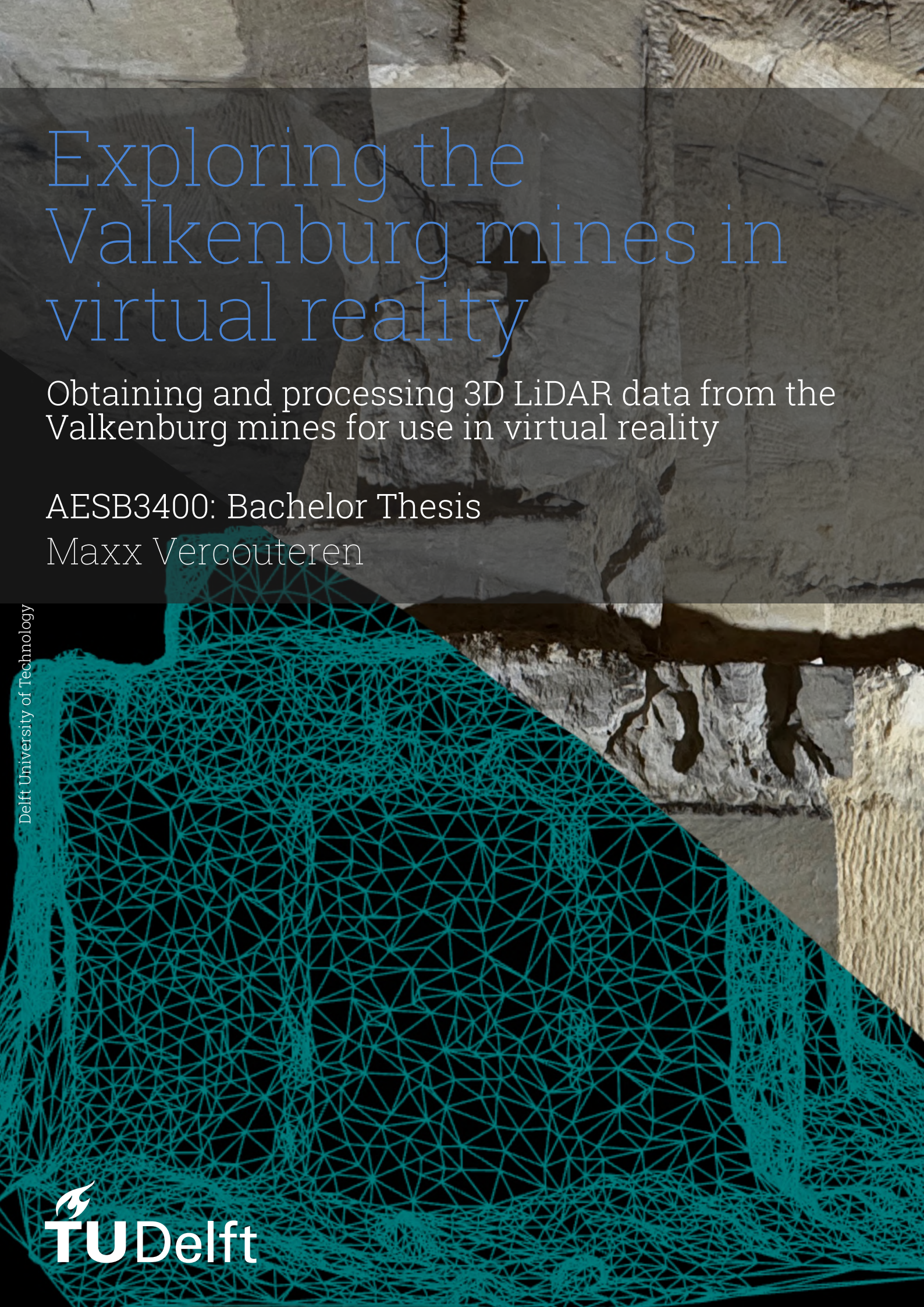


Exploring the Valkenburg mines in virtual reality

Obtaining and processing 3D LiDAR data from the Valkenburg mines for use in virtual reality

AESB3400: Bachelor Thesis

Maxx Vercouteren

Delft University of Technology

TU Delft

Exploring the Valkenburg mines in virtual reality

Obtaining and processing 3D LiDAR data from the Valkenburg mines for use in virtual reality

by

Maxx Vercouteren

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended on Tuesday July 8, 2025 from 14:00 to 14:30.

1st Supervisor:	R.C. Lindenbergh
2nd Supervisor:	D.J.M. Ngan-Tillard
External assessor:	S. Muraro
Student Number:	5811864
Project Duration:	May, 2025 - July, 2025
Faculty:	Faculty of Civil Engineering and Geosciences

Version	Date	Changes
1.0	17/06/2025	Early draft
1.1	23/06/2025	Pre-final draft
1.2	27/06/2025	Peer review Bernard Vercouteren - van den Berge
1.3	03/07/2025	Feedback supervisors
1.4	04/07/2025	Final draft

Cover: Maxx Vercouteren, April 2025

TU Delft

Preface

This thesis was written to fulfill the graduation requirements of the BSc Applied Earth Sciences, at the Delft University of Technology. I was engaged in researching and writing this thesis from May to July 2025.

I am happy to end my time as a bachelor student after three years with a project that perfectly aligns with my interests. It was a perfect combination of my passion for working and visualizing data from remote sensing and my interest in game development. It was a unique project with many different aspects, ranging from collecting data to visualizing it in a VR game, resulting in some new findings every day.

I would first like to thank my two supervisors, Roderik Lindenbergh and Dominique Ngan-Tillard, for all the support, feedback and ideas they gave me during the entire project, as well as all the material they provided and planning the fieldwork to Valkenburg. I would also like to thank Sharif Bayoumy and Yosua Andoko from the XRZone for giving me ideas and tips on how to create the game demo in Unreal Engine, as they have a lot more experience than me. I also want to thank Jacquo Silvertant and Cesco Ramakers for allowing us to scan in the mines in Valkenburg and taking their time to guide us through the caves. Without them, we would never have been able to scan in the mines in the first place. Additionally, I would like to thank all my friends and family for supporting me in my journey from start to finish!

I look back at an amazing time as a bachelor student for the last three years, however the journey doesn't end here. After concluding the bachelor I plan to follow the EC&T master at the TUDelft with the earth observation track, which is in the direction of this thesis. I am very happy with how everything went during this whole project, and I hope that it will help me during the master.

Maxx Vercouteren
Delft, July 2025

Abstract

Bachelor students of Earth, Climate & Technology (EC&T) at the TUDelft have to do post-mining risk management assignments in the old mines in Valkenburg as part of second-year courses. They will do this by inspecting and assessing the stability of underground structures. To achieve this, a virtual reality application is created, which will be part of a larger project called VROCK. For this, 3D models have been obtained in the Valkenburggroeve and the Plankertgroeve. Selected points of interest are scanned, which are often large pillars. The points of interest are scanned using two LiDAR scanners on iPhones. An app called Scaniverse is used for scanning, which gave 3D mesh models as output. These scans result in models with a high mesh density, and thus they need to be optimized to be computationally light enough for virtual reality. This is solved using quadric error metrics using edge contraction. This method ranks every edge of the 3D mesh by the error caused by optimization, and then contracts these edges based on this ranking. The optimized models are then compared with the original versions to assess their quality as well as comparing how other scan ranges and processing modes change the optimized results. It is found that a scan range of 5 meters with the detail processing mode result in the most suitable models for virtual reality, as they show the highest texture quality with the lowest polygon count. The new optimized models are prepared for the final virtual reality application, which is made in the Unreal Engine 5.3 game engine. Multiple methods such as Level of Detail, as well as the types of levels used and lighting placement and lighting type are used to increase performance as much as possible. The XRZone at the TUDelft Library will create the full application with my processed models; however, to assess performance, a small virtual reality demo is made for players to get a feeling of the final result. A theoretical limit of 1,125,000 rendered polygons is assumed based on the virtual reality hardware; however, the results show that at most 465,478 polygons were rendered. Based on this, the performance is very well, as older hardware should also be able to work with the amount of polygons being rendered. Additionally, a survey is held to assess the quality of the game demo. Players are asked about different light intensities, motion sickness as well as if they prefer three smaller levels instead of one large level. The results of this are overall very positive, with almost all players preferring three levels instead of one.

Contents

Preface i

Abstract ii

Abbreviations iv

List of Figures vi

List of Tables vii

1 Introduction 1
- 1.1 Background . . . 1
- 1.2 Problem statement . . . 1
- 1.3 Research questions . . . 1
- 1.4 Objective and scope of the study . . . 2
- 1.5 Thesis outline . . . 2

2 Acquiring LiDAR data in the Valkenburg mines 3
- 2.1 Methodology of LiDAR scanning and meshing . . . 3
- 2.2 Resulting scans from Valkenburg . . . 5
- 2.3 Discussing the Valkenburg scans . . . 7

3 Optimizing the 3D meshes 9
- 3.1 The methodology behind mesh optimization . . . 9
- 3.2 The optimized models . . . 14
- 3.3 Discussing the optimized models . . . 15

4 Evaluating the LiDAR data 18
- 4.1 Mesh evaluation methodology . . . 18
- 4.2 Results of the evaluations . . . 20
- 4.3 Discussing the evaluation results . . . 21

5 Creating the Virtual Reality application 25
- 5.1 Methodology behind the VR game . . . 25
- 5.2 Survey results . . . 29
- 5.3 The resulting VR application . . . 30
- 5.4 Discussing the results . . . 32

6 Conclusion and Recommendations 33
- 6.1 Conclusion . . . 33
- 6.2 Recommendations . . . 34

References 35

A QEM optimization comparison statistics 37

B ZO2-S0 comparisons 41

C Schematic maps of the Gemeentegroeve 44

D Game demo experiment results 47

E Equipment used during fieldwork 48

F Game demo features and items 49

Abbreviations

Abbreviation	Definition
EC&T	Earth, Climate & Technology
LiDAR	Light Detection And Ranging
LOD	Level Of Detail
POI	Point Of Interest
QEM	Quadric Error metrics
ROI	Region Of Interest
UE5.3	Unreal Engine 5.3
VR	Virtual Reality
3D	Three Dimensional
3DGS	3D Gaussian Splatting

List of Figures

2.1 An example of a mesh showing the vertices (points), edges (lines) and faces (areas). Own work. . . . 3
2.2 A photo taken during the scanning of one of the many pillars. All the lamps are placed at the corners of the pillar, and the iPhone is used for scanning. Photo from Roderik Lindenbergh, 2025. . . . 5
2.3 Unaltered 3D models of ZO2-S0. Own work. . . . 7
2.4 Unaltered 3D models of two pillars. Own work visualized in Scanivesre. . . . 7
2.5 Some of the empty polygons in ZW3-pillar. The missing polygons are enclosed by the orange line. Own work. . . . 8

3.1 The UV map of a small scan of the roof in the Valkenburggroeve. Each fragment shows part of the roof texture. Obtained from NIANTIC, INC. (n.d.) . . . 9
3.2 Visual comparison between the mesh and splat models, highlighting a cobble at ZW3-pillar with the red circle. All figures are own work. . . . 10
3.3 Edge contraction visualized: the valid pair is merged into a single vertex, removing the edge between the pair and connecting all other edges to the new vertex. Retrieved from Garland and Heckbert (1997). . . . 11
3.4 Comparison between two 3D meshes of a bunny: the original (left) and optimized (right), which has 1% of the original polygons. Model from Principia Mathematica, Inc. (2004). . 13
3.5 Various visual artifacts that affect the visual and structural fidelity. All figures are own work. 13
3.6 Simplified workflow in Blender showing the entire process for optimizing a 3D mesh. Own work. . . . 14
3.7 A selection of ZW3-pillar before (left) and after (right) optimization with a ratio of 0.76. Own work. . . . 15
3.8 Artifacts in the texture at ZO2-S5-L next to the extensometer, marked by the red circles. Own work. . . . 15
3.9 The custom UV map for the floor sample scan used for Figure 3.10. Every fragment is seen as a rainbow-like colored piece. Modified from NIANTIC, INC. (n.d.) . . . 16
3.10 The impact of QEM optimization on the UV map textures on the floor sample scan, comparing the original model (left) with the optimized model (right). The red circle highlights one of the major changes between the two models. Own work. . . . 16
3.11 One of the many (floating) structural artifacts from the ZW3-pillar site being shown as orange. Own work. . . . 17

4.1 The geometry nodes within Blender for calculating the distance between the QEM optimized mesh and the original mesh. Own work. . . . 18
4.2 The shading code within Blender to visualize the error caused by QEM. Own work. . . . 19
4.3 Box plots for all the scanned locations showing the closest distance between each point on the QEM optimized mesh and the original mesh. Own work. . . . 20
4.4 Histograms for the *Detail* and *Area* mode, showing the error caused by QEM for ZO2-S0. Own work. . . . 21
4.5 Comparison of scan ranges showing part of the optimized ZO2-S1 model. The optimization is exaggerated to better show texture artifacts, and the actual models use higher optimization ratios. . . . 22
4.6 Comparison of UV maps between rendering modes for ZO2-S0. Obtained from NIANTIC, INC. (n.d.) . . . 23
4.7 Comparison between processing modes, showing part of the optimized ZO2-S3 model which shows a long crack in the wall. . . . 23
4.8 Poor textures on permanent equipment for ZO2-S5. Own work. . . . 24

5.1 Comparison between different LOD's. It is shown as wireframe, which shows polygons. 26
5.2 Light complexity as seen in UE5.3, where the scale goes from least (left, black) to most (right, white) complex lighting. Retrieved from Epic Games, Inc. (2023c). 27
5.3 The classification used to determine the structural damage of a pillar. Retrieved from Bekendam (2008) 28
5.4 Workflow from Blender to UE5.3. Own work. 29
5.6 The light complexity in level 1 from a top-down view, which is the most complex level as there is a lot of light source overlap. The scale goes from least (left, black) to most (right, white) complex lighting. Own work visualized in UE5.3. 30
5.5 Previews of all the different levels. All figures are own work visualized in UE5.3. 31

A.1 Histograms of all the scanned models using the *area* processing mode, showing the closest distance between each point on the QEM optimized mesh and the original mesh. Own work. 38
A.2 Histograms of the scanned models using the *Detail* processing mode, where each plot shows the error caused by optimization. Own work. 39
A.3 Box plot of the scanned models using the *Detail* processing mode, showing statistics of the error caused by optimization. Own work. 40

B.1 Comparison of UV maps between scan ranges for ZO2-S0 (area processing mode). Retrieved from NIANTIC, INC. (n.d.). 41
B.2 The 3D model for ZO2-S0 with the long range scanner. Own work. 42
B.3 3D model for ZO2-S0 with the medium range scanner. Own work. 42
B.4 Difference model between the original and optimized ZO2-S0 model Own work. 43

C.1 Map of the Gemeentegroeve showing all the different regions and their names. The red circle shows the main area of interest. Retrieved from Bekendam (2008) 44
C.2 Map of the Gemeentegroeve showing the classification of every pillar, based on the classification of Figure 5.3. The red circle shows the main area of interest. Retrieved from Bekendam (2008). 45
C.3 A zoomed in part of the map from Figure C.2 within the ZO2 region, showing the locations of the important pillars. These are the only pillars where the location matters. Altered from Bekendam (2008). 46

F.1 The hub level, with the instructions on controls, and an interactive piston with cubes that can be grabbed. Own work 50
F.2 The menu with minimap which shows where the player is located in the level. From here, the player can move to any level, reset their orientation or end the demo by pressing *Quit game*. Own work 50
F.3 The street sign-like text appearing on the POI's that allow for easy navigation without having to toggle the minimap. Own work. 51
F.4 The headlamp which can be toggled on and off, which can be handy in dark situations. Own work. 51
F.5 The permanently installed extensometer at ZO2-S5. Own work. 52
F.6 The permanently installed crack-meter at ZO2-S2. Own work. 52
F.7 The Schmidt hammer being used at ZO2-extensometer by Maxx. Own work. 53

List of Tables

2.1 General statistics of the 3D mesh obtained in the Valkenburggroeve. 6
2.2 General statistics of the point cloud data obtained in the Valkenburggroeve. 6

3.1 The change in polygon count for all locations, each with a different ratio based on visual aesthetics. 14

4.1 The change in polygon count for all locations for the model using the *Detail* process option. Ratio used for QEM is the value used for the decimate option in Blender, while the total ratio is the ratio between the optimized and original polygon count. 21
4.2 Polygon counts and the optimization ratio for the Detail and Area processing modes for ZO2-S2. 22

A.1 Table showing the mean and standard deviation for the closest distance between each point on the QEM optimized mesh and the original mesh for all locations for the *Area* mode. 37
A.2 Table showing the mean and standard deviation for the error caused by optimization for each location for *Detail* processing mode. 40

B.1 Polygon counts and the optimization ratio for the long and medium scan ranges for ZO2-S2 (area processing mode). 41

D.1 The results of the experiment from the VR game demo. 47

Chapter 1
Introduction

1.1. Background

The VROCK project is a virtual reality application that already exists which is a tool for bachelor students from Earth, Climate & Technology to prepare for fieldwork and getting to know relevant equipment. This allows for a more efficient fieldwork, as students are already familiar with how to get to work. There already exists an application, where students study a road cutting in Bellmunt in Spain, which is a familiar location as the geo-engineering section has often conducted fieldwork there. VROCK is now being expanded to have more different objectives in new locations, and one of these locations will be in the Valkenburg mines.

1.2. Problem statement

Prior to 1600, miners extracted large blocks of limestone used for building stones underneath Valkenburg in quarry's (Bekendam, 2008). They did this using a room and pillar method, which horizontally mine limestone while keeping pillars that support the quarry roof (Esterhuizen, Dolinar, Ellenberger, & Prosser, 2011). Over time the stability of the pillars has changed, and to this day is still changing. Bekendam (2008) shows which areas are mostly safe and which areas have instable pillars, however as the problem is ongoing there could be changes in the stability. This is crucial for Valkenburg, as it is part of Valkenburg industrial heritage and it is a popular tourist attraction that thousands of tourists visit yearly, as well as urban safety issues. One of the areas with poor pillar stability is the ZO2 area where there is equipment placed on the pillars to monitor the pillar movement. This is exactly underneath where the Wilhelmina tower used to be.

These pillars need to be re-assessed using visual inspection to see if their instability changed, and thus determine if the place is still safe enough. This will be done mainly by second year bachelor students. Virtual reality (VR) is used as it is an easy way for students to virtually go into the mines. LiDAR (Light Detection And Ranging) is used to scan the points of interest in the Valkenburggroeve, and then models of those scans are created. Although there is a dedicated team (XRZone team) that creates an actual functioning VR application, the models still need to be scanned, optimized, assessed and then visualized in VR to see if they are suitable for this application.

1.3. Research questions

Based on the problem statement from Section 1.2, the main research question has been formulated as follows:

How can LiDAR data and meshes from the Valkenburg mines be obtained, optimized and visualized in Virtual Reality while preserving structural and visual fidelity for VROCK?

The main research question contains four elements: data collection, model processing, model evaluation and model visualization. The goal is to explain the entire process from start to finish on how to collect the data and create a virtual reality application from that data, with a main focus on preserving structural and visual fidelity.

Multiple sub-questions have been formulated to help answer the main research question:

- *How can LiDAR data be acquired in the Valkenburg mines, and what challenges can be encountered during the scanning process?*

- *How can the LiDAR model be optimized to meet VR performance requirements while maintaining visual and structural fidelity?*
- *How can the optimized 3D meshes be improved further to be more suitable for a VR model?*
- *How can the final 3D models of the Valkenburggroeve be implemented in virtual reality?*

1.4. Objective and scope of the study

The main objective of this thesis is to collect, process, assess and visualize LiDAR data from the Valkenburg mines into a VR application which can be used for VROCK. 3D scans of some mines near Valkenburg have been acquired in the past for larger areas, however the quality of the structure and texture is not always satisfactory for detailed structure damage inspection. This will be improved with high quality scans of smaller points and region of interest (ROI), which can be merged to create a singular 3D environment for users to walk through and measure things.
As the main objective suggests, the scope of this thesis is broad and has been defined beforehand. The research consists of four parts in total, based on the sub-questions from Section 1.3:

Part 1. Collection of LiDAR data in the Valkenburg mines using an iPhone.

The first part mainly focuses on the data collection part of the thesis. This part focuses mostly on the entire workflow when scanning the points of interest.

Part 2. Optimizing the collected 3D models to be light enough to run on a standalone VR device.

After part one, part two goes into the optimization of the collected data, with a main focus on which method is the most suitable for this research, while taking graphical performance and visual fidelity into account and finding a good balance. The goal is to have good performance while the VR-headset is in standalone mode, meaning it's working independently from a computer.

Part 3. Evaluating and comparing 3D models.

The third part focuses on how different models compare with each other. The optimized model will be compared with the original model, as well as different scan ranges and mesh processing modes which were used to generate a 3D mesh. The evaluation will be both quantitative and qualitative.

Part 4. Creating a suitable game environment for Virtual Reality based on a survey.

Finally, part four is mostly about creating a convincing and pleasant VR experience for the users using Unreal Engine 5.3. Unreal Engine 5.3, often referred to as UE5.3, is a widely used game engine which is what the VR application will be made in (Epic Games, Inc., 2023a). With all the optimized models available and evaluated, all that needs to be done is stitch them together while keeping the in-game objectives in mind. A survey is held to improve the experience of players in VR, and potentially improve the 3D mesh models.

1.5. Thesis outline

This thesis is split up in multiple parts as defined in Section 1.4, which each part having their own chapter. chapter 2 is about the LiDAR data collection, chapter 3 focuses on the optimization of the 3D models while Chapter 4 evaluates these 3D models with each other. Afterwards, chapter 5 is about creating the suitable VR environment in Unreal Engine 5.3. Finally, chapter 6 gives a brief conclusion for each chapter, and concludes the research as a whole as well as recommendation on how to proceed further with this part of the VROCK experience.

Chapter 2
Acquiring LiDAR data in the Valkenburg mines

This chapter starts off with the methodology of LiDAR scanning and meshing (Section 2.1), followed by the scan results are shown (Section 2.2, which are discussed afterwards (Section 2.3).

2.1. Methodology of LiDAR scanning and meshing

2.1.1. LiDAR data introduction

LiDAR, also known as Light Detection And Ranging, is the scanning method that has been used to collect 3D data from the Valkenburg mines. It can measure the distance to a surface using light waves, which travel at the speed of light. A light wave is emitted from the scanner at t_0, which is then partially reflected off the surface and received by the scanner at t_1 (Vosselman & Maas, 2010). With this delay, a distance can be calculated using the following formula:

$$Z = \frac{c(t_1 - t_0)}{2} \tag{2.1}$$

Where Z is the distance from the scanner to a point on a surface and c is the speed of light. Since the laser beams do not travel through surfaces, multiple scans have to be taken when scanning objects to create a full model, which will have to be aligned with each other afterwards.
Of course, many points are scanned at once on a surface, which will give a set of discrete points on a 3D surface. This is called a point cloud, where each point has certain Cartesian coordinates and could have other attributes such as intensity or color (Red, Green & Blue) (Vosselman & Maas, 2010).

2.1.2. Meshes and polygons

In most 3D environments, meshes are used to show surfaces and objects. A mesh is a collection of connected polygons (triangles) which make up a surface as can be seen in Figure 2.1. Each polygon consists of three elements: Vertices, the corners of the triangle, which are three points; edges, which are literally the edges of a triangle, and thus three lines; faces, which is the area that fills up the triangle. A mesh can be generated based on a point cloud, and using certain methods, a surface can be approximated.

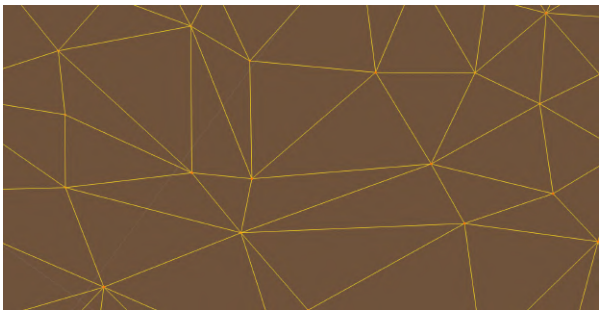


Figure 2.1: An example of a mesh showing the vertices (points), edges (lines) and faces (areas). Own work.

2.1.3. LiDAR scanning fieldwork

Prior to scanning

Before scans were made in the Valkenburggroeve, an approach on how to scan the pillars had to be made, as there were many things that had to be taken into account and time in the mines was limited.

- The points of interest are in an abandoned part of the mine, where there is no illumination. To solve this issue, four industry-size floor lamps (Appendix E) have to be brought along to completely light up any pillar from all sides. In case the pillars are too large and cannot be fully illuminated, they must be scanned multiple times. They are stitched to each other afterwards. Additionally, headlamps were brought along which were useful while traveling (Appendix E).
- When trying to stitch pillars to each other, it can be very difficult to identify neighboring pillars. Thus simple markers were brought along to help identify pillars, which are also useful to check if the scale of the model is correct.

Additionally, it had to be determined which LiDAR scanners would be brought to the mines. Many different options were available, each with their advantages. For example, the Leica P40 3D Laser Scanner was available (Leica Geosystems, n.d.). However, the scanner is very bulky and each scan will require a lot of time. Two iPhones equipped with a LiDAR scanner have been chosen to scan in the Valkenburggroeve, as they are easy to use and lightweight, reducing scanning time. The accuracy of the distance of the scanned points has an accuracy of ±10 cm. This is suitable for the VR application (Luetzenburg, Kroon, & Bjørk, 2021). The camera quality is much higher compared to the Leica P40 3D Laser Scanner, resulting in high quality textures.
For the iPhones, Scaniverse has been used to scan the pillars (NIANTIC, INC., n.d.). It is a free, easy to use app that allows users to choose the range of their scanner (ranging from 0.3 to 5 meters) and immediately process their point cloud into a textured mesh with different processing options. However, it is not possible to access the raw point cloud data, even though it is saved on the device. For this thesis that is not ideal as it is unknown how the model is meshed and how much it is simplified. Therefore, besides using Scaniverse an alternative has been sought. Polycam does allow the user to save and export raw point cloud data, but these features are not free and thus not used (Polycam, 2021). A GitHub application has been chosen to scan point cloud data in addition to Scaniverse (Ryanphilly, 2021). Although a bit experimental, as it has to be downloaded on an iPhone through Xcode, it does provide the user with the full point cloud data with a simple interface and easy to export.

During scanning

Jacquo Silvertand (professional on mining heritage) and his colleague Cesco guided us to the ROI, which was located deep in the mine. The following procedure was taken for each pillar: The lamps were placed near the corner of a pillar, and small markers were placed on the pillar. One phone used a scanning range of 5 meters, while the other a range of 2.5 meters. When scanning, it is preferred if the scanner is perpendicular to the wall that is scanned. This is to ensure high quality textures obtained from the camera of the phone. Figure 2.2 shows a picture from inside of the mine while scanning the pillars. While scanning, Scaniverse shows which parts of your model have not been properly scanned yet, allowing us to check wether the whole POI (Point Of Interest) has been scanned properly. The scan is then saved as a draft, ready to be processed. This draft shows a rough preview of the model. For the first two pillars, the experimental application was also used for future usage.

Figure 2.2: A photo taken during the scanning of one of the many pillars. All the lamps are placed at the corners of the pillar, and the iPhone is used for scanning. Photo from Roderik Lindenbergh, 2025.

After scanning

After the fieldwork, each scan was processed within Scaniverse. Although raw scan data cannot be accessed by the user, it can be stored for later re-processing. All models have been processed using the Area mode, which Scaniverse claims to be the best option for large walls (which is further discussed in Section 4.1). Each pillar has been given a specific name based on it's location and characteristics. Finally, all models are exported as .obj files.
The raw point cloud data from the GitHub application (Ryanphilly, 2021) for the first two pillars have been given the same name as their Scaniverse counterpart, but with indication that they are point clouds, and are exported as .ply files.

2.2. Resulting scans from Valkenburg

2.2.1. 3D mesh models

The processed 3D models can be previewed directly from Scaniverse. In total, 8 pillars have been scanned, along with 9 other points of interest. Below are a few examples. Figure 2.3b shows the very first pillar that has been scanned.

Figure 2.4a shows one of the points of interest, which has a smaller, manmade column. Finally, Figure 2.4b shows a different type of pillar in the Plankertgroeve, also in Valkenburg.

Table 2.1 shows the polygon count of all the different scanned objects. Some scans contain many more polygons compared to others, which is a result from two scans being stitched together due to poor illumination of the floor lamps. This results in a lot of overlap in the model, and thus a high amount of polygons. Appendix C shows the locations and naming scheme of the pillars where location is crucial for stability comparisons. For other pillars, location is not of importance, as they are spread out a lot in the enormous mine system. Different features were targeted in the mine, which will be displayed and grouped in VR for dedicated training. The -L at the end of each name indicates that this scan has been made using the long scan range.

Location	Polygon count
ZO1-1-L	120,812
ZO2-S0-L	269,116
ZO2-S1-L	128,228
ZO2-S2-L	119,758
ZO2-S3-L	147,993
ZO2-S5-L	140,796
ZO2-S6-L	92,609
ZO2-S7-L	317,125
ZW3-pillar-L	162,556
ZO2-purple-L	242,720
ZO2-extensometer-flint-L	151,588
ZO2-extensometer-L	132,143
Random ruble-L	152,295
Roof sample-L	114,564
Floor sample-L	109,070
New mine pillar-L	153,672
Central-L	119,948
Total polygon count	2,674,993

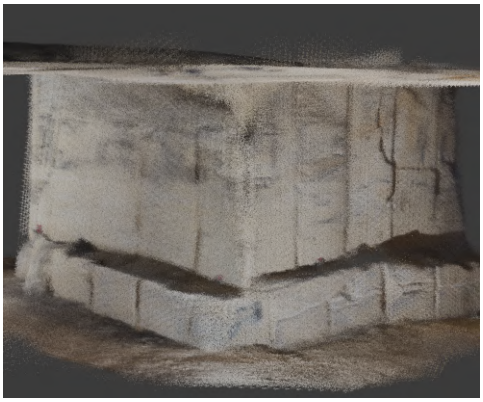
Table 2.1: General statistics of the 3D mesh obtained in the Valkenburggroeve.

2.2.2. 3D point clouds

The point cloud files from the GitHub app (Ryanphilly, 2021) have been imported into Blender for quick visualization. Blender is an industry standard when it comes to 3D modeling, and is used often for visualization and calculation in this thesis. Blender has to give each point a certain radius; a radius of 0.003 meters has been chosen for this render, as it gives the most representative view of the pillar. Figure 2.3a and Figure 2.3b show pillar ZO2-S0. Figure 2.3a looks fuzzy, but this is due to the large amount of points being rendered. Table 2.2 shows the amount of points contained in point cloud files. In Table 2.1, every name ends with -L because the long range scan was used. However, since there was no option to choose a scan range for the GitHub app, there is no -L at the end of the file names.

Location	Points
ZO2-S0	7,401,410
ZO2-S1	11,431,696
Total points	18,833,106

Table 2.2: General statistics of the point cloud data obtained in the Valkenburggroeve.

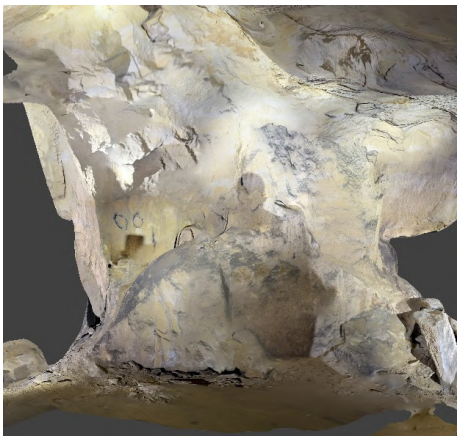


(a) Unaltered part of the 3D point cloud created from the GitHub app (Ryanphilly, 2021), as seen in Blender.

(b) Unaltered part of the 3D model, as seen in Scaniverse.

Figure 2.3: Unaltered 3D models of ZO2-S0. Own work.

(a) Unaltered part of the 3D model of a pillar in region ZW3.



(b) Unaltered part of the 3D model of a pillar in the Plankertgroeve,

Figure 2.4: Unaltered 3D models of two pillars. Own work visualized in Scanivesre.

2.3. Discussing the Valkenburg scans

2.3.1. 3D mesh models

On first sight, there don't seem to be any major issues in Figure 2.3b, which is very good considering this has not been altered yet. The quality is very good and seems the model is realistic, despite not altering any of the data yet. Figure 2.4a does have a minor issue. This was a more complex area to scan, so unfortunately some of the rubble behind the pillar has not been scanned, resulting in some areas with no polygons, as can be seen in Figure 2.5. Although not very visible from a static image, the rubble right of the pillar is modeled mostly as a flat texture, which in practice is not a big problem as the shadows cast by the lamp next to it make it seem more 3D instead of a flat texture. Unfortunately it went worse, as seen in Figure 2.4b. Due to bats in this cave, none of the floor lamps could be used, and instead a flashlight was used that followed the iPhone around. Although the structure of the pillar is mostly accurate, the texture is not, because it is very difficult to point the flashlight exactly where the iPhone is scanning at all times. This results in an uneven light distribution, and there is also a shadow which can be observed. This all makes this specific pillar seem unrealistic, which makes it difficult to implement in VR properly.

Figure 2.5: Some of the empty polygons in ZW3-pillar. The missing polygons are enclosed by the orange line. Own work.

2.3.2. 3D point clouds

Similarly to the 3D mesh of ZO2-S0, there do not seem to be any major problems with the 3D point cloud from Figure 2.3a. Some outliers can be observed, which is likely due to a small measurement error. Furthermore the overall structure and texture seem very similar to it's 3D mesh counterpart.

Chapter 3
Optimizing the 3D meshes

This chapter starts off with the theory behind the simplification method and why this specific method was used, as well as the workflow for this method (Section 3.1). Afterwards, some resulting simplified 3D meshes are presented (Section 3.2). Finally, the results are discussed (Section 3.3).

3.1. The methodology behind mesh optimization

3.1.1. Method comparison

Many different mesh simplification methods are available which can be used for the optimization of the scanned 3D meshes from the Valkenburggroeve. One of the available methods is Quadric Error Metrics, often referred to as QEM, with edge contraction (Garland & Heckbert, 1997). Simply put, it calculates how much error is created for each vertex when simplified by combining two vertices (which are connected by an edge), then each vertex is ranked from highest to lowest rank based on the error, with vertexes that create a small error when optimized being ranked the highest (Garland & Heckbert, 1997). This method is widely used, and has gotten many modification over the years. For instance, pair contraction can also be used, which is similar to edge contraction but two vertex points do not necessarily have to be connected by an edge (Garland & Heckbert, 1997). As the reference shows, this method is on the older side. However, there are many modified versions of QEM which are discussed. The following section will show why this old method is used compared to newer ones: One of the newer modified QEM methods that may be of interest is QEM optimization with appearance attributes (Hoppe, 1999). This works similarly to original QEM, however it tries to take appearance attributes such as color into account when optimizing. This sounds like a very useful tool to use for the Valkenburg mines models, however this modified method will not have any effect on the results. This has to do with the way the textures work in the models. The textures shown in the figures from 2.2 are textures from a separate UV map file. A UV map is the texture of a 3D model unwrapped and projected onto a 2D plane (in the U (horizontal) and V (vertical) direction), as shown in Figure 3.1 (Villanueva, 2022). This often works the other way around: the UV map wraps itself around the 3D model to correctly place the textures on the model. This is often used in 3D modeling with highly detailed textures, as it preserves the texture quality when a mesh is simplified. Since the color attributes are not used because of this, the modified QEM does not provide different results, and thus will not be used.

Figure 3.1: The UV map of a small scan of the roof in the Valkenburggroeve. Each fragment shows part of the roof texture. Obtained from NIANTIC, INC. (n.d.)

There are also other methods such as Vertex clustering (Low & Tan, 1997), which divides the 3D model into smaller cells, and all vertices in a cell are simplified to a single vertex within the cell. Although

computationally not demanding, the mesh quality is not as good as QEM, as the error is higher(Kim & Kouh, 2022). This method is mostly useful for extremely large 3D models, which is not the case for this thesis (Low & Tan, 1997).

In recent years, a new visualization method has been made that has risen in popularity: 3D Gaussian Splat (3DGS). Instead of having surfaces with textures, 3DGS work similarly to point clouds, but instead of being a cluster of points it is a cluster of gaussian ellipsoids with a color and opacity attribute (Taka, 2024). This method is very object centric, and if there is no specific object to render the quality of the splat will significantly decrease. However, since the opacity is also calculated, things like glass can be shown in a 3D model, which is not possible with an unaltered mesh from LiDAR data. Since there is no hard surface as with a mesh, it is possible that poor data areas may be see-through or have a very low resolution. This is shows by rendering a splat model in Scaniverse (Taka, 2024)(NIANTIC, INC., n.d.). This is shown in Figure 3.2b When directly compared with the corresponding mesh, it is clear that the splat model is not suitable for VR, as the cobble can barely be recognized in the splat model.

(a) Selected cobble from ZW3-pillar, shown as a mesh.

(b) Selected cobble from ZW3-pillar, shown as a splat model.

Figure 3.2: Visual comparison between the mesh and splat models, highlighting a cobble at ZW3-pillar with the red circle. All figures are own work.

Finally, a method is chosen. As stated above, vertex clustering and modified QEM do not improve the 3D mesh, so normal QEM is used together with edge contraction. Pair contraction is not used, as QEM with edge contraction already requires a lot of computations, and adding more valid pairs increases computational costs substantially. The software used to compute the optimized mesh is Blender, using the decimate modifier (Blender, 2025). Blender is an open-source program. This means that anyone can look at the source-code of the program. The source-code of Blender is checked to see wether it actually uses QEM with edge contraction, which it does (Blender, 2023). If the reader wants to confirm this, they should take a look at function *bm_face_triangulate* (row 475 to 684). The advantage of Blender compared to other software such as Python libraries is that the model can be immediately visualized in Blender, and there is a very high chance that optimizing outside of Blender will not translate properly to the UV map and thus show incorrect textures.

3.1.2. Quadric Error Metrics Theory

Now that the optimization method has been chosen, this method will be thoroughly explained. Edge contraction works as follows: In a mesh, every pair of vertices connected by an edge is called a valid pair. Let's call these two vertices v_a and v_b. During optimization, the two vertices are moved to a single new point, called v_c. As v_a and v_b now perfectly overlap each other, one of the two is removed. This results in the edge connecting the original pair is also removed, and that all the other edges that are connected to v_a and v_b are now connected to v_c. Figure 3.3 shows an illustration of how edge contraction works in a simple mesh (Garland & Heckbert, 1997).

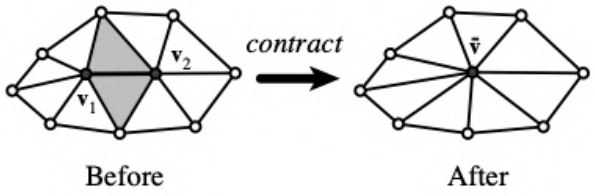


Figure 3.3: Edge contraction visualized: the valid pair is merged into a single vertex, removing the edge between the pair and connecting all other edges to the new vertex. Retrieved from Garland and Heckbert (1997).

Now that edge contraction has been explained, it is time to move on to QEM. Quadric Error Metrics is a method that calculates the error caused by optimization for every vertex, called to cost, and then ranks every edge contraction based on that cost. To define this cost for a single pair, it is defined that $v_c = [v_x, v_y, v_z, 1]^T$ with the error at any point v as shown in equation 3.1 (Garland & Heckbert, 1997).

$$\Delta(v) = v^T Q v \tag{3.1}$$

Where Q is a 4 by 4 symmetric matrix for any vertex. To compute this matrix Q, the error quadric, the error $\Delta(v)$ can be seen as the sum of the squared distance from any point to a plane. The plane is the face of the mesh that is enclosed by the edges, which can be written in the vector form $p = [a, b, c, d]^T$, which represents the formula for a linear plane $ax + by + cz + d = 0$. The distance from a point to a plane is defined as shown in equation 3.2 (Garland & Heckbert, 1997):

$$D = \frac{|ax + by + cz + d|}{\sqrt{a^2 + b^2 + c^2}} \tag{3.2}$$

The term $a^2 + b^2 + c^2$ has been normalized to be equal to 1 such that equation 3.2 can be reduced to equation 3.3 (Garland & Heckbert, 1997).

$$D = |ax + by + cz + d| \tag{3.3}$$

This allows the distance D to be written as $p_i^T \cdot v$. Since the quadric error formula used the squared distance, it can be written as shown in equation 3.4:

$$\Delta(v) = \sum_{i=1}^{n} (D_i)^2 = \sum_{i=1}^{n} (p_i^T \cdot v)^2 \tag{3.4}$$

Here, p_i is the i-th plane that is connected to the vertex v, with n amount of planes connected to this vertex. With the expression from equation 3.4, the error can already be calculated. However, the formula will be restored using linear algebra to it's original quadratic form from equation 3.1(Garland & Heckbert, 1997):

$$\Delta(v) = \sum_{i=1}^{n} (v^T p)(p^T v) = \sum_{i=1}^{n} (v^T p p^T v) = v^T (\sum_{i=1}^{n} K_{p_i}) v \tag{3.5}$$

Where K_{p_i} is the fundamental error quadric matrix for plane p_i, which can be written as (Garland & Heckbert, 1997):

$$K_p = pp^T = \begin{bmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{bmatrix} \tag{3.6}$$

As equation 3.6 shows, K_p is a symmetric matrix, similar to the Q that was defined earlier. From this, as well as equation 3.5 and equation 3.3, the following can be said about Q (Garland & Heckbert, 1997):

$$Q = \sum_{i=1}^{n} K_{p_i} \tag{3.7}$$

With this, the quadric error at any point v_c can be calculated. However, now it has to be determined at which location v_c the cost is minimized. This can be calculated. Knowing that 3.1 is quadratic, finding a minimum cost is a linear problem. v_c is calculated by solving $\frac{\delta\Delta}{\delta x} = \frac{\delta\Delta}{\delta y} = \frac{\delta\Delta}{\delta z} = 0$, which can be written out as follows (Garland & Heckbert, 1997):

$$\Delta(v_c') = a^2 \cdot x^2 + b^2 \cdot y^2 + c^2 \cdot z^2 + d^2 + 2ab \cdot xy + 2ac \cdot xz + 2bc \cdot yz + 2ad \cdot x + 2bd \cdot y + 2cd \cdot z \quad (3.8a)$$

$$\frac{\delta\Delta}{\delta x} = 2a^2 \cdot x + 2ab \cdot y + 2ac \cdot z + 2ad = 0 \quad (3.8b)$$

$$\frac{\delta\Delta}{\delta y} = 2b^2 \cdot y + 2ab \cdot x + 2bc \cdot z + 2bd = 0 \quad (3.8c)$$

$$\frac{\delta\Delta}{\delta z} = 2c^2 \cdot z + 2ac \cdot x + 2bc \cdot y + 2cd = 0 \quad (3.8d)$$

This has been rewritten into a simpler form, as shown in equation 3.9a. Each entry for the top three rows in equation 3.9b, q_{ij} is the entry of the i-th row and j-th column value of matrix Q (Garland & Heckbert, 1997).

$$Pv_c = A \quad (3.9a)$$

$$P = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.9b)$$

$$A = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (3.9c)$$

To determine the most optimal location of v_c, equation 3.9 has been rewritten slightly:

$$v_c = P^{-1}A \quad (3.10)$$

From equation 3.10, the position of the new vertex point v_c can be calculated which minimized the error created by optimization. However, there is a possibility that P can not be inverted. If this happens to be the case, there is another method to determine the location of v_c. This is done by determining the error created by optimization at point v_a, v_b and $\frac{v_a+v_b}{2}$, and from that choose the point with the lowest error. This results in a much higher chance of getting a larger error, but it does work for all cases (Garland & Heckbert, 1997).

With everything ready, now all that needs to be done is to rank each vertex. This is done based on the cost $\Delta(v_c)$ of a vertex, where lower cost vertices get ranked highest (Garland & Heckbert, 1997). When optimizing, the user needs to select how much to reduce the 3D model. Figure 3.4 shows two 3D meshes, where the right model has been reduced from 69,630 to 696 polygons using QEM with edge contraction. Here, only 1% of the original number of polygons remains. This figure also shows how well QEM can preserve structure when a model is optimized.

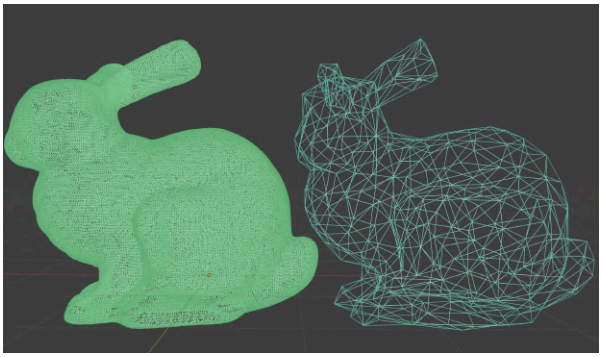


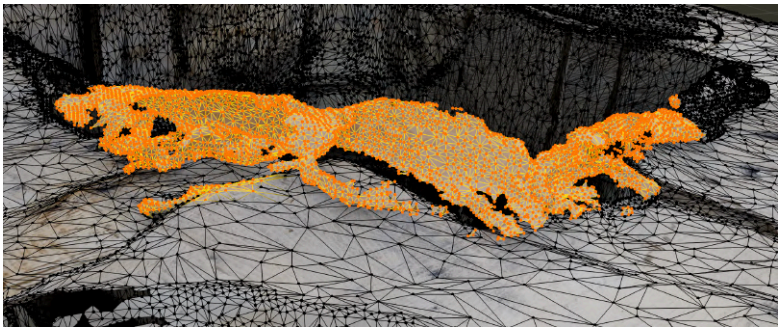
Figure 3.4: Comparison between two 3D meshes of a bunny: the original (left) and optimized (right), which has 1% of the original polygons. Model from Principia Mathematica, Inc. (2004).

3.1.3. Workflow in Blender

With the 3D models in Blender and the theory ready, the 3D models are optimized. To start, a single model is imported and visualized in Blender. Before using any optimization methods, the model is manually cleaned up. There are some polygons that are incorrect, as they are floating or attached either above the roof or below the ground due to a measurement error, as can be seen in the highlighted part of Figure 3.5a.

Using the decimate modifier, and then within this modifier the option ”collapse” is used. This is the QEM option for optimization (Blender, 2023). The ratio of polygons to reduce must be entered, which indicates how much polygons have to be removed. A target of 75,000 polygons per model is chosen for now, as fully rendering all the pillars at once (except for the floor and roof sample) would then result in 1,125,000 polygons being rendered, which is just lower then the theoretical limit the VR hardware can handle (Meta, 2024), which will be discussed in Section 5.1. In practice this goal may not always be reached, as visual fidelity is also an important factor when optimizing. The new optimized model must keep showing subtle details like cracks in the pillars as well as the equipment installed on the pillars. Optimizing too much results in the (partial) removal of these features. Additionally, some optimizations do not work well with the UV map, as shown in Figure 3.5b. There, it can be seen that some of the features that should be continuous have become clearly discontinuous. At the optimization stage before optimization, the pillars that were scanned multiple times due to poor illumination of the floor lamps are merged together to form one proper model.

After a model is optimized enough while having correct textures and clear features, general statistics are noted down. The entire workflow is then repeated for all models, each with their own ratio of polygons to reduce. A schematic and generalized workflow can be seen in Figure 3.6



(a) Orange section shows a structural artifact, where incorrect polygons are floating above the roof of the mine, which have to be manually removed.

(b) A texture artifact on one of the pillars, marked by the red circle. The cause is due to the UV map not working well with with optimization.

Figure 3.5: Various visual artifacts that affect the visual and structural fidelity. All figures are own work.

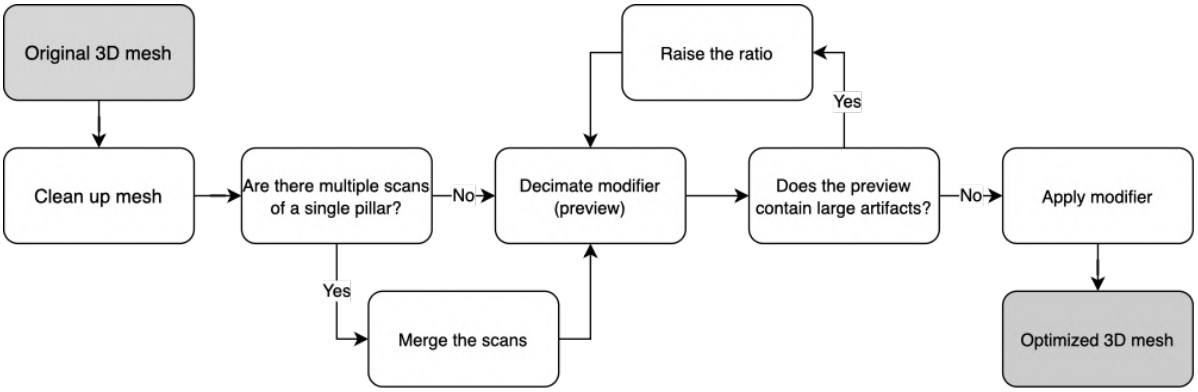


Figure 3.6: Simplified workflow in Blender showing the entire process for optimizing a 3D mesh. Own work.

3.2. The optimized models

3.2.1. Mesh optimization results

A total of 17 individual points of interest (POI's) have been scanned and optimized, as can be seen in Table 3.1. The table shows that the original polygon count varies a lot, as well as the ratio. This is due to texture artifacts that appear when the ratio becomes too low, as discussed in Section 3.1.3. Figure 3.7 shows the polygons before and after optimization. Of course the lower the ratio the better for performance in VR, but it is important to note that although some ratios are low, there are still mostly no major texture artifacts.

Table 3.1: The change in polygon count for all locations, each with a different ratio based on visual aesthetics.

Location	Original polygon count	Polygon count after manual cleaning	Polygon count after QEM optimization	Ratio used for QEM	Total ratio (%)
ZO1-1-L	120,812	98,355	43,276	0.44	35.82
ZO2-S0-L	269,116	121,738	82,780	0.68	30.76
ZO2-S1-L	128,228	89,636	64,537	0.72	50.33
ZO2-S2-L	119,758	97,803	67,815	0.69	56.63
ZO2-S3-L	147,993	125,982	91,966	0.73	62.22
ZO2-S5-L	140,796	101,079	75,806	0.75	53.84
ZO2-S6-L	92,609	70,843	51,006	0.72	55.08
ZO2-S7-L	317,125	132,739	74,332	0.56	23.44
ZW3-pillar-L	162,556	98,542	75,300	0.76	46.32
ZO2-purple-L	242,720	142,431	86,666	0.61	35.71
ZO2-extensometer-flint-L	151,588	38,942	20,249	0.52	13.36
ZO2-extensometer-L	132,143	110,635	36,509	0.33	27.63
Random ruble-L	152,295	142,393	45,564	0.32	29.92
Roof sample-L	114,564	107,965	17,273	0.16	15.08
Floor sample-L	109,070	107,960	20,512	0.19	18.81
New mine pillar-L	153,672	142,915	85,748	0.60	55.80
Central-L	119,948	103,841	59,189	0.57	49.35

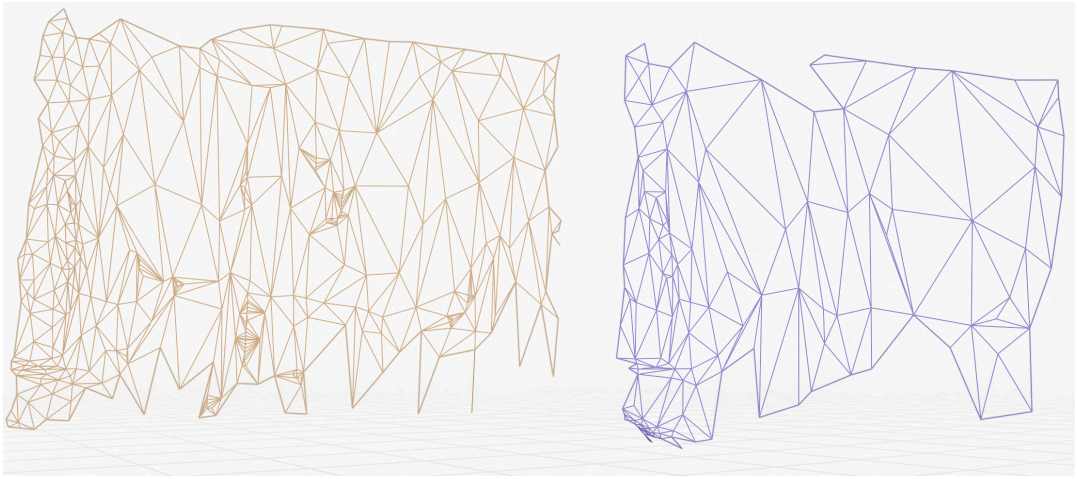


Figure 3.7: A selection of ZW3-pillar before (left) and after (right) optimization with a ratio of 0.76. Own work.

3.3. Discussing the optimized models

As shown in Figure 3.7, the overall shape of the wall has been preserved well. With larger and more complex structures, it is difficult to get a low ratio. Texture artifacts are still visible, and the polygon count per model after optimization is often still above the goal of 75,000 polygons. Some of these small texture artifacts can be seen in Figure 3.8, as not all the features are continuous. As shown by the red circles, the textures seem stretched and warped, which seems unrealistic.

Figure 3.8: Artifacts in the texture at ZO2-S5-L next to the extensometer, marked by the red circles. Own work.

These texture artifacts are caused by the UV map, which is explained in Section 3.1.1. However, to explain this texture artifact take a look at Figure 3.10. This shows the floor sample scan using a custom UV map created in Python, as can be seen in Figure 3.9. Here each fragment can be seen more easily. Due to optimization, the polygons move around all the time, and when this happens near the edge of a fragment, the UV map sometimes cannot interpolate the edge of that fragment properly, resulting in the stretching of the UV map on the 3D model (Villanueva, 2022). This is observed in Figure 3.10, where the most deformation in the textures is observed near the edges of the fragments. Although a method was created to minimize this type of visual artifact, this is beyond the scope of this thesis (et al., 2022). This also means that more fragmented UV maps have more fragment edges, and thus more potential to create texture artifacts. Thus, a UV map with as few fragments as possible is the most suitable for optimization.

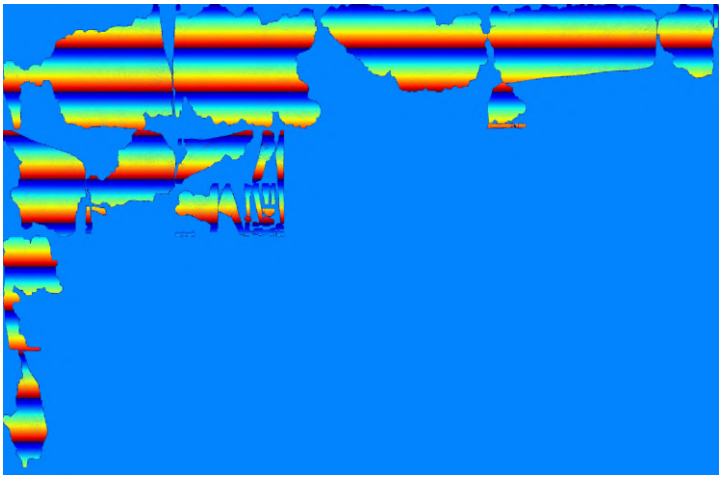


Figure 3.9: The custom UV map for the floor sample scan used for Figure 3.10. Every fragment is seen as a rainbow-like colored piece. Modified from NIANTIC, INC. (n.d.)

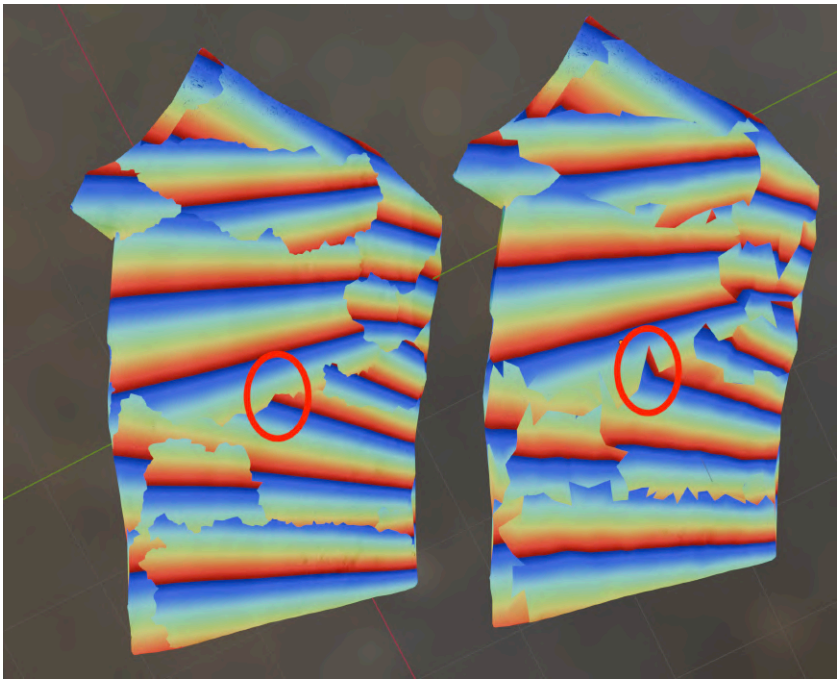


Figure 3.10: The impact of QEM optimization on the UV map textures on the floor sample scan, comparing the original model (left) with the optimized model (right). The red circle highlights one of the major changes between the two models. Own work.

As shown in Table 3.1, often many polygons are removed in the cleaning process. This is especially the case for the ZW3-pillar. The reason for this is due to it's complex rubble floor where measurement errors are located. This results in many parts to lie below the actual floor. Figure 3.11 highlights one of these many structural artifacts in orange. This view is from below the floor. Additionally, for ZO2-S7-L and ZO2-S0-L it is observed that during the cleaning of the model even more polygons are removed. This has to do with the fact that these two locations required two scans. Each scan covered roughly 75% of the pillar, so this explains why more than half of the polygons are missing in these two cases.

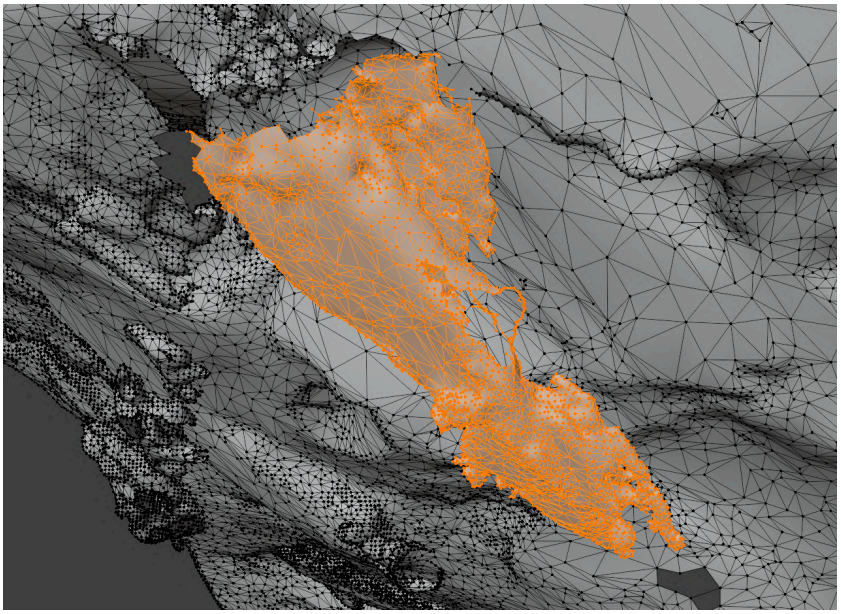


Figure 3.11: One of the many (floating) structural artifacts from the ZW3-pillar site being shown as orange. Own work.

Chapter 4
Evaluating the LiDAR data

This chapter starts off with the theory behind how different scans and rendering methods of a pillar compare with each other and assess the quality of the optimization (Section 4.1). These results are shown and visualized (Section 4.2), which are then discussed (Section 4.3).

4.1. Mesh evaluation methodology

4.1.1. QEM evaluation

First off, the quality of the QEM (Quadric Error Metrics) optimization is determined which were done in Section 3.1. This is done by calculating the minimum Euclidean distance between any point on the optimized mesh and the original mesh. This is calculated within Blender, as the geometry nodes option allows for simple code writing in the form of nodes. Figure 4.1 shows the code that has been used to calculate this closest distance. This example uses the ZO2_S0_L mesh. To start the coding, the optimized mesh is selected, and a new geometry nodes modifier is selected. From this, using object info, the original mesh is imported in *Object Info*, and with it's geometry and the *position* of the optimized mesh, the closest distance is calculated in *Geometry Proximity*. With all the minimum distances between any point between the two models calculated, the calculations are stored and added as an attribute to the optimized mesh in *Group Output*.

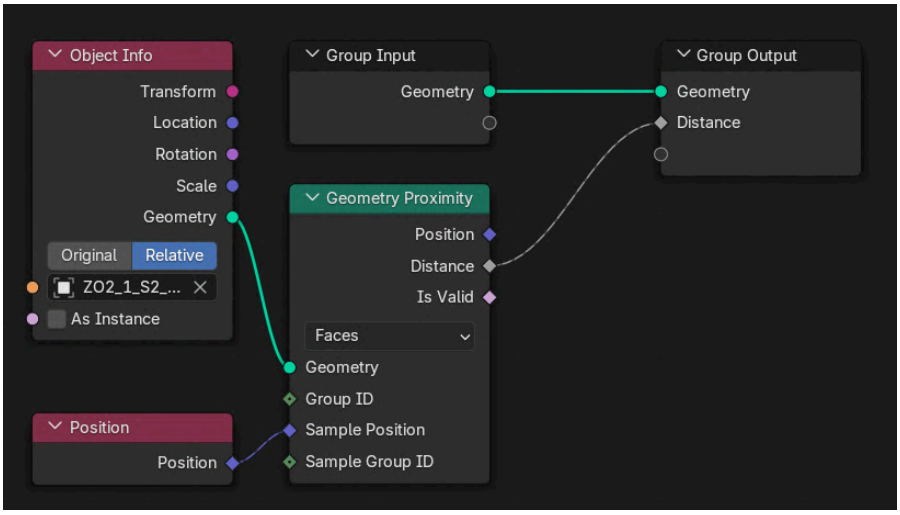


Figure 4.1: The geometry nodes within Blender for calculating the distance between the QEM optimized mesh and the original mesh. Own work.

This is done for all models, and using *The Super Duper Batch Exporter* models are exported as .ply files (Blender Foundation &, 2025). Using Python, all files are read and the minimum distances for each model are visualized in individual histograms.

Additionally, these results have been visualized on the 3D model itself. This is done to see in which regions there is the worst QEM optimization. Within the shading option of Blender, a new node code has been made which visualized the error attribute. The code can be seen in Figure 4.2. First, the attribute *DISST* is called, which is the error attribute. The errors are taken in a range of 0 to 1 mm,

which is done since most errors are initially very small. These values are then given a certain color based on the color ramp that is used, this is then assumed to be the texture of the model on the surface.

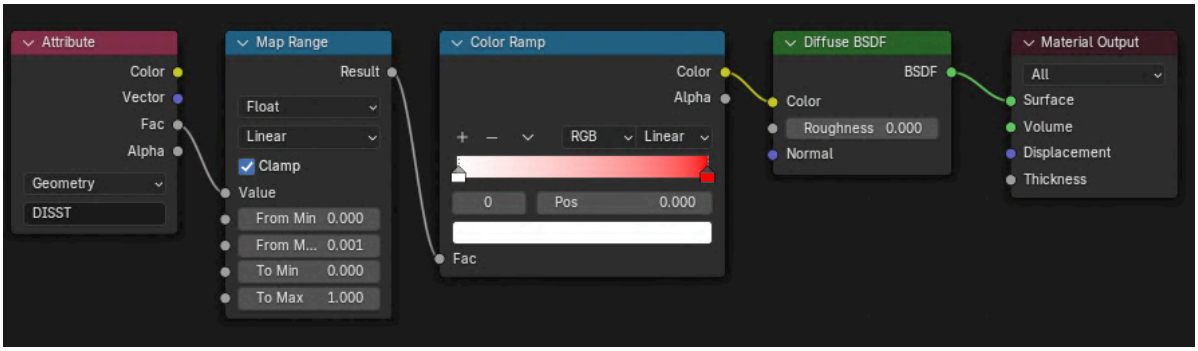


Figure 4.2: The shading code within Blender to visualize the error caused by QEM. Own work.

4.1.2. Scan range comparison

Preselected scanning options in Scaniverse are 0.8 (small), 2.5 (medium) and 5 (large) meters, but the user can select a custom range anywhere between these ranges. Since two iPhones were used to scan the pillars, two scan ranges were used, such that they can be compared afterwards. Since Scaniverse is not open source many of the scanning and processing methods are unknown, so it is useful to check if there are differences between scanning ranges. The quality of the medium range scan is based on a visual comparison based on texture quality and structure quality, as well as polygon counts. Based on these two criteria, it is determined whether the long range used in previous chapters is still sufficient of whether the smaller range scans are more appropriate for Virtual Reality.

4.1.3. Processing mode comparison

The processing modes provided in Scaniverse are also compared with each other. Scaniverse claims in their software that room-scale objects are most suitable for the *Area* processing mode, and thus for the previous chapters this mode has been used. However, there is also the *Detail* mode, which is said to have difficulties with large flat surfaces. Since it is unknown how Scaniverse processes the raw data in the different modes, the quality of the *Detail* processing mode is assessed for a singular location, which is ZO2-S0-L in this case, as that is the only pillar where there is also a full point cloud scan available, which is used later. The only thing mentioned in the Scaniverse app is that *Detail* mode uses photogrammetry, although it is assumed that it uses LiDAR together with photogrammetry (Noviana, Plank, & Wand, 2024) (Dick et al., 2025).

Photogrammetry is another scanning method that uses multiple images to measure the distance from the camera to a point or the deformation. To do 3D measurements, a stereo camera is needed (two cameras with a fixed distance from each other) that captures images periodically. The iPhone models used for scanning have three cameras at a fixed distance from each other (Baqersad, Poozesh, Niezrecki, & Avitabile, 2017). However, since it is unknown how Scaniverse processes the LiDAR scans it is not entirely known how the *Detail* mode works. The only thing that is known is that it probably does not use photogrammetry to do the 3D measurements but still uses LiDAR, as it still gives a normal result if only one camera is used during scanning. At the start of scanning, the Apple AR initializing sequence is shown, which uses LiDAR. Thus photogrammetry is probably used as a post-processing tool to reconstruct parts of the 3D mesh.

A visual assessment is made, looking at how detailed the textures and structure of the model are and how the model looks after it has been optimized, similar to Section 3.1. Afterwards, the optimized model is compared to the original model, similarly to Section 4.1.1, which is then compared to the results of the *Area*-processed model.

Based on the error caused by optimization, the number of polygons after optimization, the quality of the texture and structure of the model after optimization and other characteristics, it is determined whether the *Detail* mode is more suitable for Virtual Reality.

4.2. Results of the evaluations

As can be seen in Figure A.1 from Appendix A, histograms of the error have been made for every location. Corresponding box plots can also be seen in Figure 4.3. However, as discussed in Section 4.1.2 and Section 4.1.3, it is possible that other scan ranges or processing modes may be more suitable for Virtual Reality. Additionally, Appendix B visualizes the difference between the optimized and non-optimized models, with a scale that goes from white (0 mm) to red (1 mm) (as seen in Figure 4.2).

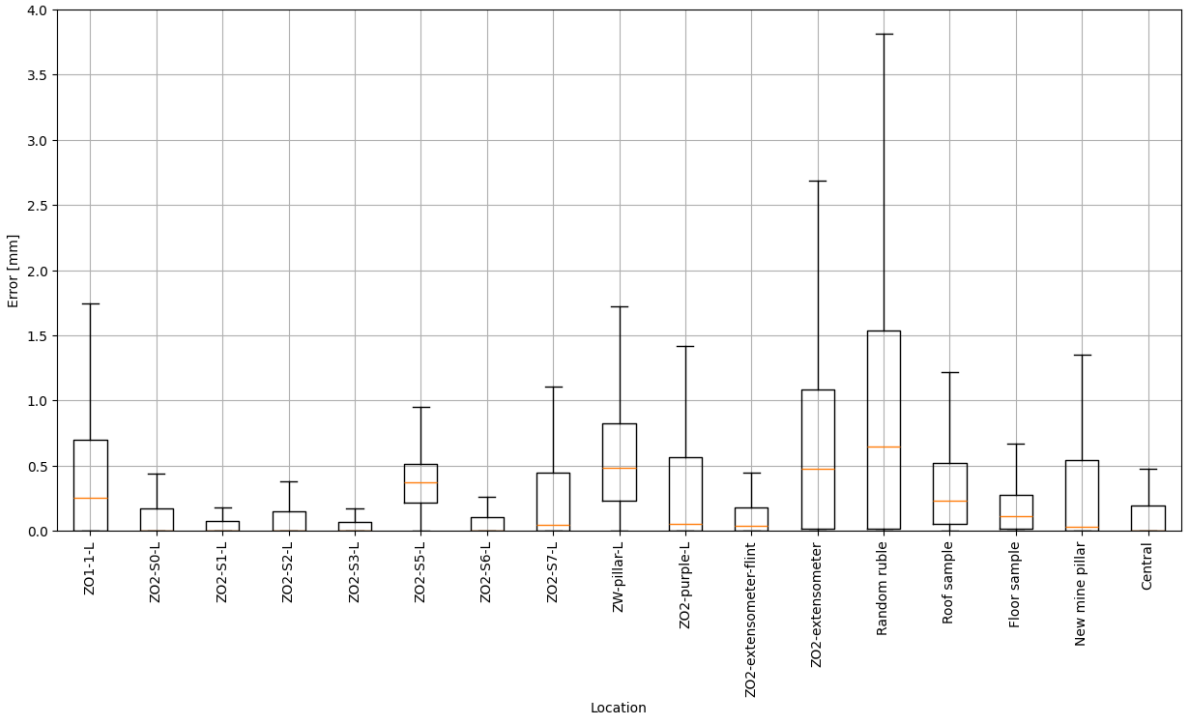


Figure 4.3: Box plots for all the scanned locations showing the closest distance between each point on the QEM optimized mesh and the original mesh. Own work.

The medium range scan of ZO2-S0 is initially visually very similar compared to the long range. The same features can be seen with the same textures. However, this scan has double the amount of polygons compared to the long range scan, as can be seen with other statistics in Table B.1 in Appendix B. Furthermore, as can be seen in Appendix B, the UV map of the medium range scan is slightly more fragmented than the long range scan, meaning that the surface is split up in many different texture parts.

Furthermore, using the *Detail* processing mode in Scaniverse, all models have been re-processed and optimized. The results of this can be seen in Table 4.1, which shows the original polygon count before any operations were done. The *Detail* models are processed similarly to the *Area* mode models in Section 3.1.3. The final two columns show the ratio used for optimization and the total ratio, which is the ratio between the final and original polygon count. For location ZO2-S0, the difference between the optimized and original mesh has been calculated, similarly to 4.1.1. Figure 4.4 shows the error caused by optimization for both processing modes. The mean error for detail mode is equal to 0.482 mm with a standard deviation of 0.367 mm. Appendix A shows the error and statistics for all other locations, including histograms (Figure A.2) and box plots (Figure A.3).

Table 4.1: The change in polygon count for all locations for the model using the *Detail* process option. Ratio used for QEM is the value used for the decimate option in Blender, while the total ratio is the ratio between the optimized and original polygon count.

Location	Original polygon count	Polygon count after manual cleaning	Polygon count after QEM optimization	Ratio used for QEM	Total ratio (%)
ZO1-1-L	425,185	425,185	51,022	0.12	12.00
ZO2-S0-L	612,609	412,143	41,213	0.10	6.73
ZO2-S1-L	320,318	320,318	51,250	0.16	16.00
ZO2-S2-L	317,534	317,534	53,980	0.17	17.00
ZO2-S3-L	308,741	308,741	77,185	0.25	25.00
ZO2-S5-L	312,070	311,629	65,441	0.21	21.00
ZO2-S6-L	320,829	320,829	54,539	0.17	17.00
ZO2-S7-L	652,235	519,093	77,863	0.15	11.94
ZW3-pillar-L	306,790	306,790	55,222	0.18	18.00
ZO2-purple-L	826,539	746,494	67,183	0.09	8.13
ZO2-extensometer-flint-L	48,954	45,954	6,364	0.13	13.00
ZO2-extensometer-L	168,167	168,167	25,239	0.15	15.00
Random ruble-L	276,859	276,859	24,917	0.09	9.00
Roof sample-L	140,097	140,097	4,202	0.03	3.00
Floor sample-L	44,039	44,039	3,082	0.07	7.00
New mine pillar-L	380,224	379,952	53,193	0.14	14.00
Central-L	116,879	116,879	19,869	0.17	17.00

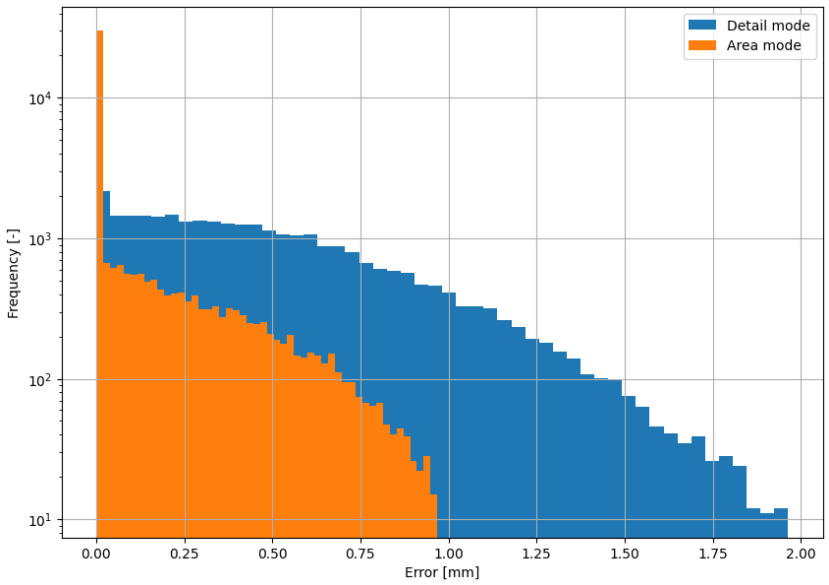


Figure 4.4: Histograms for the *Detail* and *Area* mode, showing the error caused by QEM for ZO2-S0. Own work.

4.3. Discussing the evaluation results

When the ZO2-S0 medium range scan was optimized using QEM with edge contraction, it became clear that due to the higher polygon count and fractured UV map there would more visual artifacts compared to the long range. Figure 4.5 shows this for part one of the pillars, where it can be seen that the P and marker are way more distorted for the medium range scan. This makes it not suitable for Virtual Reality, as higher polygon counts are needed for the medium range scan to achieve the same texture quality. Thus, the medium range scans are discarded for this thesis.

(a) The original shape of the P, before any optimization. Shown using the large range scan. Own work

(b) The P for the large range scan, optimized with a ratio of 0.10. Own work.

(c) The P for the medium range scan, optimized with a ratio of 0.10

Figure 4.5: Comparison of scan ranges showing part of the optimized ZO2-S1 model. The optimization is exaggerated to better show texture artifacts, and the actual models use higher optimization ratios.

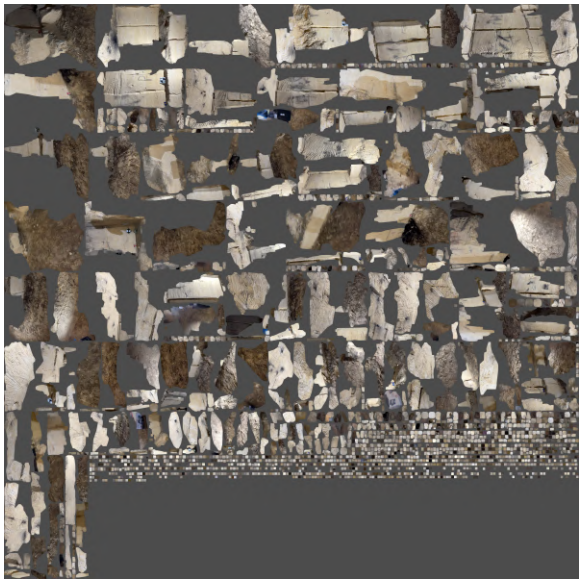
However, the *Detail* processing mode in Scaniverse gives more significant changes. As already observed from Table 4.1, the original polygon count varies significantly. This is expected, as the name *detail* does indeed imply that there are more details and thus more polygons. Cleaning was often not necessary, as this rendering mode seems to detect and remove floating artifacts very well, and fills most of the gaps present in the *Area* processing mode models (discussed in Section 3.3). Furthermore, the UV map is significantly different than prior UV maps, as this UV map is much less fragmented, as can be seen in Figure 4.6a and Figure 4.6b.

Using the *Detail* processing mode will have major impact on the quality of the QEM optimization. This is due to the fact that each surface in the figure is larger, and thus there is more flexibility for a mesh to move on the UV map without having to stretch the texture on the 3D model, as discussed in Section 3.3. This allows for much lower polygon counts without major texture artifacts, which can be seen in Table 4.1 and in Table 4.2, which shows a direct comparison. The error of optimization compared to the *Area* mode seems to be higher, as can be seen in Figure 4.4. *Detail* mode has an overall larger error. However, this is not a problem, as visual assessment is more important. The minor errors, which are all still smaller than 2 mm, are not visible in the 3D model, and thus do not result in a worse result. Figure 4.7 shows the comparison between the optimized ZO2-S3 model, focusing on a crack found on the wall. Here, it shows that the detail processing mode is less distorted compared to area processing mode. It can be said that *Detail* processing mode in Scaniverse provides better results for Virtual Reality, and thus from now on, only *Detail* mode models will be used.

Table 4.2: Polygon counts and the optimization ratio for the Detail and Area processing modes for ZO2-S2.

Processing mode	Original polygon count	Polygon count after manual cleaning	Polygon count after QEM optimization	Ratio used for QEM	Total ratio (%)
Detail	317,534	317,534	53,980	0.17	17.00
Area	119,758	97,803	67,815	0.69	56.63

(a) A less fragmented UV map obtained by the *detail* rendering mode.

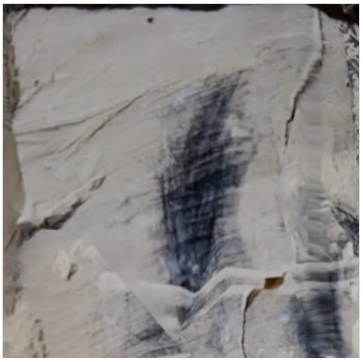


(b) A highly fragmented UV map obtained by the *area* rendering mode.

Figure 4.6: Comparison of UV maps between rendering modes for ZO2-S0. Obtained from NIANTIC, INC. (n.d.)

(a) The original crack in the wall without optimization. Shown using the detail model. Own work

(b) The crack after optimizing the detail model with a ratio of 0.25. Own work



(c) The crack after optimizing the area model with a ratio of 0.73. Own work.

Figure 4.7: Comparison between processing modes, showing part of the optimized ZO2-S3 model which shows a long crack in the wall.

One of the issues that is observed with all the models is how the permanent equipment, especially the extensometer, is that the model itself does not look very good, as can be seen in Figure 4.8. This bad texturing can be explained by the scanning method from Section 2.1.3. There it is explained that the scanner should be perpendicular to the wall that is scanned. However, for this elongated box there were no scan made from all sides of this box due to time constraints.

Figure 4.8: Poor textures on permanent equipment for ZO2-S5. Own work.

Finally, the point clouds from the GitHub app (Ryanphilly, 2021) are finally again. They could have been useful to compare with the mesh, which may have given some additional insight on how the rendering methods work. However, this does not add to the thesis, as the goal is not to figure out how Scaniverse works, but how to get suitable models for our VR application. This goal can be achieved without the raw point cloud data, and thus the point clouds will not be used in this thesis.

Chapter 5
Creating the Virtual Reality application

This chapter starts of with explaining factors that impact rendering performance and how to improve this performance, as well as explaining the workflow and the introduction of a survey (Section 5.1). This is followed by the results of a survey (Section 5.2) and the final virtual reality levels with their used performance optimization methods (Section 5.3), followed by the discussion of those results (Section 5.4).

5.1. Methodology behind the VR game

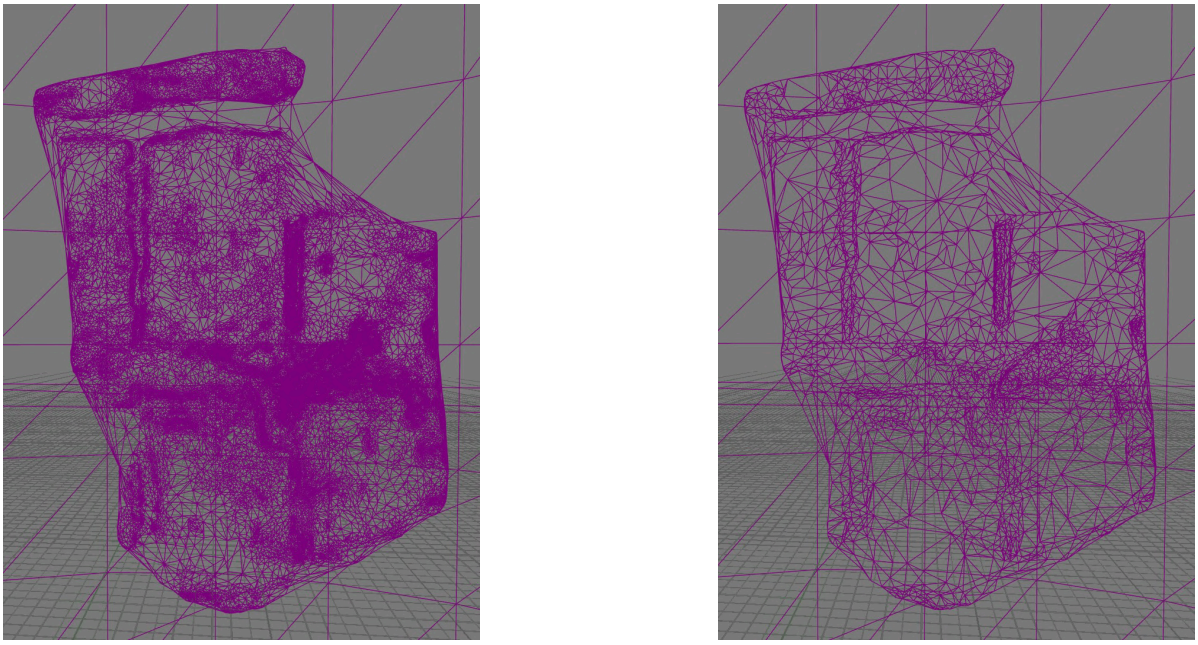
5.1.1. Performance

One of the key things to keep in mind when creating a video game is the performance. One of the main drivers that affects performance is polygon count. The Meta Quest 3 (Meta, 2024) is used for VROCK (Meta, 2024) recommends to have no more than 1.3 to 1.6 million polygons, as otherwise there will be drops in the frame rate. There are many more factors that impact the performance, which are discussed later, thus a maximum polygon count of 1 million that is rendered and seen by the VR headset is set as a maximum limit. When all the models are stitched together in the level with all the gaps in between filled with the floor and roof model, there are more than 1 million polygons in the level. Simply optimizing the models further is not an option, thus alternative methods are used to increase the performance.

Level of Detail

A common method used in the gaming industry to drastically increase performance is with the use of Level of Detail (LOD). When implemented, each 3D model contains multiple further optimized models. These lower polygon models replace the original model the further a player walks away from the model. The main issue with optimization in Section 3.1.3 is that after a while there are still artifacts in the texture due to the UV map. However, these artifacts are not clearly visible from far away, so this method does not affect the visual fidelity of the level. For LOD's, the amount of optimized models can be chosen, as well as how much each version is optimized and from how far away each version is rendered. The render distance is not determined by distance, but instead it is determined by how the ratio of how much it fills up the screen. For example, if a model perfectly fits on the screen, it gets a ratio of 1. However, if the player moves backwards, the ratio decreases as less screen space is filled with that certain model. The same can be said if the player walks forward, which results in a higher ratio. This way of determining the render distance is more logical than determining it by distance, as large objects would otherwise be simplified at the same time as small objects, while large objects are more visible. Thus, with the ratio small objects are simplified earlier as they take up less space on the screen. Figure 5.1 shows two LOD levels: when the player is close to the model (Figure 5.1a) and when they are far away from the model (Figure 5.1b). The ratios are automatically calculated in UE5.3 (Unreal Engine 5.3), and are the same for all models. As mentioned in Section 1.4, the Virtual Reality demo is made in Unreal Engine 5.3, often referred to as UE5.3, a widely used game engine (Epic Games, Inc., 2023a).

With UE5.3 this type of model optimization can be taken one step further using Nanite (Epic Games, Inc., 2023b), which works similarly to LOD but instead of optimizing the entire mesh all at once it clusters the mesh and then based on where the player is looking it optimizes those clusters without losing any details. However, this is not yet supported in VR, and thus is not used (Meta & Epic Games, 2023).



(a) LOD 0: The original mesh. Own work.

(b) LOD 4: The most optimized mesh. Own work.

Figure 5.1: Comparison between different LOD's. It is shown as wireframe, which shows polygons.

Level system

Another way to optimize performance is by splitting the game environment up into multiple smaller levels. A level in this scenario would be the world the player is in. Three smaller worlds can be chosen, or one large world. The word level will be used as it is a common term in the gaming industry, and using other would such as *room* could result in confusion. Not only does it significantly reduce the amount of polygons rendered in a level at any time, but from a game design perspective there are also some advantages:

- Each level could have its own learning objective. This could make the objectives clearer, and the level could be potentially more intuitive as it can be designed for that specific objective.
- The smaller levels may be significantly easier to traverse. In the mines, everything looked similar and it was relatively easy to get lost. This may also be the case in VR, and thus smaller levels could help.
- The smaller levels can reduce the time needed to traverse the mines, as players can be teleported from level to level after completing an objective. As time for the player might be limited, this can be quite useful. Additionally, a feature could be implemented that allows players to travel from level to level without completing the objective if desired.

Splitting the game environment up into smaller levels would be an obvious choice. However, multiple levels also means that there are loading times when moving between levels. Due to these concerns, an survey is held where people with varying virtual reality and game familiarity play through a simplified demo of the game. Based on obtained statistics and feedback it can be determined which method would be the most suitable for VROCK.

Lighting and shaders

The previous two parts are about how to optimize performance. However, there is one major factor that significantly decreases performance and needs to be minimized: lights and shading. In a virtual world, a light source has two very important factors: the intensity (brightness) and the attenuation (range) of the light. The game has to compute the amount of light received at any surface within the range of the source, and do that for every frame. When more than one lighting source is needed, which is the case for the Valkenburg mines, things could get difficult. There need to be thought of a method of lighting placement that minimizes the performance impact while keeping everything decently lit. UE5.3 has the

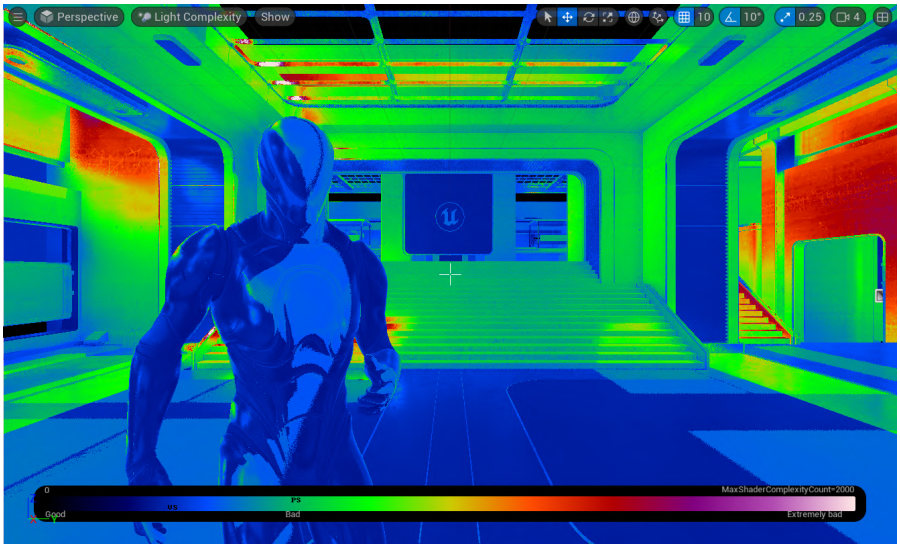


Figure 5.2: Light complexity as seen in UE5.3, where the scale goes from least (left, black) to most (right, white) complex lighting. Retrieved from Epic Games, Inc. (2023c).

option to view the level in *light complexity mode*, which grades how complex a pixel is to render based on the overlap of lights (Epic Games, Inc., 2023c). Multiple different light source types have seen used:

- Point source, which works similar to a normal light bulb which emits light in all directions from a single point.
- Rectangular source, which works similar to most generic office ceiling lamps as it emits light from a surface in the orientation of the panel.
- Directional light, which is similar to the light received from the sun, as it is a very large light source extremely far away.

Additionally, to minimize the performance impact even more, the source mobility is also looked into. Each light source has a mobility type: static, stationary or movable. Static and stationary lights are sources that cannot move, and the shadows and lighting are calculated beforehand and baked onto static textures. Movable lights can move and thus cannot calculate shadows beforehand, and thus will not be used as having to constantly calculate the shadows for static objects is computationally demanding. The main difference between static and stationary is how dynamics objects are lit and how their shadows are projected. With static lighting, a volumetric light map is also calculated beforehand. This works similar to calculating lighting and shadows on static textures, but now that information is calculated for many points in a volume. Together with interpolation the lighting and shadows of a dynamic object can be estimated. Stationary sources on the other hand directly calculates lighting and shadows on dynamic objects. This is computationally more heavy, but gives more accurate results. Based on quality and performance, one of the two mobilities is chosen which will be the most suitable for our VR application.

5.1.2. Preparing and exporting models to Unreal Engine 5.3

There are two ways the game level can be made: Either stitch all the models together in Blender and export the level as one file to UE5.3 or export all the models to UE5.3 and stitch all the models together there. In general, Blender is more suitable for stitching models together, as the program is mostly designed for modeling and altering models, which is not necessarily the case for UE5.3. However, as discussed in Section 5.1.1 working with LOD is preferable as it drastically increases performance. However, in the case that one large stitched model is imported into UE5.3, the LOD will not function properly, as the player will always be close to the model. Options such as Nanite (Epic Games, Inc., 2023b) would have been ideal in this case, however as discussed in Section 5.1.1 this is not possible. Thus, it is chosen to import all models in UE5.3 individually and create the level there.

However, before that is done, the models need to be prepared in Blender. Since modifying the mesh

in UE5.3 is not an option for this project, it must be ensured that all models are ready to be stitched together with smooth transitions between models, to try to make the level as realistic as possible. To achieve this, the dedicated floor and roof mesh have been altered slightly to give them dome shaped edges, allowing them to better fill all the gaps between POI's (Point Of Interest) and creating a clear and realistic path. Additionally, the mine floors, which are often partially scanned at POI's have been altered such that it is always at $z = 0$ meters, which allows for easier filling of the gaps with a smoother transition between models. This is done for all models.

The models are now ready for exporting. The models will be exported using *The Super Duper Batch Exporter* Add-on in Blender (Blender Foundation &, 2025), but there are still some slight changes that need to be made. For all models, LOD's are calculated. In total, for each model there are 4 LOD's, going from ratios of 0.8 to 0.5 to 0.2 to 0.1. Finally, based on the markers that were placed on the pillars, it seems that the scale of the models is off, and thus all the models are scaled. The cardboard markers were 10 by 10 cm, while in the model they were 7.7 by 7.7 cm, thus all the models are scaled by 1.3. This error can be traced all the way back to Scaniverse (NIANTIC, INC., n.d.), where the built-in measuring tool shows that the markers are also 7.7 by 7.7 cm. Finally, the models are all exported as .fbx files, which are specifically designed for game engines such as UE5.3.

5.1.3. Creating and enhancing the game

Now that all the models are exported into Unreal Engine, it is time to construct the level. There is a set of predetermined objectives that need to be completed by the player. However, as mentioned in Section 1.2, there is a whole team that makes the actual VR application. Thus this thesis will not concern itself with the final objectives, but instead will create levels based on those objectives to assess performance in a VR game demo. These objectives include:

1. Assessing and classifying pillars based on Figure 5.3, which shows an increase in structural damage per class. This is afterwards compared with the classification of Bekendam (2008).
2. Using a Schmidt Hammer to measure strength of different rocks types in the mines (shown in Appendix F).
3. Look and read off values from permanently installed monitoring equipment in the mines (shown in Appendix F).

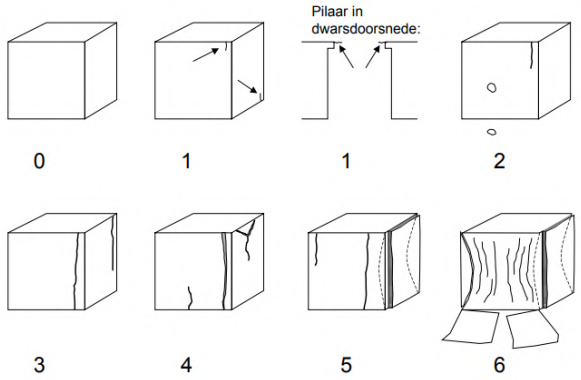


Figure 5.3: The classification used to determine the structural damage of a pillar. Retrieved from Bekendam (2008)

.

For objective one, the pillars need to be located in their original locations based on the mine. For every scan of these pillars, an adjacent pillar was also scanned (often ZO2-S0) which now allows for easier placement instead of having to rely on Figure C.2. After the pillars have been placed in their correct location, the empty space on the floor and roof is filled up with the previously obtained roof and floor sample scans. These are then slightly stretched such that the transition between the floor/roof and pillar models is as smooth and realistic as possible. Finally, a wall sample from ZO2-S2 is taken, which is then used to enclose the entire level. Since this wall is mostly flat, it can be simplified much further, down to 424 polygons. This is possible because the UV map is very simple there and not fragmented,

and thus it can be simplified a lot without obtaining visual artifacts. This is also done for objective two and three. As discussed in Section 5.1.1, if three levels or one level is used will be determined by the results of the survey. However, since a level-system has the most potential it is created first. Figure 5.4 shows a schematic flowchart from the entire process of preparing and importing models into Unreal Engine and how to create the levels.

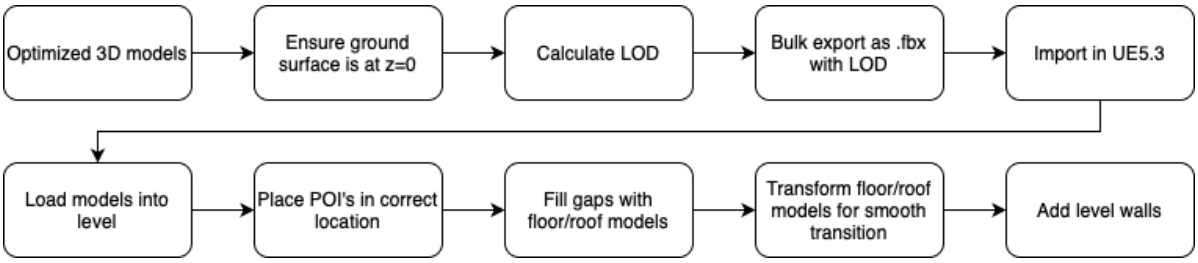


Figure 5.4: Workflow from Blender to UE5.3. Own work.

Additionally, several features have been implemented in the game to enhance the player experience. These features are briefly shown in the list below, and can be seen in Appendix F:

- A minimap of the level has been created which shows a top down view of the level with the location and orientation of the player. POI's are labeled on the map. This map can be accessed by going to a menu which can be accessed by the player. From this menu, players can also move between levels and exit the game.
- Each pillar has been given labels similar to street signs. This has the same purpose as the minimap, but may be easier for navigation close by.
- Besides the levels within the mines, a hub level is created. This is the starting point, and players get familiar with VR and the VR controllers. However, players can teleport to any level at any point, regardless of location.
- A headlamp is added. It provides some additional lighting which may help with visibility. It can be toggled on or off.
- There is an option for a second player, which can be a spectator. They can control a flying camera on the connected laptop to aid the player in VR and write down results. As the VR player is aware of their surroundings, the spectator has their webcam enabled and shown on the camera model. This can allow for better communication.

There is a final method to potentially improve the player experience. This is by trying to improve the 3D models once more. Simple QEM (Quadric Error Metrics) optimization simplifies the model, and slightly rounds most of the sharp corners that are present. The models can be potentially improved by classifying regions with sharp edges and then reprocessing (Wang, 2006). However, it may be difficult for players to even notice the difference, so before this method is implemented it is tested whether players even notice smoothed edges to begin with. Thus, from the survey held in Section 5.1.1 it is asked to players whether they even notice the slightly rounded corners.

5.2. Survey results

Based on the feedback from the survey from Section 5.1.1, many things can be observed and changed. In total, 10 participants played through the demo with a large range of age, VR experience and overall game experience. All the results can be seen in Appendix D. Overall, players were quite satisfied with the movement in the demo compared to their previous experience with the original VROCK, as the movement in the demo is teleporting to locations where the user is pointing instead of gradually walking forwards. Most players favored the level system in the demo, as it shows clear progression. One single level would be too large, as the size of the largest level was already reaching it's limit in terms of easy navigation. The loading time between levels is so short that it can be ignored. Due to the map added for navigation, most players were not having trouble with navigating from POI to POI. Some players thought the levels were too dark, while others though they were too bright. Although the

position and range of the lights cannot be drastically changed as that will decrease performance, the intensity and range of the headlamp can be increased, which was generally well received.

5.3. The resulting VR application

5.3.1. 3D levels & models in Unreal Engine

With the results from Section 5.2 showing that a level-system is preferred, these levels are created using the models imported using the workflow from Figure 5.4. In total, there are four levels in the demo based on the objectives from Section 5.1.3, with an additional level which is a hub world where the players start their experience and understand the basic controls. Please note that these are not the final objectives of the new VROCK experience, as mentioned in 5.1.3. The figures from Figure 5.5 show a preview of each level.

5.3.2. Level Optimization results

For all the levels, a combination of point and rectangular light sources have been taken. To increase realism, all the lights are placed at the corners of the pillars, and if that light is close to the bounding wall of the level, it will be a rectangular source, otherwise a point source. This slightly decreases the impact on performance as the bounding walls do not need to be well lit. All light sources are chosen as static lights. Although the shadows of moving objects may be less detailed as the reflections are not as good, it is still plenty good enough for VROCK. Figure 5.6 visualizes the most complex level when it comes to lights. The further a color is in the spectrum, the more light overlap and thus the more complex an area is. As can be seen, almost all areas are green or blue making them not complex.

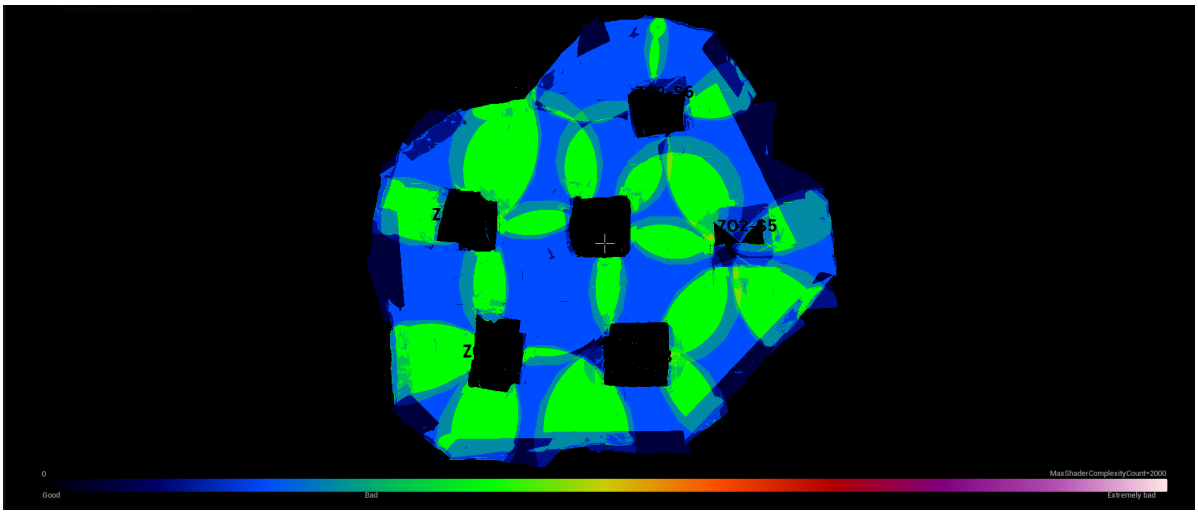


Figure 5.6: The light complexity in level 1 from a top-down view, which is the most complex level as there is a lot of light source overlap. The scale goes from least (left, black) to most (right, white) complex lighting. Own work visualized in UE5.3.

The highest amount of polygons rendered at any point in time is a total of 465,478 polygons with an average around 300,000 polygons, which is well below the goal of 1 million set in section 5.1.1. This is partially thanks to the LOD optimizations. Although it is difficult to say how much LOD's actually reduces the amount of rendered polygons as it highly depends on location, it is estimated to be around a reduction of 150,000 polygons on average.

(a) The hub world with the basic control information.

(b) Level 1 with the pillars in their original place.

(c) Level 2 with ZO2-Purple and it's unstable wall.

(d) Level 3 with the two pillars with permanently installed equipment.

Figure 5.5: Previews of all the different levels. All figures are own work visualized in UE5.3.

5.4. Discussing the results

The main issue with the levels is still the transitions between the POI's and the floor and roof textures. Sometimes, the floor of some POI's is raised, making it difficult to add a convincing floor next to it as there is only a flat floor model. This can be solved by having more types of floor and roof models that may be curved, however this is beyond the scope of this thesis, as the thesis focuses on making a playable demo that looks and performs good and there is no focus on smaller aesthetic details.

Since the highest number of polygons rendered at any point is still below than the recommended amount, there is a lot of wiggle room for other features. Think of increasing polygon count of the models for even better representation, or more overlap in the lighting. However, it is still chosen to have the highest performance possible. This is due to the fact that it allows the VR application to run on older hardware. It is able to run on the Meta Quest 2, which should be able to render 750,000 to 1,000,000 polygons at a time (Meta, 2024). Theoretically it should also be possible to run it on the Meta Quest 1, which allows 350,000 to 500,000 polygons at a time (Meta, 2024), although this is really reaching the limits of the hardware and may result in frame drops in the worst case scenario.

Other problems that were found were also mostly issues that can be fixed but are outside the scope of the thesis. For instance, some players did not like the movement system of teleporting where the player points towards and preferred having the option to change it, or some found that holding a tool such as a Schmidt Hammer (although in the demo it was a template gun) for long periods of time would result in a cramped hand and would rather toggle to grab and hold instead of hold.

Chapter 6
Conclusion and Recommendations

6.1. Conclusion

The goal of this thesis was collecting, optimizing, evaluating and visualizing LiDAR data from the Valkenburg mines. This was done to add a new part of the VROCK experience, which is a virtual reality application used by bachelor students in Earth, Climate & Technology at the TUDeflt. Based on this information, the following main research question was formulated:

How can LiDAR data and meshes from the Valkenburg mines be obtained, optimized and visualized in Virtual Reality while preserving structural and visual fidelity for VROCK?

This research question was split up in multiple sub-questions, and the conclusion of the main question will be split up based on these sub-questions.

6.1.1. LiDAR data collection

How can LiDAR data be acquired in the Valkenburg mines, and what challenges can be encountered during the scanning process?

Points of interest in the mines underneath Valkenburg were scanned using the LiDAR scanner on two iPhones, as they are easy to use and provide high quality textures. Huge light sources and small markers were brought along to illuminate and identify the points of interest, although sometimes multiple scans of a pillar were needed due to them not being illuminated enough. Scaniverse was the main software used to scan the points of interest, as it provides high quality meshes and gives the option for different render modes, although it does not provide the raw LiDAR point cloud data. Overall the obtained unaltered 3D meshes were visually quite good, but in some cases the lighting was off or parts of the mesh were missing. Although the iPhone LiDAR scanner has a lower resolution compared to other scanners such as the Leica P40, it more appropriate for the application of scanning POI's underneath Valkenburg, as it is versitile and texture quality is good.

6.1.2. 3D mesh optimization

How can the LiDAR model be optimized to meet VR performance requirements while maintaining visual and structural fidelity?

Multiple mesh optimization methods were compared and quadric error metrics with edge contraction was used as it is a well known and widely used method that works well with the obtained 3D meshes. It works by contracting two vertices connected by an edge, and every possible contraction has their cost calculated beforehand and is then ranked, such that during optimization the contractions with the lowest cost are contracted first. These calculations were performed within Blender, where models first had to be cleaned up of measurement errors. There were two factors that impacted how much a model could be optimized: Structural and texture artifacts, with the latter being the most influential. Due to the UV map, artifacts appeared in the textures of the model, stretching them and making them discontinuous. Although this impacted optimization, the goal of a model having less than 75,000 polygons was often reached. Besides using a mesh model, a Gaussian splat model was also used to see if it could be an alternative to the mesh. However splats were not suitable for the Valkenburg mines, as many small details get lost and everything becomes hazy.

6.1.3. Optimized 3D mesh evaluation

How can the optimized 3D meshes be improved further to be more suitable for a VR model?

The quality of the quadric error metrics with edge contraction was calculated by calculating the distance between all the points on the original and optimized model, which was done through blender. Based on these results it was found that most errors were no more than 2 mm, which is barely visible on a 3D model. Additionally, different scan ranges and processing modes in Scaniverse were compared with each other using ZO2-S0. The lower scan range resulted in a higher polygon count and more fractured UV map, and thus resulted in an overall higher polygon count after optimization, making it not suitable. However, the different processing mode called *Detail* had a higher polygon count but significantly less fractured UV map, allowing for lower polygon counts after optimization as well as a higher detailed model in general. Although the rendering method is not entirely known, it is known is that it uses LiDAR together with photogrammetry. Thus, from then on the *Detail* models using the long scan range were used as they were most suitable for VROCK.

6.1.4. 3D mesh visualization in VR

How can the final 3D models of the Valkenburggroeve be implemented in virtual reality?

The optimized *Detail* models were exported out of Blender with their LOD calculated. They are imported in Unreal Engine, which is the game engine used for VROCK. Here, further optimizations were thought of and implemented within the game engine. For instance, the light placement had to be considered as it impacts performance, and there was thought of having a level system instead of one large world the player walks through. Based on surveys where players perform simple tasks in a demo, three smaller levels were used instead of a large one. Additionally, multiple simple features were added such as a minimap to ease navigation as well as many other things such as a headlamp.

6.2. Recommendations

This thesis handled a broad range of topics, following the entire process from data collection to data visualization. However, the VROCK project is far from over, and it is expected that the current models as well as some of the levels are sufficient enough to be used by the team at the XRZone to create the actual game with it's objectives fully worked out. However, there are some things future research can look into, as shown in the list below:

- Currently it is unknown how Scaniverse processes it's 3D meshes, and how photogrammetry is used. Therefore, it is recommended to use the obtained raw point cloud data to mesh it in Python as well as use quadric error metrics with edge contraction in Python, allowing the automation of the optimization of 3D LiDAR scans. This can be done with the already obtained point cloud data, which can then be compared with the current models to see if the new model is more suitable for VR. If that is the case, more point cloud data can be collected.
- Currently there is a way to numerically assess the structural quality of the optimization, but the texture quality has to be visually assessed. This is not optimal, as there can be many inconsistencies this way. Therefore, it is recommended that a method is designed that can numerically assess the texture quality of a model after it has been optimized. For this, further research needs to be done for UV mapping. This could also help with automating the optimization process.

References

Baqersad, J., Poozesh, P., Niezrecki, C., & Avitabile, P. (2017). Photogrammetry and optical methods in structural dynamics – a review. *Mechanical Systems and Signal Processing*, *86*, 17–34. Full-field, non-contact vibration measurement methods: comparisons and applications. doi:https://doi.org/10.1016/j.ymssp.2016.02.011

Bekendam, R. F. (2008, May). *Inventarisatie stabiliteit en milieu- onvriendelijke situaties in de gemeentegroeve en de monstergrot* (No. GeoControl rapport M00811).

Blender. (2023). Bmesh decimate collapse blender. Retrieved May 20, 2025, from https://github.com/blender/blender/blob/5bab6126c1beb1bb1959c03e69f476012510a5b4/source/blender/bmesh/tools/bmesh_decimate_collapse.cc

Blender. (2025, March 18). Blender (Version 4.4). Retrieved May 26, 2025, from https://www.blender.org

Blender Foundation, &, B. [BastianLS]. (2025, April 8). Super duper batch exporter. Retrieved May 27, 2025, from https://extensions.blender.org/add-ons/superduperbatchexporter/

COAST Products. (n.d.). Wph30r. Retrieved June 30, 2025, from https://coastportland.com/products/wph30r?srsltid=AfmBOop3q3kd3nVxc3oHb9Qln_1Uc-csxeUPvIqiEU2zmFB1rajiTCCV

Dick, D., Hum, K., Smith, L., Street, K., Hansen, J., Schotanus, P., … Eyles, C. (2025). Improving accessibility of in-situ paleontological geoheritage via digital conservation: A case-study using ipad-based lidar in the niagara aspiring geopark. *Geoheritage*, *17*(2). doi:10.1007/s12371-025-01127-z

Epic Games, Inc. (2023a, September). Unreal engine (Version 5.3). Retrieved June 2, 2025, from https://www.unrealengine.com/en-US/blog/unreal-engine-5-3-is-now-available

Epic Games, Inc. (2023b, September). Unreal engine nanite virtualized geometry (Version 5.3). Retrieved June 2, 2025, from https://dev.epicgames.com/documentation/en-us/unreal-engine/nanite-virtualized-geometry-in-unreal-engine?application_version=5.3

Epic Games, Inc. (2023c, September). Viewport modes (Version 5.3). Retrieved June 4, 2025, from https://dev.epicgames.com/documentation/en-us/unreal-engine/viewport-modes-in-unreal-engine?application_version=5.3&utm_source=editor&utm_medium=docs&utm_campaign=rich_tooltips#lightcomplexity

Esterhuizen, G. S., Dolinar, D. R., Ellenberger, J. L., & Prosser, L. J. (2011). Pillar and roof span design guidelines for underground stone mines.

et al., W. Z. (2022). Seamless simplification of multi-chart textured meshes with adaptively updated correspondence. *Computers & Graphics*, *106*, 77–87. doi:https://doi.org/10.1016/j.cag.2022.05.021

Garland, M., & Heckbert, P. S. (1997). Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on computer graphics and interactive techniques* (pp. 209–216). doi:10.1145/258734.258849

Hoppe, H. (1999). New quadric metric for simplifying meshes with appearance attributes. In *Proceedings visualization '99 (cat. no.99cb37067)* (pp. 59–510). doi:10.1109/VISUAL.1999.809869

Interface Medien GmbH - www.interface-medien.de. (n.d.). Professional mobiele led accuspot bs 5000 ma, bosch professional 18v system, 6000lm, ip55 | brennenstuhl®. Retrieved June 30, 2025, from https://www.brennenstuhl.nl/nl-NL/producten/led-bouwlamp-werklamp/professional-mobiele-led-accuspot-bs-5000-ma-bosch-professional-18v-system-6000lm-ip55

Kim, Y.-T., & Kouh, H.-J. (2022). A study on deep learning based simplification for lightweighting of underground geospatial information data. *IEEJ Transactions on Electrical and Electronic Engineering*, *18*. doi:10.1002/tee.23745

Leica Geosystems. (n.d.). Leica scanstation p40 / p30 - 3d-laserscanoplossing. Retrieved May 20, 2025, from https://leica-geosystems.com/nl-nl/products/laser-scanners/scanners/leica-scanstation-p40--p30

Low, K.-L., & Tan, T.-S. (1997, January). Model simplification using vertex-clustering. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics, SI3D '97, Providence, RI, USA, April 27-30, 1997* (pp. 75–82, 188). doi:10.1145/253284.253310

Luetzenburg, G., Kroon, A., & Bjørk, A. (2021). Evaluation of the apple iphone 12 pro lidar for an application in geosciences. *Scientific Reports*, *11*. doi:10.1038/s41598-021-01763-9

Meta. (2024, October). Testing and performance analysis. Retrieved June 2, 2025, from https://developers.meta.com/horizon/documentation/unity/unity-perf/?locale=en_US

Meta, & Epic Games. (2023, March). Announcing official support for unreal engine 5. Retrieved June 3, 2025, from https://developers.meta.com/horizon/blog/updates-developers-unreal-engine-5/

NIANTIC, INC. (n.d.). Scaniverse | free 3d scanner | capture & explore with gaussian splatting (Version 5.0.1). Retrieved May 20, 2025, from https://scaniverse.com/

Noviana, E., Plank, U., & Wand, E. (2024). Photogrammetry and iphone lidar in remote places for preparation of 3d-objects in a virtual museum. In *Informatik 2024* (pp. 1007–1016). doi:10.18420/inf2024_90

Polycam. (2021). Cross-platform 3d scanning floor plans & drone mapping. Retrieved May 20, 2025, from https://poly.cam/

Principia Mathematica, Inc. (2004). Objects in obj format. Retrieved June 15, 2025, from https://www.prinmath.com/csci5229/OBJ/index.html

, R. [Ryanphilly]. (2021, August). Github - ryanphilly/ios-pointcloud: Create, save, and export point clouds w/ lidar equipped iphones. Retrieved May 20, 2025, from https://github.com/ryanphilly/IOS-PointCloud#

Taka, V. (2024, September 12). *3d gaussian splatting theory and variance rendering extension*.

Villanueva, N. (2022). Uv mapping. In *Beginning 3d game assets development pipeline: Learn to integrate from maya to unity* (pp. 117–149). doi:10.1007/978-1-4842-7196-4_5

Vosselman, G., & Maas, H.-G. (Eds.). (2010). *Airborne and terrestrial laser scanning*. Whittles Publishing.

Wang, C. C. (2006). Incremental reconstruction of sharp edges on mesh surfaces. *Computer-Aided Design*, *38*(6), 689–702. doi:https://doi.org/10.1016/j.cad.2006.02.009

Appendix A
QEM optimization comparison statistics

This appendix shows the statistics obtained from the comparison done in Section 4.1.1. Table A.1 shows the mean and standard deviation of the obtained data and Figure A.1 shows the histogram for each location of the obtained data.

Table A.1: Table showing the mean and standard deviation for the closest distance between each point on the QEM optimized mesh and the original mesh for all locations for the *Area* mode.

Location	Mean [mm]	Standard deviation [mm]
ZO1-1-L	0.49	0.49
ZO2-S0-L	0.24	0.24
ZO2-S1-L	0.18	0.18
ZO2-S2-L	0.22	0.22
ZO2-S3-L	0.19	0.19
ZO2-S5-L	0.25	0.25
ZO2-S6-L	0.21	0.21
ZO2-S7-L	0.36	0.36
ZW3-pillar-L	0.42	0.42
ZO2-purple-L	0.44	0.44
ZO2-extensometer-flint-L	0.14	0.14
ZO2-extensometer-L	0.75	0.75
Random ruble-L	1.10	1.10
Roof sample-L	0.38	0.38
Floor sample-L	0.22	0.22
New mine pillar-L	0.44	0.44
Central-L	0.17	0.17

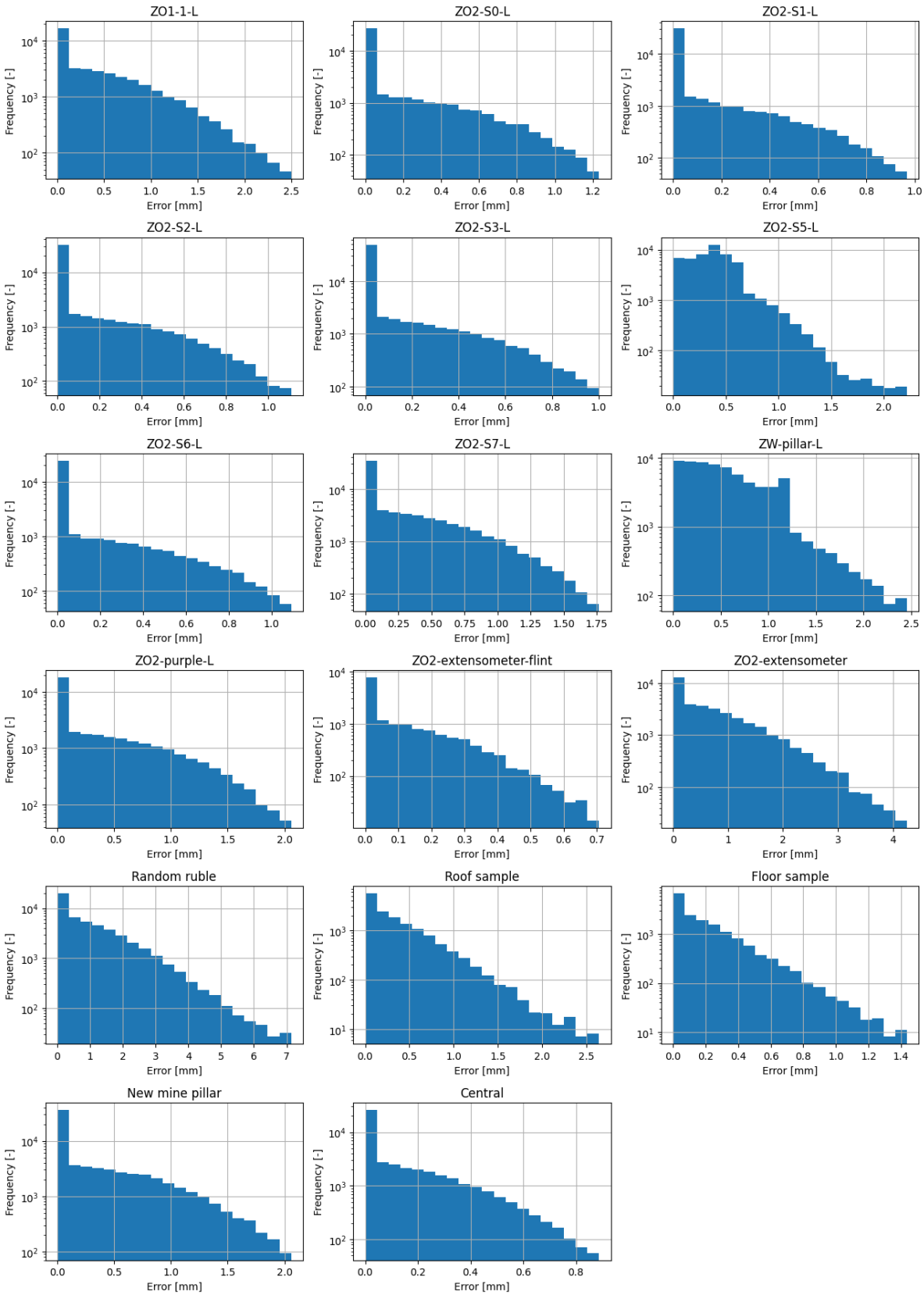


Figure A.1: Histograms of all the scanned models using the *area* processing mode, showing the closest distance between each point on the QEM optimized mesh and the original mesh. Own work.

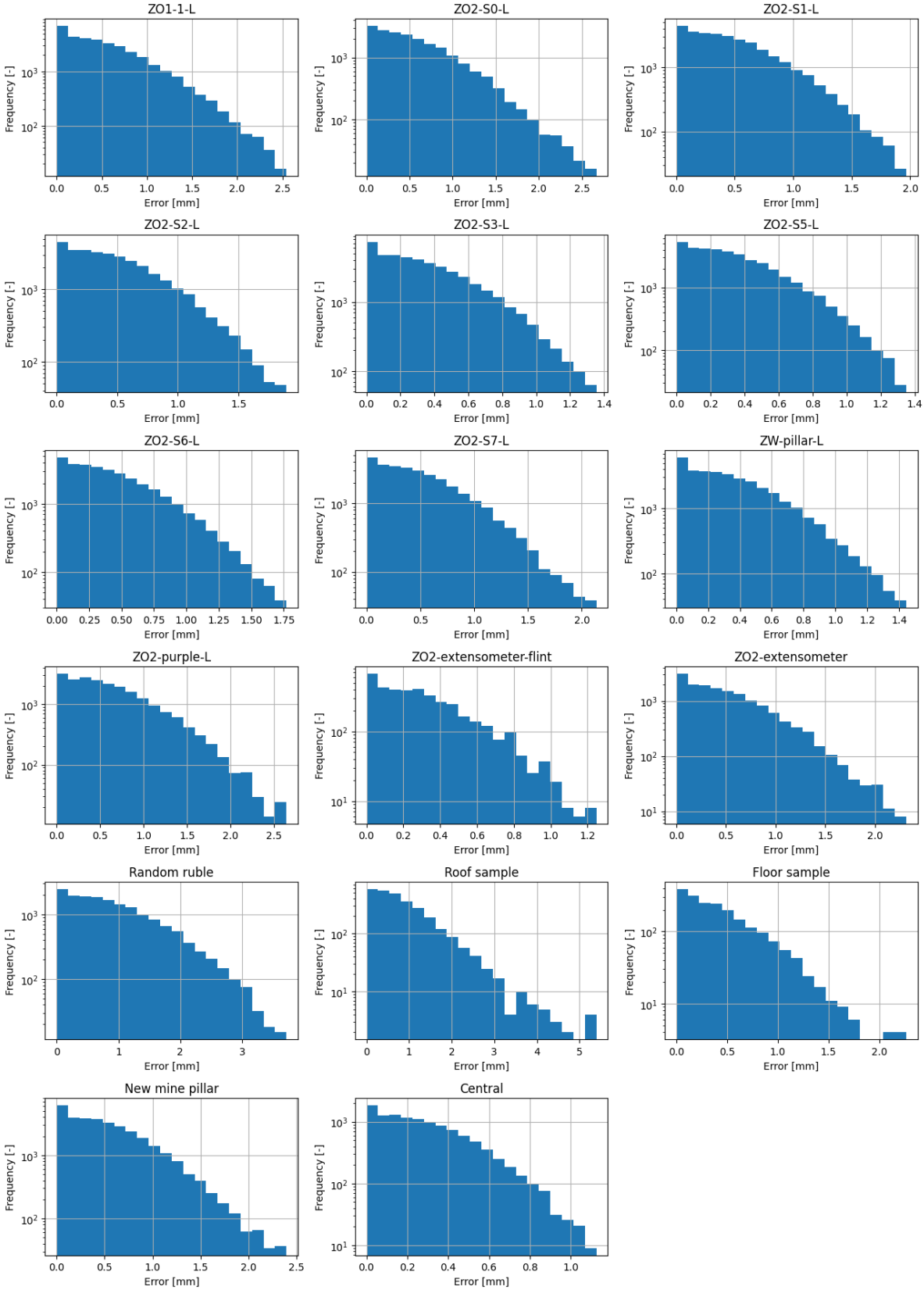


Figure A.2: Histograms of the scanned models using the *Detail* processing mode, where each plot shows the error caused by optimization. Own work.

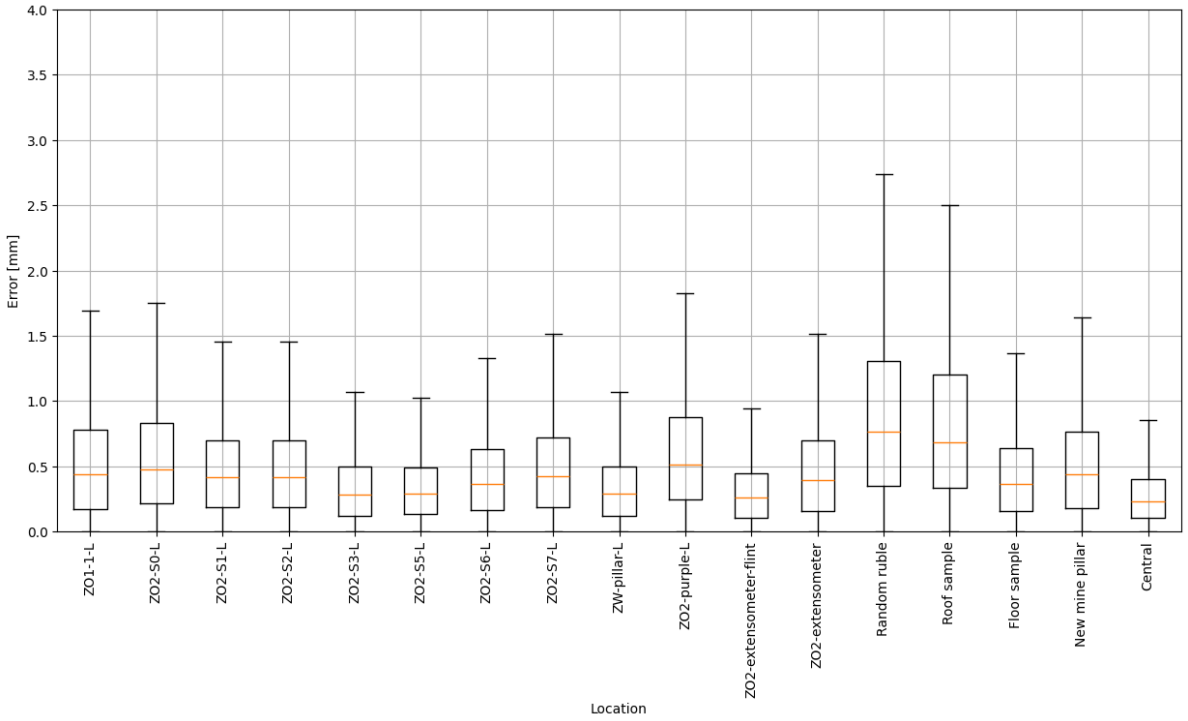


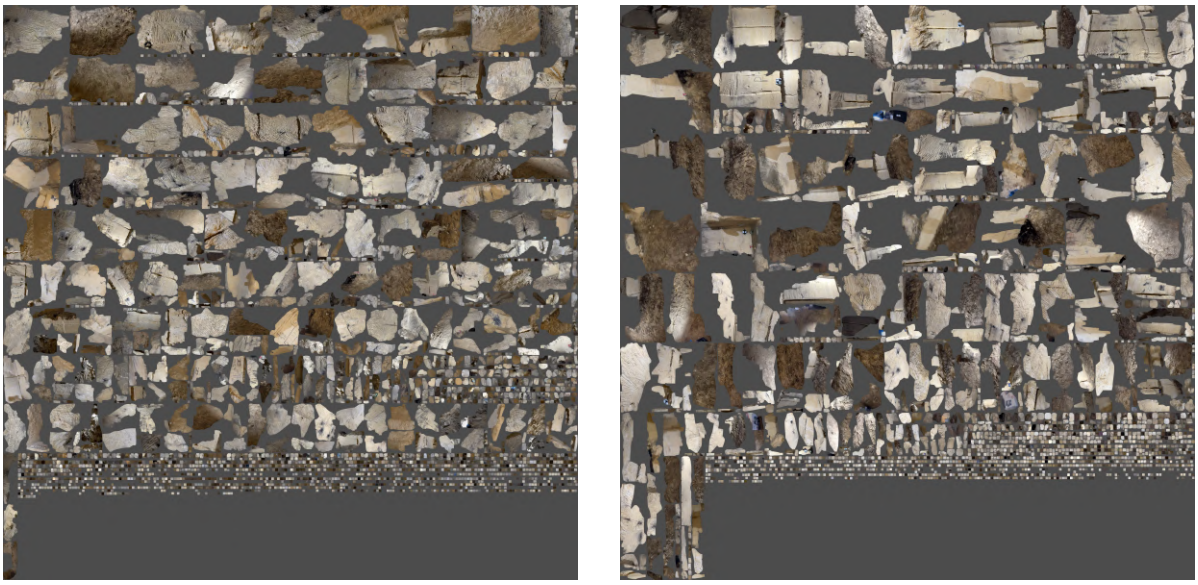
Figure A.3: Box plot of the scanned models using the *Detail* processing mode, showing statistics of the error caused by optimization. Own work.

Table A.2: Table showing the mean and standard deviation for the error caused by optimization for each location for *Detail* processing mode.

Location	Mean [mm]	Standard deviation [mm]
ZO1-1-L	0.53	0.44
ZO2-S0-L	0.58	0.46
ZO2-S1-L	0.48	0.37
ZO2-S2-L	0.48	0.36
ZO2-S3-L	0.34	0.27
ZO2-S5-L	0.34	0.26
ZO2-S6-L	0.43	0.33
ZO2-S7-L	0.50	0.39
ZW3-pillar-L	0.34	0.27
ZO2-purple-L	0.61	0.47
ZO2-extensometer-flint-L	0.30	0.24
ZO2-extensometer-L	0.47	0.40
Random ruble-L	0.90	0.69
Roof sample-L	0.87	0.74
Floor sample-L	0.45	0.37
New mine pillar-L	0.52	0.42
Central-L	0.27	0.21

Appendix B
ZO2-S0 comparisons

Appendix B shows the comparisons between the different scan ranges and the difference between the original and optimized model for ZO2-S0.



(a) An UV map obtained by the medium range scan.

(b) An UV map obtained by the long scan range.

Figure B.1: Comparison of UV maps between scan ranges for ZO2-S0 (area processing mode). Retrieved from NIANTIC, INC. (n.d.).

Table B.1: Polygon counts and the optimization ratio for the long and medium scan ranges for ZO2-S2 (area processing mode).

Scan range	Original polygon count	Polygon count after manual cleaning	Polygon count after QEM optimization	Ratio used for QEM	Total ratio (%)
Medium	235,470	224,447	152,623	0.68	64.82
Long	119,758	97,803	67,815	0.69	56.63

Figure B.2: The 3D model for ZO2-S0 with the long range scanner. Own work.

Figure B.3: 3D model for ZO2-S0 with the medium range scanner. Own work.

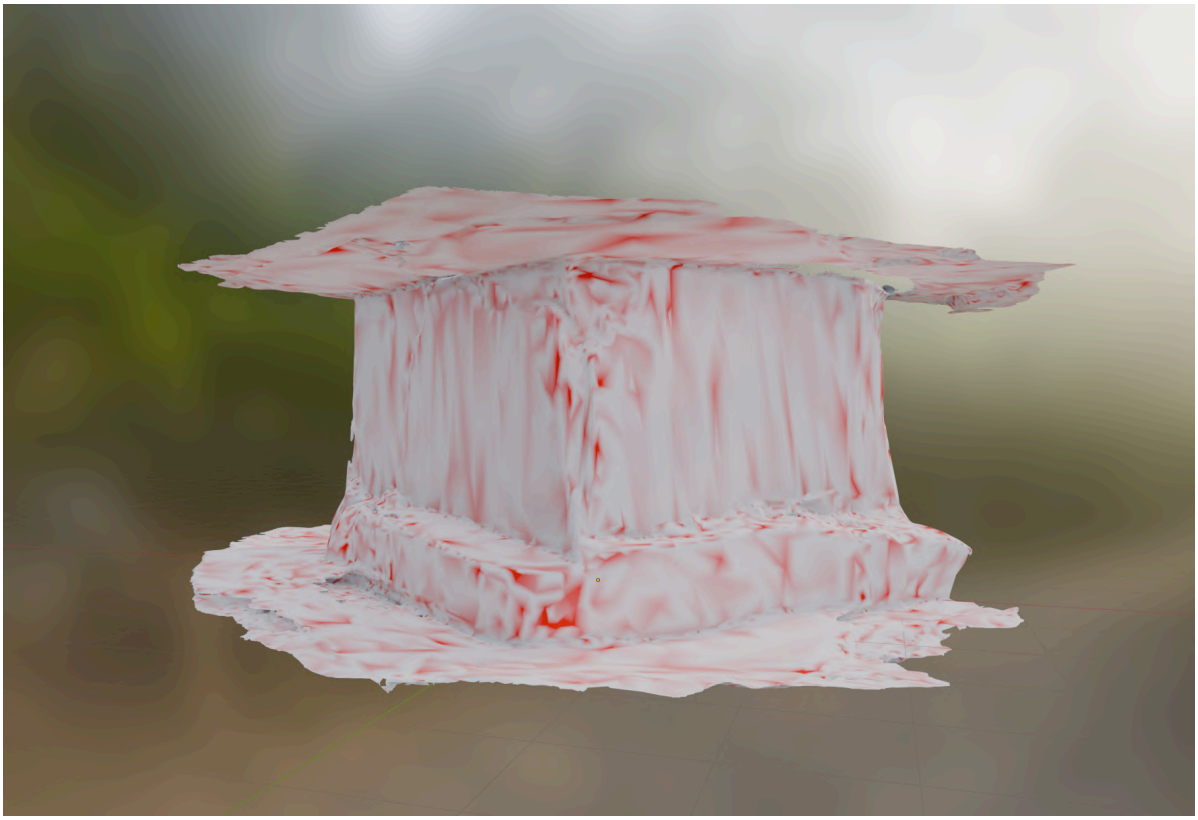


Figure B.4: Difference model between the original and optimized ZO2-S0 model Own work.

Appendix C
Schematic maps of the Gemeentegroeve

Appendix C shows the schematic maps of the Gemeentegroeve, showing where pillars with an important location have been scanned.

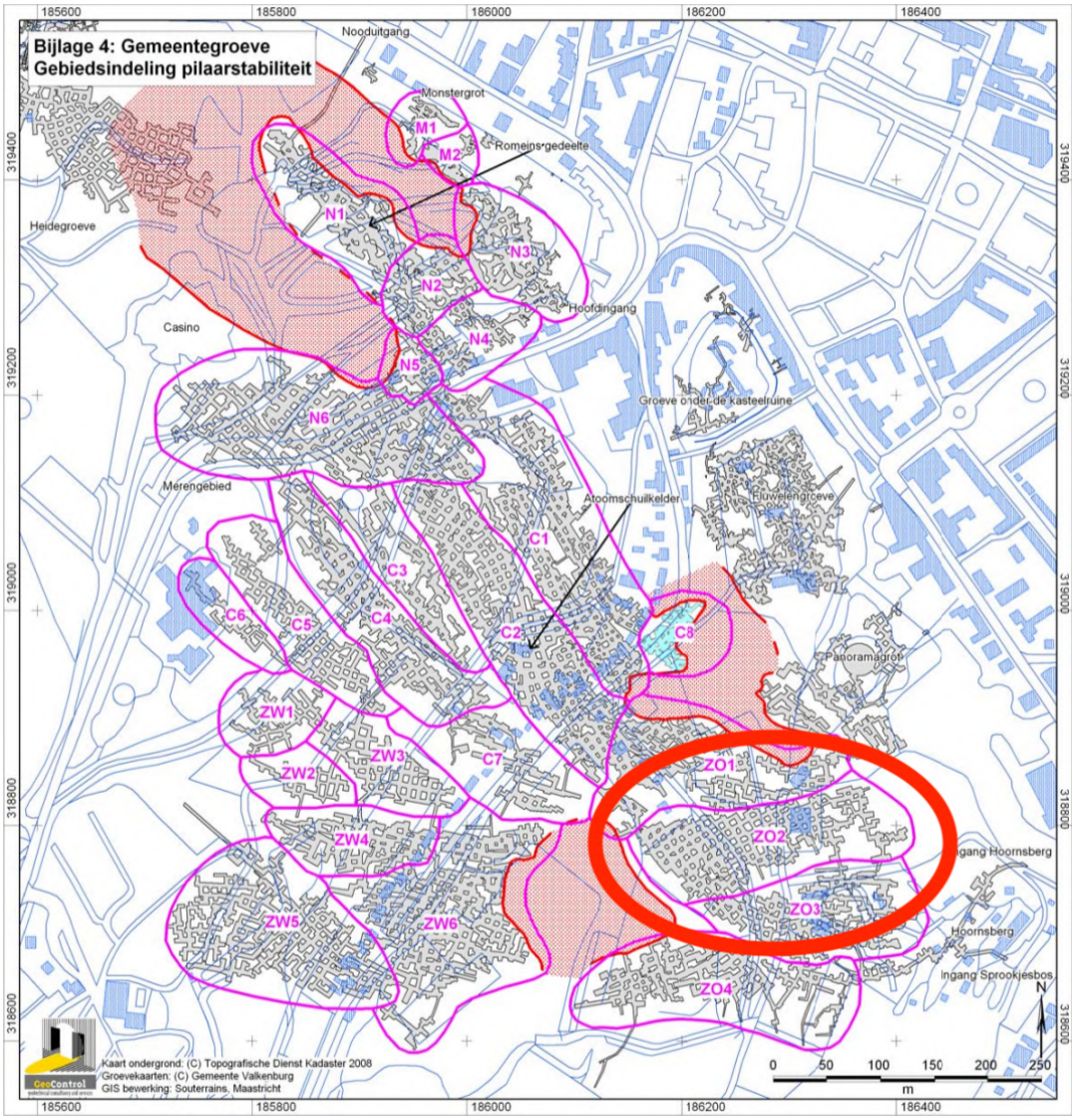


Figure C.1: Map of the Gemeentegroeve showing all the different regions and their names. The red circle shows the main area of interest. Retrieved from Bekendam (2008)

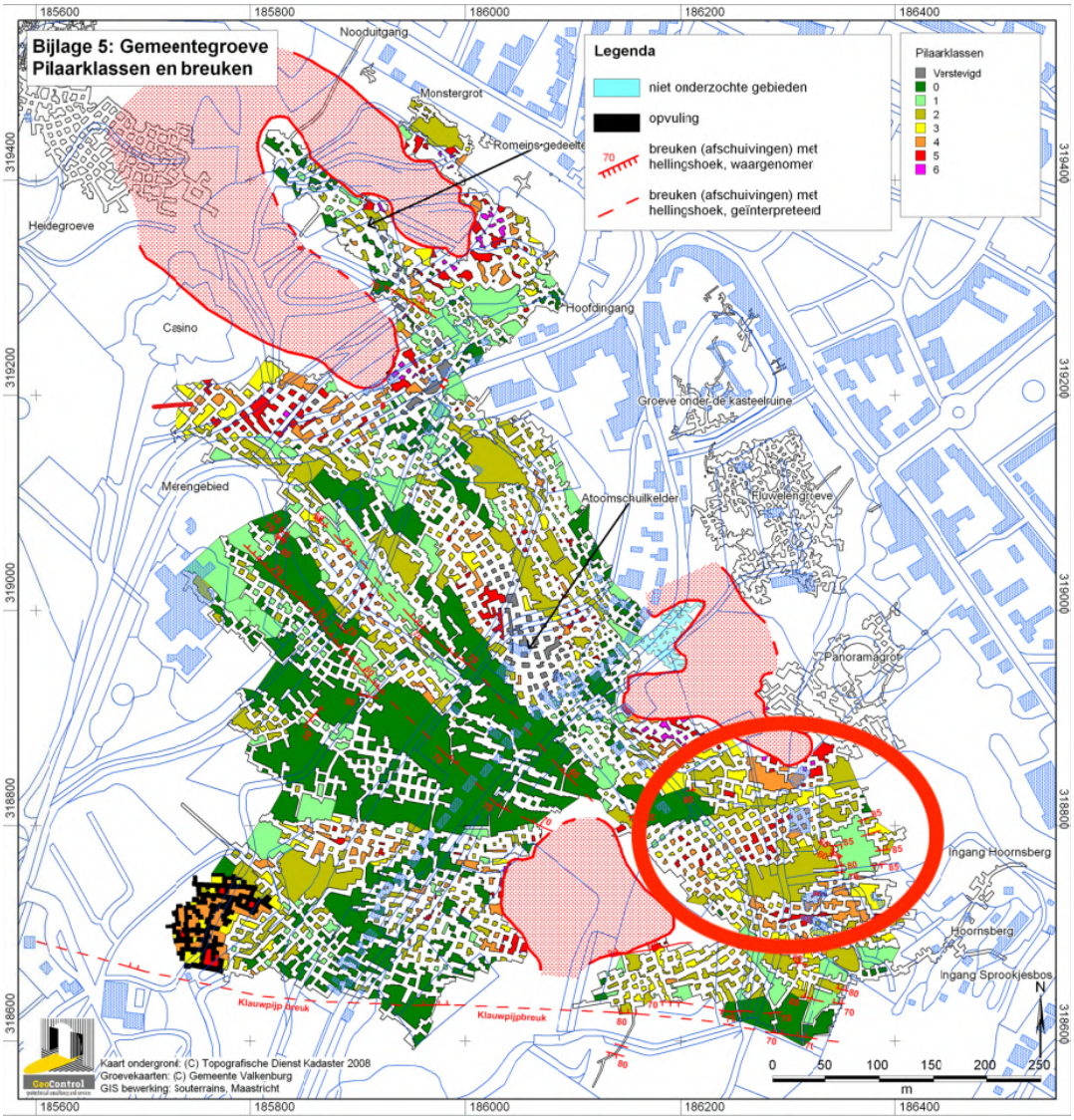


Figure C.2: Map of the Gemeentegroeve showing the classification of every pillar, based on the classification of Figure 5.3. The red circle shows the main area of interest. Retrieved from Bekendam (2008).

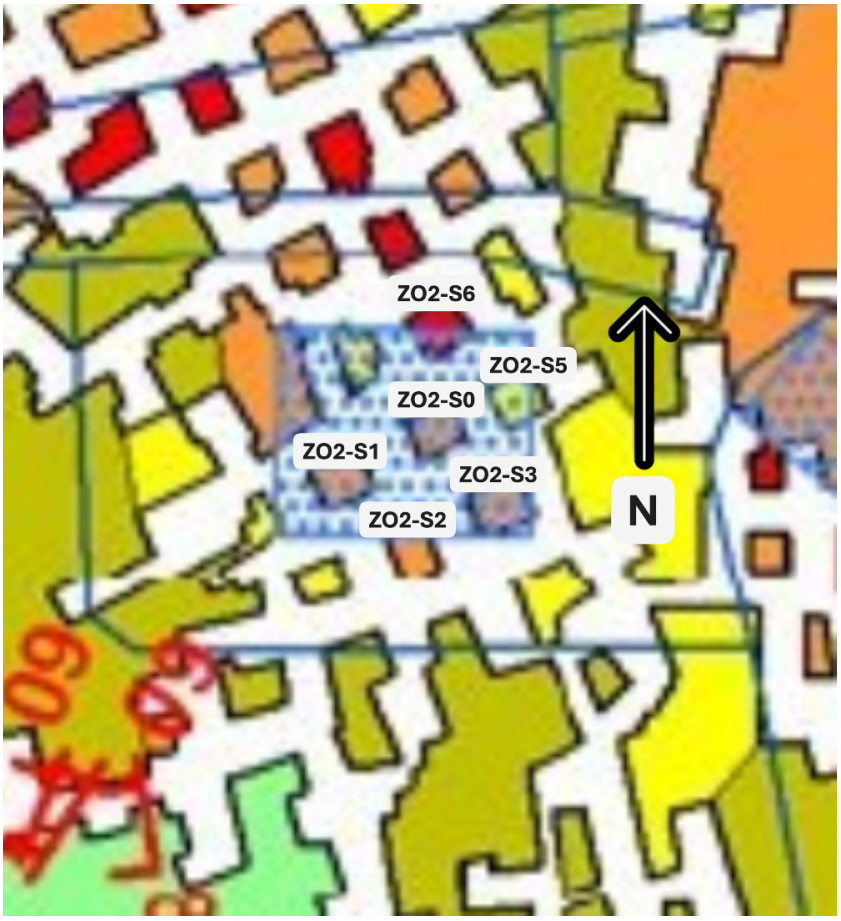


Figure C.3: A zoomed in part of the map from Figure C.2 within the ZO2 region, showing the locations of the important pillars. These are the only pillars where the location matters. Altered from Bekendam (2008).

Appendix D
Game demo experiment results

Appendix D shows the full final results of the experiment performed in Chapter 5. This can be seen Table D.1. A total of 10 players participated in the experiment, with ages ranging between 20 and 57 and ranging VR experiences.

Table D.1: The results of the experiment from the VR game demo.

Question	Mean grade	Standard deviation
General statistics		
Player age	30.50	16.44
Participants	10.00	-
Time needed for game completion.	19.88	3.98
Level 1		
How well lit does this level feel?	7.50	0.76
How useful is the headlamp here?	6.38	1.92
How much does it feel like you are walking in a mine?	8.00	0.93
How much does the minimap help?	7.13	1.13
How open does the level feel?	8.00	1.51
How easy is it to move between pillars using this movement system?	7.88	0.83
How realistic do the rounded corners of the pillars look?	9.25	0.46
Level 2		
How well lit does this level feel?	7.00	1.51
How useful is the headlamp here?	7.38	1.06
How natural is it to hold and use the tool?	8.43	0.53
How easy is it to move between pillars while holding the tool?	7.57	1.62
How open does the level feel?	8.00	0.53
Level 3		
How well lit does this level feel?	6.29	2.14
How useful is the headlamp here?	8.50	1.12
How open does the level feel?	6.86	1.35
How easy was it to find the specific permanently installed tool?	5.71	2.69
Final general questions		
How well do you feel?	8.50	0.76
How happy are you with having three levels instead of one?	8.13	0.64
How happy are you with the navigation methods?	7.88	0.83

Appendix E
Equipment used during fieldwork

During fieldwork two types of lights were brought along:

- *Professional LED bouwlamp BS 5000 MA / Krachtige werklamp 6000lm compatibel met Bosch Professional 18V accu-systeem (voor binnen en buiten IP55, 2 schakelniveaus, dimbaar, max. 5h lichtduur, powerbank-functie, BGI608 K2, zonder batterij, Engineered in Germany)* by (Interface Medien GmbH - www.interface-medien.de, n.d.).
- *WPH30R 1000 LUMENS 152M BEAM 23H* by (COAST Products, n.d.).

Appendix F
Game demo features and items

The following figures are the features and Items that can be found in the VR demo. Figure F.1 up until Figure F.4 show the features that have been implemented in the VR game demo by me. Figure F.5 and Figure F.6 show the permanently installed equipment, as seen in the VR game demo. Finally, Figure F.7 shows the Schmidt hammer during the fieldwork. Since modeling the Schmidt hammer is beyond the scope of the thesis, it is temporary replaced with an interactive pistol that shoots balls.

Figure F.1: The hub level, with the instructions on controls, and an interactive piston with cubes that can be grabbed. Own work

Figure F.2: The menu with minimap which shows where the player is located in the level. From here, the player can move to any level, reset their orientation or end the demo by pressing *Quit game*. Own work

Figure F.3: The street sign-like text appearing on the POI's that allow for easy navigation without having to toggle the minimap. Own work.

Figure F.4: The headlamp which can be toggled on and off, which can be handy in dark situations. Own work.

Figure F.5: The permanently installed extensometer at ZO2-S5. Own work.

Figure F.6: The permanently installed crack-meter at ZO2-S2. Own work.

Figure F.7: The Schmidt hammer being used at ZO2-extensometer by Maxx. Own work.