improve performance, smaller submeshes were combined before being exported into the engine to ensure as few draw calls as possible.

5.2 Rebuilding physically based materials

Physically based materials are a close approximation of how a material would reflect light in real life. These materials commonly include albedo, roughness, metallic and normal maps, which are applied to the model in the engine. Photogrammetry computes a simple albedo texture during the scanning process that is particularly useful in the reconstruction of the 3D object's material (Figure 12).

Figure 12. Scanned albedo map compared to the final fixed and de-lit albedo map

During the reconstruction, the scanned texture was projected onto the now UV unwrapped model and brought into the texturing software, where it served as a

guideline for building the rest of the material. Like with missing mesh data, the albedo map can also contain incorrect or absent fragments that will have to be addressed manually. This could be done through photo manipulation or directly in the texturing software depending on the desired workflow. Should the albedo map contain any reflections and shadows, a process called de-lighting was applied.

De-lighting is the process of removing shadows and uneven lighting on the albedo map that occurred during suboptimal lighting conditions during photogrammetry scanning sessions. The undesired lighting information is extracted through masks and filters resulting in an evenly lit texture that contains close to no shadows. De-lighting ensures that meshes are rendered accurately in 3D lighting conditions, as baked shadows in the textures will cause inconsistencies with the otherwise realistic lighting model. These inconsistencies become especially apparent when the environmental lighting changes around the object and cast shadows move, as can be seen in the appended media. Baked lighting in scanned albedo maps looks very convincing as the shadows and reflections are based on realistic lighting. Unfortunately, these are quickly revealed when moving virtual lights or simply observing the object from different angles with the unedited albedo map applied. Other inconsistencies include brightness and exposure differences in the texture that will have to be addressed, as they can cause brightness differences in the color values. Irregular surfaces on objects will also include self-shadowing in the scanned albedo map, that need to be removed through color correction. This can be done by inverting a baked ambient occlusion map, which will serve as a mask that can be blended back into the texture. To remove large lighting imperfections, a high pass filter can be applied to a copy of the image. The high pass filter creates a strong contrasted version of the original texture which can be blended back to even out the lighting difference while retaining micro details. Smaller imperfections such as sharp reflections can be removed through photo manipulation software.

Unless performed in perfect studio setups, scanned albedo maps contain inaccurate color data that needs to be color corrected to match the real-life counterpart. Fortunately, as mentioned above, many manufacturers provide detailed information about their products including precise color data and hex codes. This proved to be of great value, as the virtual textures can be exactly matched according to the provided data during the material reconstruction. Other times stock photos of the items were used to reference the true colors of the artifacts. In ideal conditions, a color calibration chart, for example ColorChecker, should be used during the scanning process when working with complex objects to ensure that the colors are matched correctly.

Once the albedo map was established, other physically based material data such as roughness and metallic maps were reconstructed manually. To accurately rebuild those material data, reference photos were taken during the scanning process from various angles. A virtual lighting environment was then created that approximately equals the reference and aided to match the rebuilt materials to the photos. Through careful masking and internal generators in the texturing software, roughness and metallic data was reconstructed. Wear and small imperfections on the surfaces were especially crucial to make the objects feel authentic. To simulate a feel of depth and wear, normal maps were added to represent variation in the material. This was especially important for the VR experience, as the 3D objects were observed from many angles. Rotating the objects in a lit environment revealed all these small details that added a layer of realism to them. To add these fine details back to the model, a height map was computed from the albedo map. This was achieved through applying a greyscale high-pass filter to the texture which ensured that the height data matched the details on the albedo map (Figure 13). Texturing software such as Substance Painter could then easily convert the greyscale height map into a normal map, that was applied in the engine.

Figure 13. Generated height map through high-pass filter (contrast has been increased for clarity)

The design artifacts frequently featured small details such as labels and other markings, that had to be taken into consideration when building the virtual replications. These details often got lost in the scanning process, as they came out too blurry or inaccurate for the final material. Close-up photographs of these details were requested, as they are used to partially project them back onto the model. Labels were especially tricky, as they could not be replicated by hand and relied on photographs. Other details such as complex patterns and logos utilized a similar approach and were combined with manual manipulation to fit the model. More insignificant details such as screws, indentations and markings were added through creating alpha masks in picture manipulation software and projecting them in the texturing software as shown in Figure 14. No surface is truly flat and must come with a handful of imperfections, may they be large or small, that break the otherwise perfect surface. Adding similar details to the models aided in properly representing the original artifact as a virtual reconstruction and added a sense of realism.

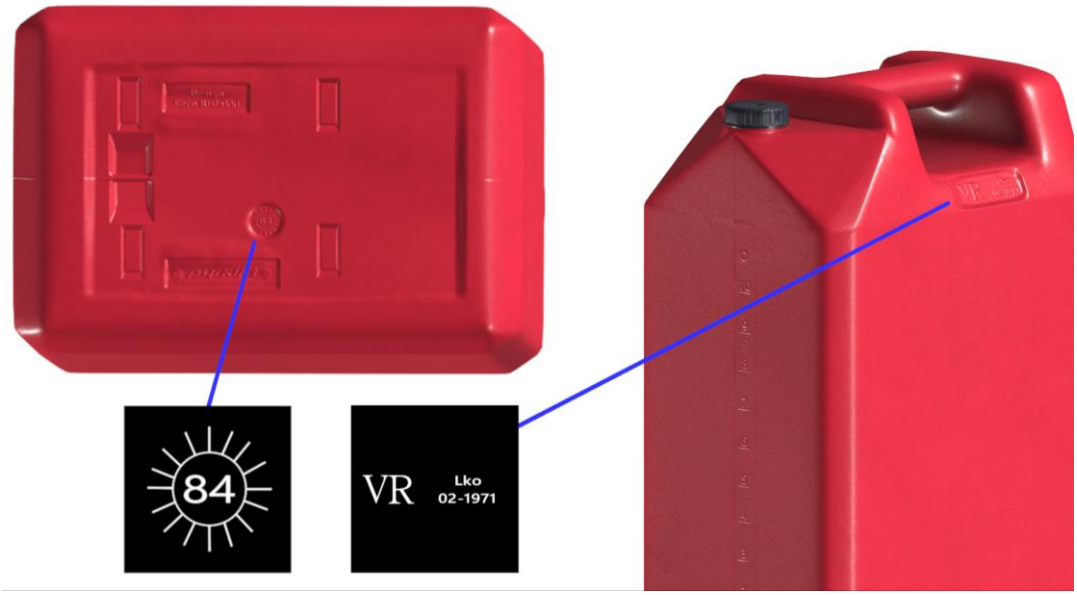

Figure 14. Alpha masks for the details on the object

To ensure the materials would not drastically impact performance, alpha blended textures were avoided for almost every object with only a few exceptions. Most details were added through geometry since opaque materials are cheaper to render, especially when using TBDR. Occasionally some details required the use of alpha blending, where it was used in moderation. Such details included a metal grate at the bottom of a larger model and the edges of a set of rugs. All textures were exported with the power of two in mind to guarantee no complications to occur in the engine when applying texture compression and mipmapping.

5.3 Custom shaders

Pop-Up VR Museum featured objects that took advantage of custom shaders to depict glass and rugs more realistically. Those shaders, on the other hand, were also heavier on the Meta Quest 2 device than common PBR material shaders and caused occasional performance drops. Especially the custom shader used for the rugs had to undergo optimizations, as it was using alpha blended textures in combinations with a cone step mapping shader. Cone step mapping is a technique that utilizes the objects' height map to calculate a height projection onto the flat plane to create the illusion of a displaced object. Steps

are used to project the height on the object and adding more steps will increase the quality of the displacement. A similar and more common technique to cone step mapping is parallax mapping which computes the height through fixed intervals instead of cones.

Figure 15. Mesh separation to reduce the geometry that is affected by the alpha blended shader

Provided studio grade photos of each rug were used as the albedo map for the models and served as the base to generate the remaining textures. De-lighting was not necessary as the near to perfect lighting environment in the photographs caused little to no shadows or imperfections. Any small self-shadow occlusions in the albedo map resulted in added depth and did not impact the appearance negatively, as the shader did not cast any shadows on its own. The required height maps for the shader were generated through applying a high pass filter on the albedo map, similarly to what is done for the PBR materials. The height map required an overall dark texture with pronounced details, which was achieved through adjusting the contrast. Once imported into the engine, the height map was converted into a texture format that the cone step mapper utilized to project the height onto the geometry. In

this process, two sets of height data were stored in the RGB channels of the map. The R or red channel of the texture stored the original height map data while the G or green channel included the cone mapped data that was computed during the import. To further reduce performance impact, the edges of the rugs were cut off the otherwise opaque and solid center, which had a significant impact on the framerate.

Figure 16. Depth sorting issue in the glass on the left

The VR application also featured a wide variety of glass objects, that utilized a refraction shader to give them a more realistic appearance. This caused a lot of depth sorting issues, as most of the objects are made of inside and outside faces. To tackle this, the faces were separated into two submeshes, where each received an own material. The depth sorting order was then adjusted to force the outer mesh to be rendered in front (Figure 16). This method worked for singular objects but would not fix sorting issues with several glass objects present in the scene. Fortunately, the engine was able to sort the objects correctly most of the time and no further depth sorting adjustments had to be made.

5.4 Engine set up and performance

Each 3D object was imported into the project as an Autodesk FBX file and set up as a prefab. Prefabs acted as a template which could be instanced into the scene. Should changes be made to the template, it would apply said changes to every instance of that that prefab. This was immensely useful, as prefabs ensured consistency throughout the project. Furthermore, LOD's were added directly to each prefab and set up through the engines' in-built LOD system. The system allowed for straightforward camera distance adjustments and supported manual LOD level configurations. Each object was tested through a Meta Quest 2 testing headset to make sure the camera distance is set up correctly and transitions between the levels worked seamlessly. Adding LOD levels to the objects had an immediate impact on the performance, due to the reduced polycount in the scene. When testing builds of Pop-Up VR Museum in the early stages, a performance drop could be observed when an abundance of small objects without the LOD system applied were present in the scene. While the current version still drops in performance occasionally, it can render a lot more objects at a time with little to no impact on the framerate. Further topology and material optimizations would most likely reduce those occasional hiccups and could be considered in a possible future iteration.

In terms of texture compression, similar methods as described in chapter 4.4 were integrated into the project. All textures were compressed to half their original size using the ASTC compression method to ensure memory usage and bandwidth was kept low. In the early stages of development, real-time shadows were kept enabled as they did not impact the performance. As more objects were added to the project, performance loss became increasingly noticeable and therefore shadows were disabled. Only a small handful of important assets were kept casting real-time shadows as they received a lot of attention during the user testing. For the general surrounding settings, lightmaps were baked and a reflection probe was added to the scene. The reflection probes' purpose was to cast real-time bounce light to the surrounding models, to ensure consistent lighting from all sides as users interact with them. This also aided to

brighten and ground the models to the scene, as shadows were otherwise creating a rather dark and gloomy contrast to the otherwise idyllic environment.

Option	Impact
4x MSAA	Cheap on mobile (TBDR), important for VR
No real-time shadows	Slow on mobile, requires geometry to be rendered into shadowmaps
Baked Lightmaps	Deletes all drawbacks from real-time shadows and causes no negative impact on performance
Texture Compression	Reduces memory bandwidth due to smaller memory footprint of the textures

Performance profiling could be achieved through Unity's built-in profiler and Meta's provided OVR Metrics Tool profiler on the headset. OVR Metrics Tool included a HUD overlay that reported information inside the application such as frame rate, heat, GPU, and CPU throttle. This was of immense help to debug scenes and objects that caused heavy performance loss, as throttling was tracked in real-time. Further optimizations such as enabling 4x MSAA anti-aliasing tackled jagged and flickering edges that were especially apparent in the VR experience where the user moves the view constantly. Meta Quest 2 unfortunately has a rather low screen resolution, which blurred a lot of the details on the objects. User testing revealed that people observe even the smallest details on objects and bring them closer to their view, thus culling was enabled and adjusted as the objects are otherwise clipping the camera when brought too close to the view.

6 Conclusion

This thesis discussed some of the most crucial aspects of mobile VR optimizations and how I made use of them in my case study, Pop-Up VR Museum. It also went into deeper detail on how to utilize photogrammetry as a means of accelerating the reconstruction process of artifacts. VR experiences especially rely on heavy optimizations, as performance loss can cause nausea and disorientation. Pop-Up VR Museum was designed with senior citizens in mind, which posed its own set of challenges, as most users are not accustomed to using virtual media. Frequent workshops at Design Museum Helsinki or care centers provided firsthand user- and stress-testing, that gave valuable information on how to improve the experience. Pop-Up VR Museum gave me the chance to study and experiment with these optimizations and experience how much of an impact small adjustment can have. Though this thesis focuses on mobile VR devices, such as TBDR, all the optimizations can be translated into game development for mobile phones, as mobile devices rely on similar architectures and rendering methods.

Working on Pop-Up VR Museum was incredibly insightful, not only in terms of optimizing, but also in the 3D reconstruction of artifacts that are present in a museum. Preserving cultural heritage digitally is a powerful approach that allows making museum collections more accessible for people with diverse backgrounds or those who struggle with limited mobility. The digital objects can then be utilized to create interactive media to promote engagement with the audience, either on-site or through other digital means. This can create exciting and enriching experiences for the visitors, that encourages interaction with cultural heritage. To ensure precise reconstruction of the artifacts for Pop-Up VR Museum, photogrammetry and reference photos were used to analyze the objects. Small details and markings were added back to the otherwise flat geometry to give a sense of depth and realism. Of course, there is a limit to how accurately objects can be reconstructed in a restricted time so a small margin of error will always occur. Many hundreds of hours could be spent making one digital replica, if a 1:1 reconstruction is desired. However, the time spent on the

model needs to be always evaluated to the significance and purpose of the object in the application and a sense of artistic touch is often appreciated.

I approached the topic in this thesis with the 3D models and visual representation of the VR experience in mind. Many optimizations could be pushed further through more technical means and extended research. There are many ways to achieve similar results and it is up to the reader how to approach this topic. The workflow I used from photogrammetry scans to texturing and importing the assets into the engine was rather straightforward and allowed for adjustments to be made in each step. Writing this thesis has also given me further insight into the internal restrictions mobile devices propose and how to tackle them. The prototype for Pop-Up VR Museum was completed in April 2023 and is set to serve as a baseline on how museums can consider similar approaches and techniques when digitalizing their collection.

References

Android Developers 2023. Lighting for mobile games with Unity. https://developer.android.com/games/optimize/lighting-for-mobile-games-with-unity (Accessed 19 September 2023).

Apple Developer 2023. Tailor Your Apps for Apple GPUs and Tile-Based Deferred Rendering. https://developer.apple.com/documentation/metal/tailor_your_apps_for_apple_gpus_and_tile-based_deferred_rendering?language=objc (Accessed: 17 October 2023).

Arm Developer 2023. Documentation. https://developer.arm.com/documentation/# (Accessed: 16 October 2023).

Bardi, Joe 2022. What Is Virtual Reality: Definitions, Devices, and Examples. 3D Cloud by Marxent. https://www.marxentlabs.com/what-is-virtual-reality/ (Accessed February 22, 2023).

Dan 2022. Glossary of 3D Terms. Sketchfab Help Center. https://help.sketchfab.com/hc/en-us/articles/360017681872-Glossary-of-3D-Terms (Accessed 13 September 2023).

Google VR 2018. VR performance best practices. https://developers.google.com/vr/develop/best-practices/perf-best-practices (Accessed 13 September 2023).

LinkedIn 2023. How do you fix aliasing in Computer Graphics? https://www.linkedin.com/advice/1/how-do-you-fix-aliasing-computer-graphics-skills-computer-graphics (Accessed: 28 October 2023).

Marmoset Toolbag Documentation 2023. Tangent & Handedness. https://docs.marmoset.co/docs/tangent-handedness/ (Accessed: 18 October 2023).

Mubanga, Kanyanta 2022. What is photogrammetry. Artec 3D. https://www.artec3d.com/learning-center/what-is-photogrammetry (Accessed February 22, 2023).

Palmi, Daniel 2013. Mobiilipeligrafiikan optimointi. Bachelor´s Thesis. Helsinki: Metropolia University of Applied Sciences, 3D Animation and Visualization. https://www.theseus.fi/bitstream/handle/10024/64687/Palmi_Daniel.pdf?sequence=1&isAllowed=y> (Accessed 16.02.2023).

Pangilinan, E., Lukas, S. and Mohan, V. 2019. Creating augmented and virtual realities: Theory and practice for next-generation spatial computing. Sebastopol, CA: O'Reilly Media.

Polycount Wiki 2021. Polygon Count. http://wiki.polycount.com/wiki/Polygon_Count (Accessed 14 October 2023).

Racoon Artworks 2019. Triangulation. https://www.racoon-artworks.de/cgbasics/triangulation.php (Accessed: 18 October 2023).

Rajani, Ricky 2019. Translucent vs Masked Rendering in Real-Time Applications. Meta Quest Developer Center. https://developer.oculus.com/blog/translucent-vs-masked-rendering-in-real-time-applications/ (Accessed: 27 October 2023).

Stepp, Emerson 2020. Polycount? understanding model efficiency. LinkedIn. https://www.linkedin.com/pulse/polycount-understanding-model-efficiency-emerson-stepp (Accessed 14 October 2023).

Unity Documentation 2023. Level of Detail (LOD) for meshes. https://docs.unity3d.com/Manual/LevelOfDetail.html (Accessed 15 September 2023).

Vishwanath, Gautam 2023. Enhancing engagement through Digital Cultural Heritage: A case study about senior citizens using a virtual reality museum, ACM Digital Library. https://dl.acm.org/doi/10.1145/3573381.3596154 (Accessed: 03 November 2023).

Ylinen, Jani and Karlsson, Juhani 2021. How to do photogrammetry for VR: Dynamic high-resolution scene. Varjo.com. https://varjo.com/blog/koyasan/ (Accessed February 22, 2023).

Figure References

Figure 1. Mubanga, Kanyanta 2022. https://www.artec3d.com/learning-center/what-is-photogrammetry

Figure 2. Mubanga, Kanyanta 2022. https://www.artec3d.com/learning-center/what-is-photogrammetry

Figure 5. Reed, Nathan 2012. https://www.reedbeta.com/blog/understanding-bcn-texture-compression-formats/

Appendices

Appendix 1 — Example renders of reconstruted 3D models


MOBIRA

Appendix 2 — Video

Showcase of the differences between the scanned and fixed models and includes examples of in-engine footage.

https://www.youtube.com/watch?v=q35XOE5Liak