Universitat Politècnica de València

Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Hochschule Karlsruhe – Technik und Wirtschaft

University of Applied Sciences

Fakultät für Informationsmanagement und Medien

Master's Programme Geomatics

Master's Thesis

of

Raquel Arcón Navarro

Capturing improved TLS data of Maulbronn Monastery and integration of the mesh into the existing UNITY visualization

First Referee: Prof. Dr. -Ing. H. Saler

Second Referee: Dipl.- Ing. (FH) K. Berner

Third Referee: Prof. Dr. -Ing. J.L. Lerma-García

Karlsruhe, December 2019

Terms of reference

Master Thesis

of

Raquel Arcón

Capturing improved TLS data of Maulbronn Monastery and integration of the mesh into the existing UNITY visualization

1. Introduction

The World Heritage site Maulbronn Monastery is to be made visually accessible to the public with the help of Virtual Reality via internet visualization. For the monastery, an accurate 3D model based on TLS data will be generated. The TLS point cloud will then be meshed and mapped with scanner photos. The mesh is then in UNITY integrated in online visualization.

2. Tasks

The aim of this thesis is the implementation of improved TLS data in the existing visualization of the Maulbronn Monastery in the 3D game engine Unity.

- Additional TLS data of Maulbronn Monastery have to be captured for obtaining an improved TLS data set of Maulbronn Monastery.
- When taking pictures by the scanner, special attention have to be paid on optimal exposure by using HDR mode.
- All undesired marks in the photos have to be masked out and overlapped by better images
- All noise in the point clouds should be removed.
- After registration and geo-referencing of the point cloud follows the generation of the mesh, which will be mapped by the images in Agisoft Metashape.
- The mesh should be implemented in the existing 3D visualization in UNITY.
- A tiled model should be generated and exported from Agisoft Metashape. The performance improvement possibilities resulting in the use of tiled models in Unity should be analyzed and tested.

Duration:	6 months	1st Referee:	Prof. Dr.-Ing. Heinz Saler
Handing out:	05.06.2019	2nd Referee:	Dipl.-Ing. (FH) Konrad Berner
Closing date:	05.12.2019	3rd Referee:	Prof. Dr.-Ing. José Luis Lerma-García

Approval and Declaration

This Master Thesis called "Capturing improved TLS data of Maulbronn Monastery and integration of the mesh into the existing UNITY visualization" has been submitted by Raquel Arcón Navarro and has been accepted as partial fulfilment of the requirement for the Master Thesis of Geomatics in Karlsruhe University of Applied Science in December 2019.

I hereby declare that I have written the present work independently, based and produced by my own research. The works, publications or lectures contributed by other authors were clearly defined with references in the thesis. The work has not yet been submitted in whole or in part to another examination authority and has not yet been published.

Abstract

This Master Thesis consists in improving the existing 3D visualization of the Maulbronn monastery, because there are areas with excess brightness that is produced by the windows. To achieve this purpose, the old scans that were part of an existing FARO SCENE project have been analysed. After analysing the scans, those areas that had to be repeated to improve texture were detected. Additionally, tests have been done to find out which parameters are best suited to improve the quality of the HDR images. Afterwards, different scans have been taken with the best parameters. This data has been processed and recorded with the data from the previous scans, resulting in the creation of a mesh for each zone, along with the position file and HDR images. Geomagic Qualify has also been used to improve mesh geometry. Then the images have been edited in Photoshop to represent a better texture for the mesh, as well as masks have been created not to apply those areas of the images that do not have good quality. In order to reproject the images on the mesh, the Agisoft Metashape program has been used, resulting in a tiled model. Once the tiled model is obtained, only the last level has been used to incorporate the new meshes into UNITY. Finally, the texture and some parts related to walkability have been improved through the use of several scripts.

This project is divided into three parts. The first is the theoretical part, where the basic concepts of 3D visualization and data processing are explained. The different types of software that have been used are also explained. The second part is the explanation of the practical part, in what it consists and in what steps it is divided. Finally, in the last part of the document are the results, conclusions, future lines of the project and references.

Table of contents

List of Figures

List of Equations

1. Introduction

The Maulbronn Monastery is considered one of the best-preserved medieval Cistercian monasteries in Europe and is part of the UNESCO World Heritage Site from 1993. It dates to the 12th centuries and among its buildings are several types and degrees of architectural development represented, from Romanesque to late Gothic.

The monastery is situated in Maulbronn (Baden-Württemberg, Germany) (*Figure 1*).

Figure 1. Location of the monastery (Monasterio de Maulbronn - Wikipedia)

This project aims to improve the 3D visualization that already exists of the monastery. Therefore, the two main works are

- On the one hand, improve the existing 3D visualization of the Maulbronn Monastery by using panoramic images to give texture to the model. For this step an existing project will be used where all parts of the monastery are located except the church.

- On the other hand, deal with the incorporation of the church to the total 3D visualization of the monastery, creating the mesh and projecting the images to obtain a good texture quality.

Once these two purposes have been accomplished, the result will be an improvement in the representation of the monastery in Unity.

Unity is a multiplatform game engine, and currently the game engines are taking a stronger presence in terms of 3D representation of objects and structures is concerned, the objective being to provide the user with a more realistic immersion that allows user to explore virtually the spaces generated by different techniques.

2. Software that has been used

Taking advantage of the techniques and knowledge acquired, it was carried out the two purposes previously explained. To do this, five different software were used.

1. The FARO SCENE software was used (since the laser scanner was from FARO Company) for the filtering, visualization of the data, and creation of the mesh.
2. Geomagic Qualify was useful to clean and improve the meshes.
3. Photoshop was useful to create mask for the images.
4. The software Agisoft Metashape was used to reproject the spherical images into the mesh and create the tiled model of each part of the monastery.
5. UNITY software was used with the aim of creating a 3D representation in which the user can move around the 3D visualization of the monastery.

In the following chapters, the process within each of the different programs is explained with more detail.

3. Theory about the laser scanner

The laser scanner is a tool commonly used in surveying for the acquisition of data in a massive way, which generates a three-dimensional point cloud based on the measurement of angles and distances through the use of a laser beam.

The laser scanner is divided into two basic components for its operation. The first one is about the laser itself as useful to measure distances. The second component is a sweeping mechanism, a motorized mirror that diverts the laser in different possible directions (vertical and horizontal). With this mechanism the laser is able to measure the distance of a large number of points. In addition, the laser scanner also obtains information about the angles, and the reflectance of the different surfaces in which the laser scanner impacts.

At the same time, there are two types of laser scanners used in heritage documentation depending on how they measure distances; the first is the scanner based on the measurement of flight time and the second the scanner based on the measurement of a triangulation.

In the first group are the time measurement scanners. They are based on the calculation of the distance from the measurement of the time that elapses between the emission and the reception of the pulse. Within this group are two other types of scanners, those based on pulses (or flight time) and those based on phase measurements.

The second group, a triangulation-based scanner, is based on a high precision method (microns) and short range (less than approximately 10 m). This method can be useful for scanning small objects that require a large number of detail and precision.

In this project the scanner that is used (FARO S70) belongs to the group of phase measurement instruments, since this type of tools are characterized by the speed of measurement, with a distance range of less than 100 meters but with great precision, reaching to obtain millimetre precision.

To sum up, the laser phase measurement scanners emit a sine wave and compare the wave that has been emitted with the wave that has been returned after impacting the object and obtain the phase difference between both waves. Therefore, by knowing the length of the wave, the medium that crosses, and knowing that a complete wave corresponds to a phase difference of 2 per pi radians, it could be possible to obtain the distance with the following equation (*Equation* 1):

$$d = \lambda * \frac{\Delta\phi}{2\pi}$$

Equation 1. Equation to calculate the distance. *(García-Gómez et al, 2012)*

The letter "d" represents the distance to be calculated, taking into account the going and the return of the wave. λ represents the wavelength and Φ the phase difference between the two waves (round trip).

The accuracy of the result depends on the accuracy obtained in determining the phase difference, but to obtain good results the reflectivity must also be considered.

When it is being measured, the object absorbs part of the energy of the laser light beam (the amount of energy that is absorbed depends on the material that makes up the object); The remaining energy part is reflected by the object and collected by the laser scanner, which signal is used to calculate the distance. The more signal intensity returns in the wave to the laser scanner, the more accurate the measurement will be; Therefore, to speak of reflectivity is to speak of accuracy, since these concepts are highly related.

As it has been mentioned previously, the amount of energy that is reflected depends on the material of the object but also depends on the geometric conditions of the measurement and that in long distances and at certain angles the signal noise can increase. Thus, this can influence the intensity of the energy that is reflected back to the scanner.

Therefore, there are several errors due to the instrument or its use:

- Errors due to the diameter of the laser beam. This case occurs when the diameter of the laser is considered small and if it falls on a vertex of an object or an edge it may cause two different distances to be returned. In this case, when working with phase comparison, it may be the case that there are several returns for the same point.

- Errors due to reflection of the signal. These in turn depend on the atmospheric conditions and the angles of incidence in the measurement.

4. Basic concepts on 3D Visualization and data processing.

This chapter is about the importance of the laser scanner for the visualization of buildings, as well as what are the most important aspects and their procedure.

As it has been previously shown to talk about resolution is to talk about detail, and this is a key element when scanning a building because depending on the level of detail to achieve many elements vary within a project. With more detail, the scan times are multiplied, the size of the files also and with it the volume of data to be handled. For this reason, it must be clear about the purpose of the project and know what is the smallest detail that it is wanted to represent, considering at the same time the maximum precision of the laser scanner.

The resolution is a parameter that can be configured in the laser scanner but that also depends on the situation of the instrument in front of the element that it is wanted to scan. In some cases, in the cloud of points the separation between points is not uniform throughout the surface that has been scanned. This may depend on the orientation and distance of the surface to the laser scanner. Therefore, it can be said that the geometry and the location of the instrument is a key factor when making a scan of a building.

Another part that must be considered when scanning buildings is the coordinate system. Technically, each scan has a relative coordinate system since each time that the position of the laser scan is changed to make a new scan, a new coordinate system is defined. Then, to be able to fit all the scans it is necessary that between them there are areas of overlap that allow a few clouds of points to fit with others.

Once the scanning of a building has been completed, a lot of point clouds with information about the coordinates of each point are obtained, as well as information about the reflectivity. Subsequently, it is the software that is responsible for reading and translating those values and generating a visual representation of the building.

To process the data, the point cloud is first imported, then the data is filtered to eliminate the noise or elements that are not interesting and finally the scans are registered. In most of cases the software used is from the same company as the laser scanner.

As can be seen in the following image (*Figure 2):*

- The first step is to detect the reference points and the overlaps between different scans. Scanner Coordinates system.
- The second step is to move and rotate the scans with the common reference points. Relative Coordinates.
- The third step is to assign coordinates to the reference points. Absolute coordinates.

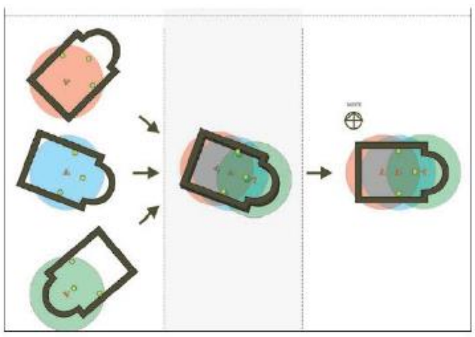


Figure 2. Process of georeferencing and joining of scans. (*García-Gómez et al., 2012*)

In order to create a good visualization, it must be supported by a good cover of the point of cloud. For this it is necessary to apply several filters and to depurate the data to work with those that really interest, that means, to eliminate the noise and the erroneous points.

As Main concepts when processing data for future 3D visualization, the most fundamental parts are the mesh and the images that will give the texture.

4.1 The meshes and their components

A mesh is defined as the triangulation of the points that the laser scanner collects, creating 3D shapes that represent an object or surface.

The meshes are formed by three components:

- Vertices: are the points that form the different triangles of the mesh.
- Edges: are the lines that join the vertices forming triangles with each other. These lines define how the vertices are connected and the direction of these lines gives orientation to the faces.
- Faces: it is the triangular 2D surface that is created by joining the vertices through the edges.

In the following image it is possible to see an example of the three components of a mesh (*Figure 3*):

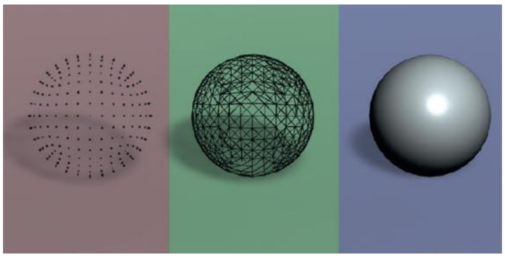


Figure 3. Example of mesh components. (Halladay, 2019)

The meshes have a very simple way of being stored so that it is easier to interpret in the different programs. Of the three main components, only the vertices are stored in memory, since the edges and faces are implicitly defined when defining the vertices in a specific order.

The order in which the vertices are defined is very important because depending on that factor, the front and back sides of each 2D face are identified. For example, if three vertices are defined clockwise, that will be the front face. Whereas, if defined counter clockwise that face will be the back part.

This concept is something that must be considered because many programs use a Backface culling optimization. This optimization is responsible for not rendering the backs of the faces, and only leaves the front visible. In the chapter 7.5.1 it could be found more information about this optimization.

4.2 Images and UV coordinates

The purpose of this project is to reproject the spherical HDR images in the mesh, to give it texture. This process is called “texture mapping” and is based on using a texture generally in 2D (in this case the images) and applying it to a 3D surface (in this case the mesh).

This process is carried out using UV coordinates. Each vertex of the mesh is assigned a UV coordinate, and in turn, the images also contain UV coordinates. The UV coordinates are interpolated to obtain the values of the intermediate pixels on the 3D surface. These coordinates begin in the lower left corner and increase to the upper right corner.

In this project, as it previously mentioned, it is pretended to create a visualization of the monastery in 3D with which the user can interact that is, the user can "get inside" the 3D visualization and "navigate" inside it. For this purpose, a software called UNITY will be used, which has been used mainly as a tool to create videogames, but which has also recently been applied to broader approaches, such as the representation of interactive buildings in 3D.

In the next chapter it will explain what UNITY, its functionalities and how can contribute to the 3D visualization of buildings.

5. Unity software: Advantages and disadvantages

Virtual reality (VR) tools have revolutionized the conception and way in which engineers and architects focus their projects, since the rendering engines of the new VR software have a simple interface and add additional value to project visualization.

Unity is software that was developed by Unity Technologies in 2005 and is based on C ++ and C# programming.

As previously mentioned, it is mainly focus on the creation of video games, but it also has many more branches. As indicated on its own page, Unity can be useful for the transportation, automotive and production, animation, engineering, architecture and construction sectors.

There is currently a wide variety of software on the market that can be useful for 3D rendering, such as Unity, Unreal Engine 4 and Vray RT. But it has been chosen to work with Unity, since it is a fairly complex software and easy to handle at the beginner level. In addition, it has many features that can be useful in the project.

Speaking more specifically about Unity, it should be noted that this program has a fairly simple graphical interface and that it has a wide variety of menus that are easy to use and modify without having to write code, although if it is wanted to create certain behaviours, one must program or edit parts of code that are not too complicated. In addition, this software has a large number of add-ons that can be found for free.

Being a known program with a large number of users, it is very easy to find information, documentation, videos, courses and tutorials on its use.

<u>Advantages and disadvantages of Unity</u>

This section analyses the use that can be given to this tool for 3D visualization of the Maulbronn Monastery. For this, a series of advantages and disadvantages of such software are shown below:

- Simple use: As mentioned earlier, the use of this application in the creation of basic stages is not very complicated. Knowing approximately 20% of the total capacity of the tool can generate medium level games / visualizations.
- Programming languages: As previously read, most of the programming with Unity is carried out in C#, although depending on the applications it is wanted to grant the visualization, it can also be programmed in C ++. In addition, it is possible also to program some things with JavaScript, but it is not usual.
- Assets Store: Unity has a store of accessories for display, such as scenes, sounds or even control modules of any object already designed. These assets allow the user to work faster and more easily. The only disadvantage of using assets is that they increase the storage space of projects.
- Multiplatform: Unity can be used on Windows, Mac or iOS and Android, as it generates codes for all types of systems

One of the main disadvantages of Unity is the poor administration of memory and ".Net" libraries because they can be somewhat slow in some cases.

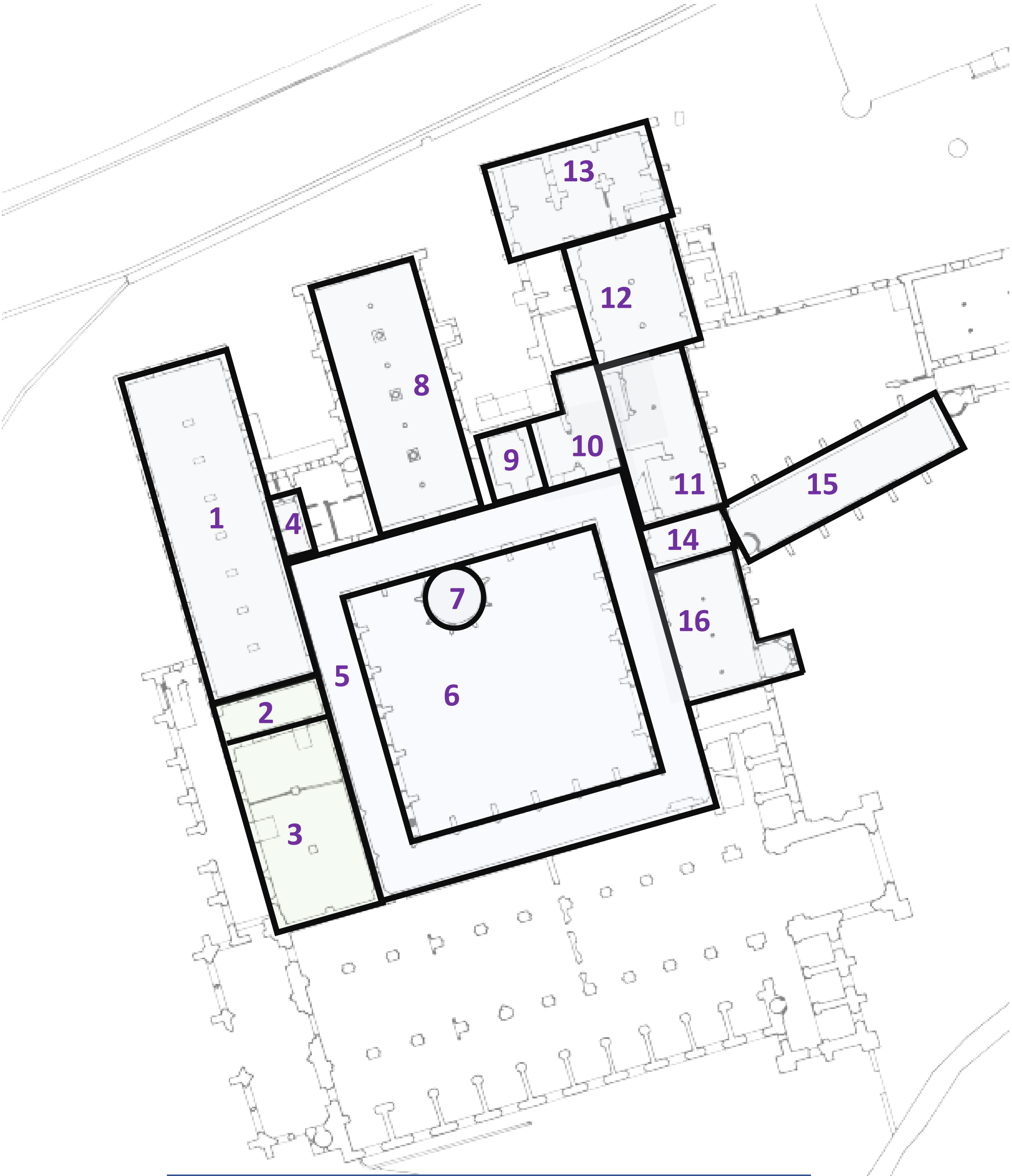
6. What has been done so far?

Until now it has been scanned and a 3D visualization of almost the entire monastery is available, the last missing part is the monastery church. Originally there are two SCENE projects in which are the Maulbronn scans. In the first project (General Project) are all rooms except the church (Church Project). In the second project it is about the church.

Additionally, there is also a project in UNITY, in which all the rooms of the monastery can be visualized, excluding the church.

6.1 General project in SCENE

In total there are 16 "rooms" (Regardless of the church). In the following image it can be seen a sketch. (*Figure 4*). The names that appear in the legend refer to the original names of the SCENE project.



1- Laienrefrektorium
2- Ern
3- Cellarium
4- Ausgang
5- Kreuzgang
6- Innenhof
7- Brunnenhaus
8- Herrenprefektorium
9- Kalefaktorium
10- Gangraum
11- Frateria
12- Grosser Keller
13- Annexräume
14- Ostern
15- Marienkapelle
16- Kapitelsaal

Figure 4. Parts of the general Project.

This project consists of a total of 201 scans distributed across the 16 rooms (*Figure 5*):

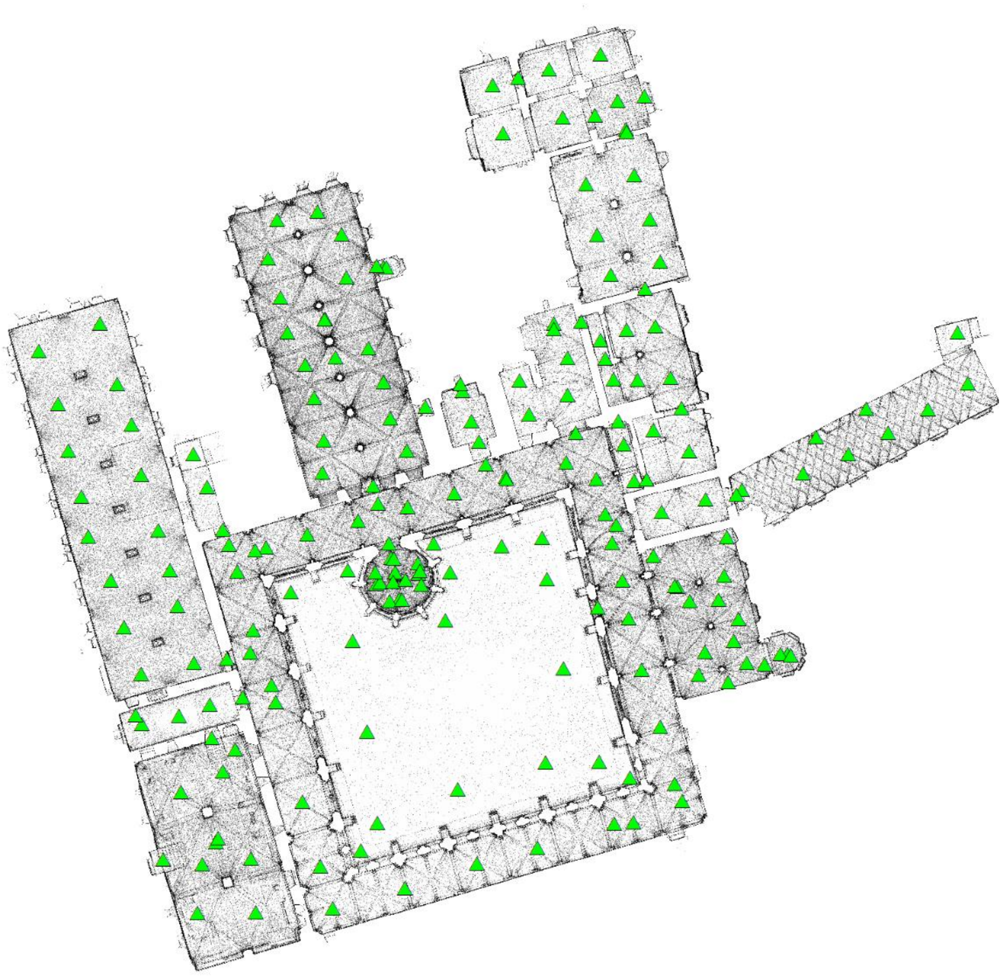


Figure 5. Scan positions.

When creating the texture in this project the images were not used, and colour was given through the intensity levels captured by the laser scanner. This makes some areas look blurry, not well defined and too bright.

For example, there are several areas in which the problem to reflect a good texture on the surfaces is the luminosity. This problem arises when it is wanted to scan a room with a fairly dark luminosity, which has windows that look outward with much more light. There are also areas where no outside light came out and artificial light had to be used. Due to the excess or absence of brightness, the texture in those areas is not good.

For all this, techniques and solutions that improve the current situation of the representation of the monastery should be sought. Therefore, some scans have been repeated with the aim of trying to improve and solve the problems of representation quality.

The purpose of this project is not only to improve the areas that have already been scanned, but also to include the part of the church in the visualization.

6.2 Church project in SCENE

As the name indicates, exist a project in unity with the church of the monastery. The Scans were taken this year as a part of the Bachelor Thesis from Simon Baumer (Simon Baumer, 2019). The scans of the project are already processed, registered and the project point cloud is created.

The following figure shows a sketch of the monastery, which indicates in red the missing area to include in the visualization of the project UNITY (*Figure 6*).

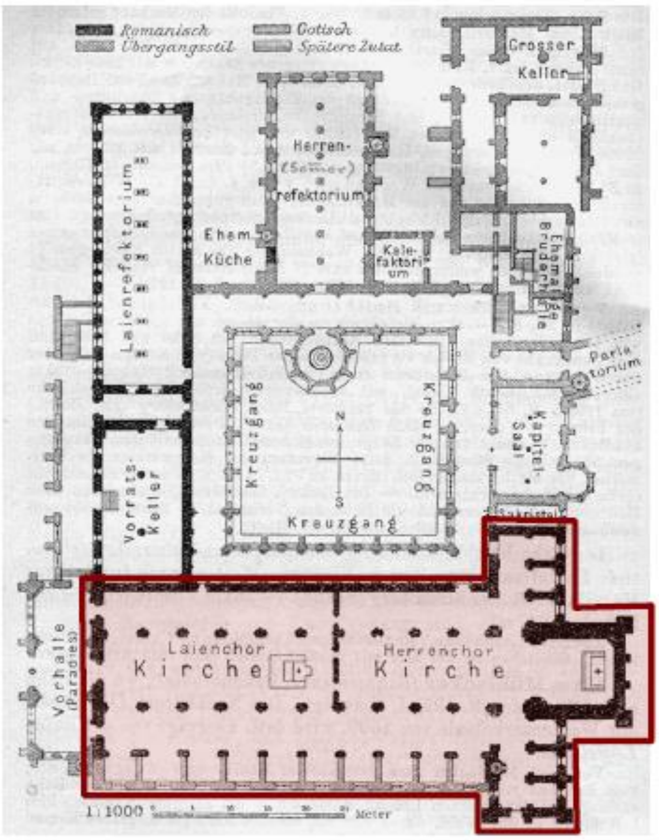


Figure 6. Location of the church. (Monasterio de Maulbronn - Wikipedia)

Specifically, the church scan project consists of 149 scans, distributed in such a way that a good 3D representation is guaranteed. Below, it is possible to see schematically how the scans were organized (*Figure 7*):

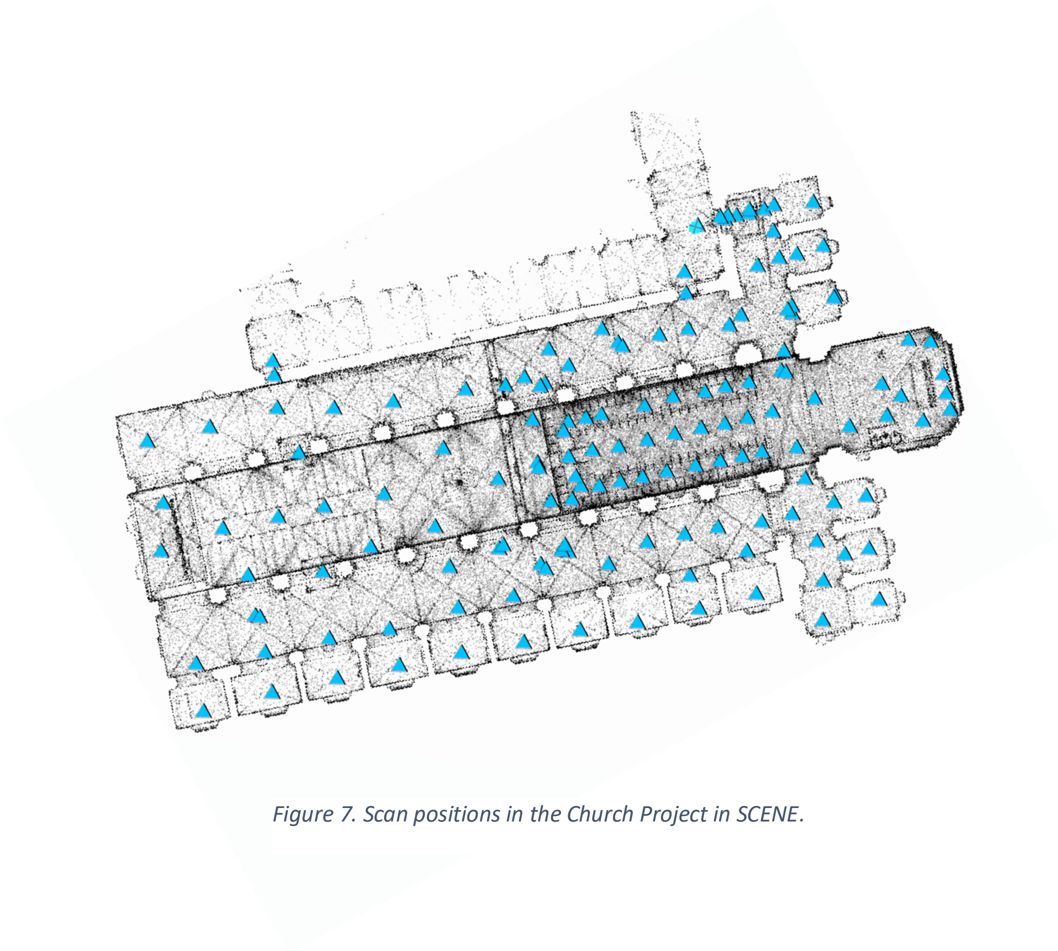


Figure 7. Scan positions in the Church Project in SCENE.

6.3 Unity Project

This section will talk about how the project is composed in UNITY and will show several examples of how the model looked like before.

6.3.1 Project Organization

The UNITY project has 4 different scenes. This structure has been maintained since it has only been necessary to modify scene number 4, since it is in that scene two the visualization of the rooms of the monastery is carried out and that is where the meshes have been replaced.

<u>SCENE 1</u>

Scene 1 is basically an information panel near the monastery. Additionally, it also contains a menu to navigate between the different scenes that the project has. Scene 1 is the first scene that is displayed as soon as the project is executed. Next, an image of how the structure of the project in scene 1 (*Figure 8*) and another image of how it looks when executing the program (*Figure 9*) will be displayed.

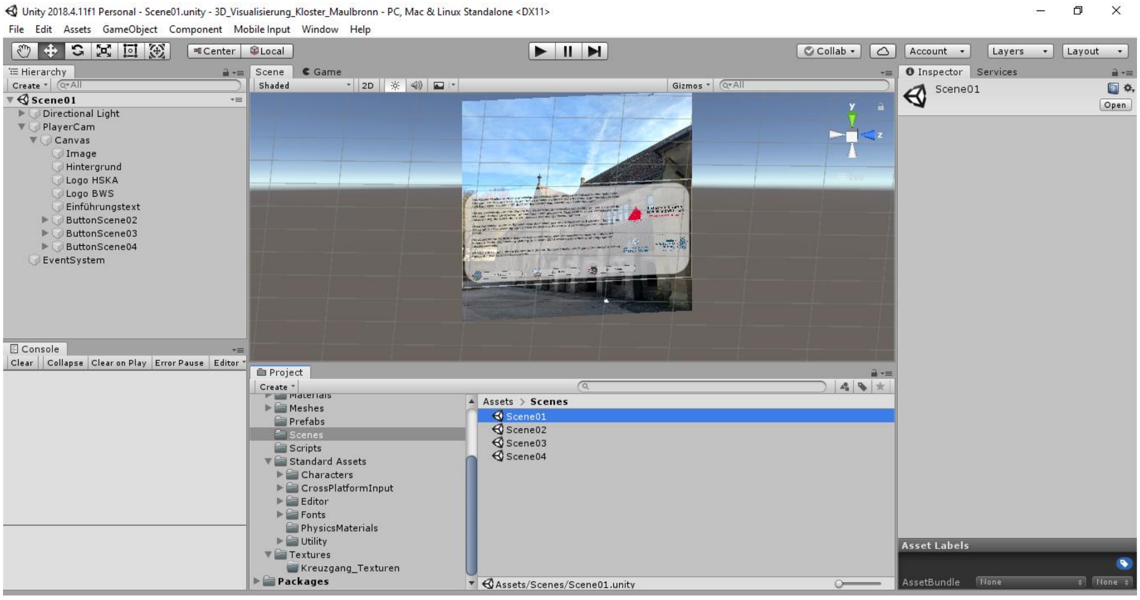


Figure 8. Visualization of Scene 1 in UNITY.

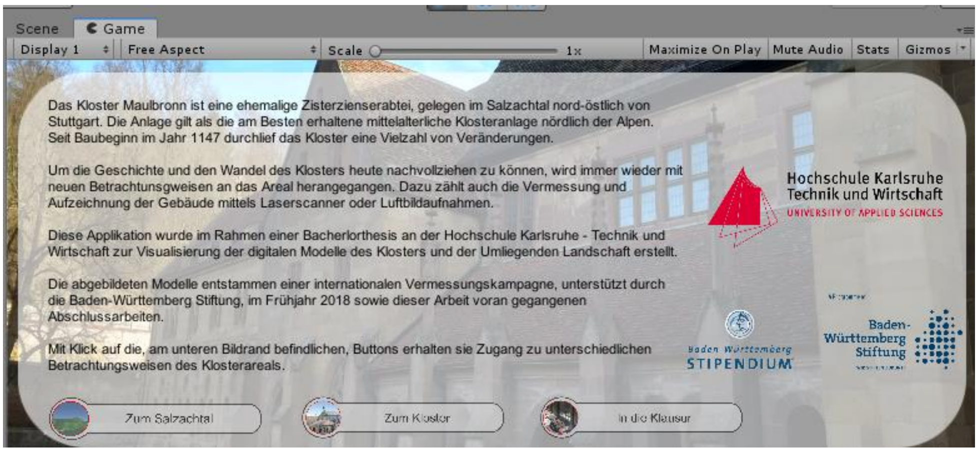


Figure 9. Visualization of Scene 1 in UNITY.

<u>SCENE 2</u>

Scene 2 could be defined as an overview of the valley where the monastery is located. Not only can the monastery be visualized, but also the rest of the buildings that are around it. To give more reality to the 3D rendering, a DTM has been used. (*Figures 10 and 11*)

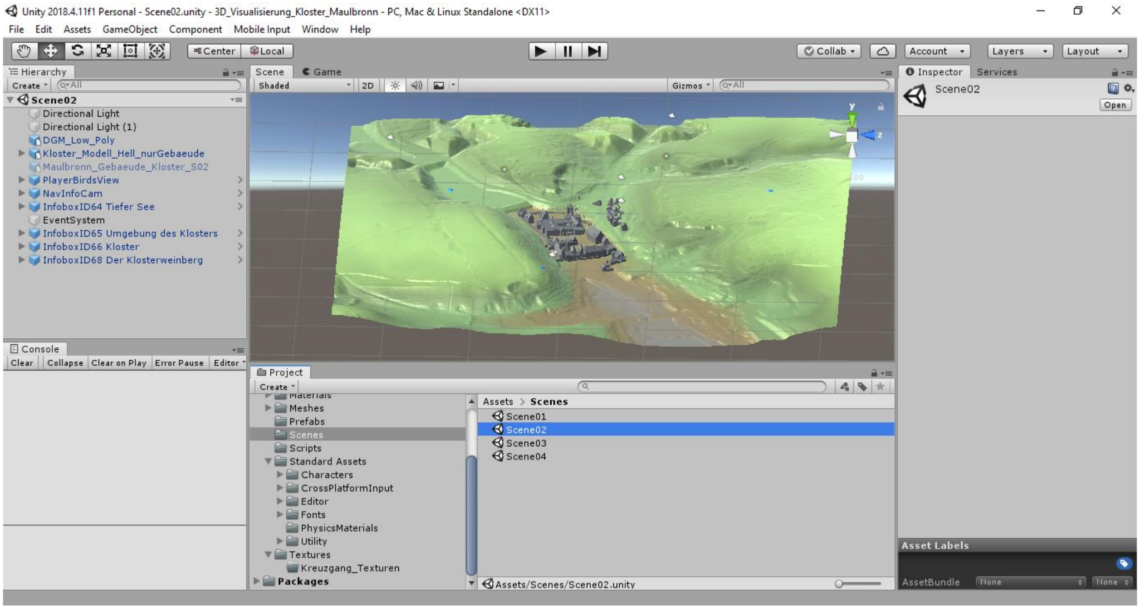


Figure 10. Visualization of Scene 2 in UNITY.

Figure 11. Visualization of Scene 2 in UNITY.

<u>Scene 3</u>

Scene 3 is very similar to the previous scene, but it presents more detail. In this scene it is possible to see perfectly how the model of the monastery is on the outside. Different meshes were created to represent the 3D models of the houses surrounding the monastery. (*Figures 12 and 13*)

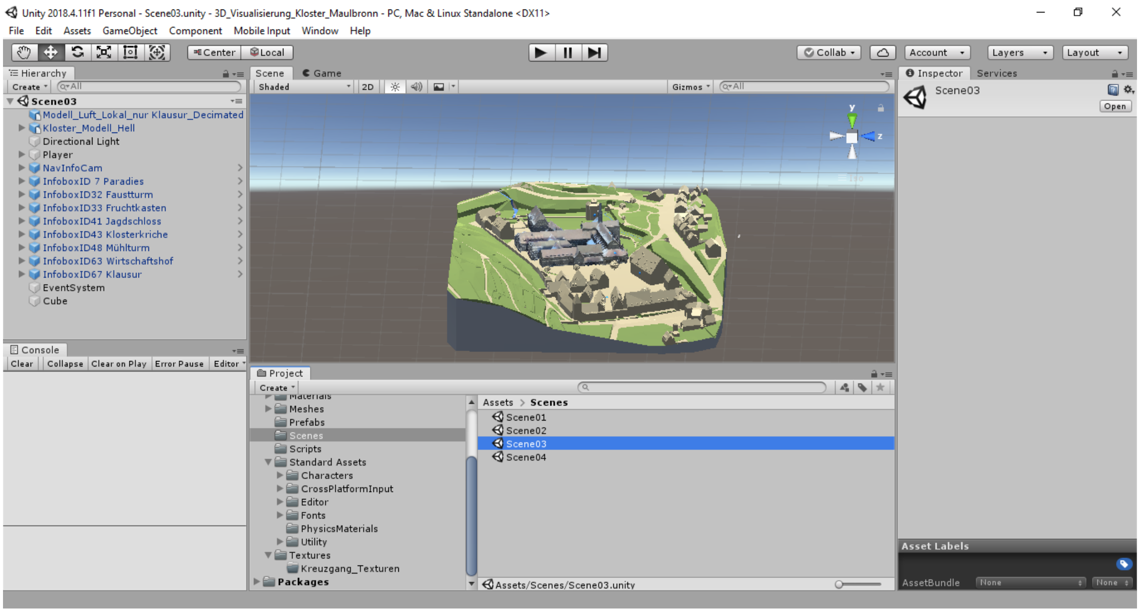


Figure 12. Visualization of Scene 3 in UNITY

Figure 13. Visualization of Scene 3 in UNITY.

<u>SCENE 4</u>

As previously mentioned, scene 4 is the one that has been modified in this master thesis. This scene consists of all the rooms of the monastery except the church. The model of the roof of the monastery has been carried out thanks to the incorporation of a UAV. Additionally, some colliders were incorporated to the scene 4. (*Figures 14 and 51*)

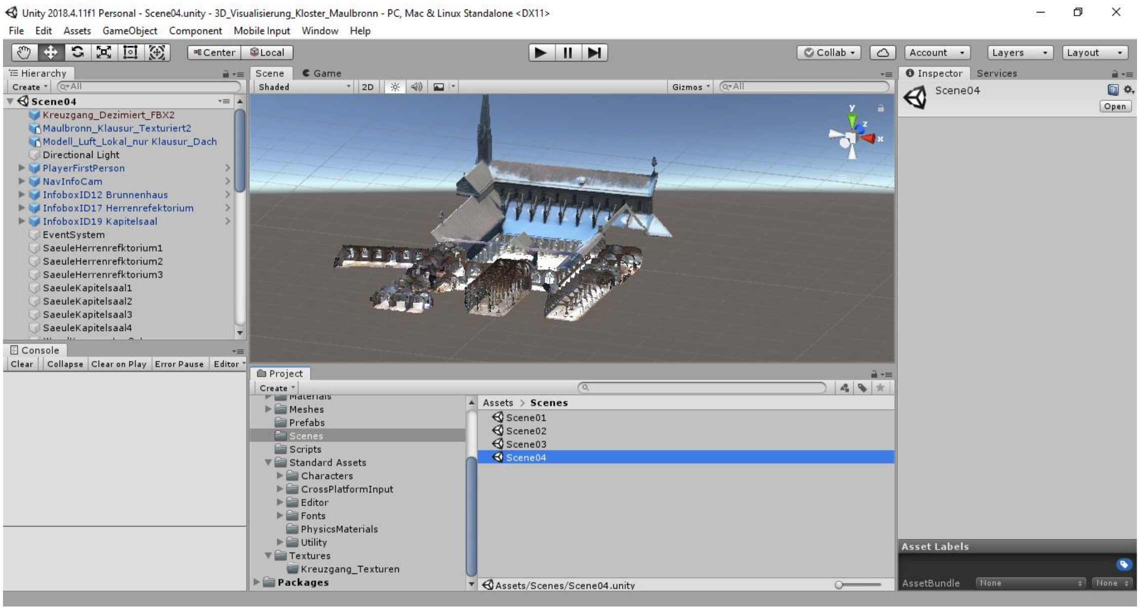


Figure 14. Visualization of Scene 4 in UNITY.

Figure 15. Visualization of Scene 4 in UNITY.

6.3.2 Project Visualization

This section will show two images corresponding to the original 3D visualization in UNITY. In the first image it can be seen part of the central hall and the room called “Kapitelsaal”. As it can be seen the floor is completely white due to excessive exposure of light in the image, the windows of the windows are not appreciated, and the texture of the walls is very muted. (*Figure 16*)

Figure 16. Visualization of Kapitelsaal in UNITY.

The second image shows the inner courtyard of the monastery called "Innenhof", in which it is also possible to see that the whole floor is completely white. In that case the excess brightness was also affected by snow, since when some measurements were taken the grass was covered with snow. At the same time, it can also be seen that the texture of the walls does not look very clear. (*Figure 17*)

Figure 17. Visualization of Innenhof in UNITY.

It is also worth mentioning that the project is programmed in such a way that the user can walk inside some rooms. Currently although all rooms are incorporated into the project, not all can be visited walking inside. The only rooms that can be visited are Kapitelsaal, Herreprefektorium, Innenhof and Brunnenhaus. This is because there are colliders that do not allow access to these rooms (Chapter 10.2 explains that they are the colliders in more detail). Additionally, in Kapitasaal, Herrenprefektorium and Brunnenhaus there is a way to visit these rooms in a different way, thanks to a blue “info ball”. The info ball as its name indicates is a sphere that is placed in each of these rooms and that allows when the user clicks on it, can see how the room is inside without having to enter inside. Additionally, the user can also read some information about the room on the left side of the screen. Below it can be seen an example of what the info ball program looks like (*Figure 18*) and what the user's visualization would look like once the user clicks inside the info ball program (*Figure 19*).

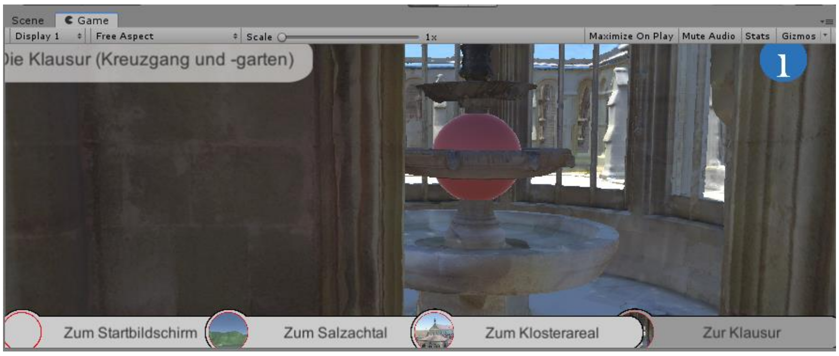


Figure 18. Example Info ball location

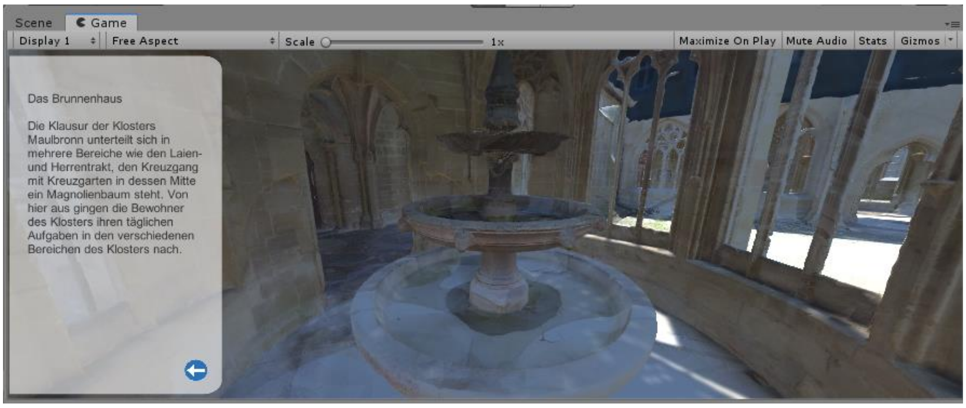


Figure 19. Info balls game view.

7. Project Workflow

In these sections, the workflow that has been followed in each program will be explained step by step. It will be summarized so that it is easier to interpret the procedures and later we will talk more specifically about how the workflow has been carried out in each room.

In chapters 8.6 and 9.1 it is possible to read more information about the steps and the results obtained with each software in each room, this section is only to have a general idea of how it works and to have an overview of the Workflow of the project.

7.1 Using FARO SCENE

FARO SCENE has been used to process the new scans and register them with the rest of the scans of the Project already created. Once everything was registered, the meshes of each room were created. Finally, the spherical HDR images of each scan, the positions, and the meshes have been extracted. These three elements are necessary to be able to use the images to texture in Agisoft Metashape later.

7.2 Using Geomagic Qualify

In Geomagic, the meshes have been imported and edited. In some cases, unnecessary pieces have been trimmed, possible gaps filled and the three-dimensional structure of the meshes softened. In turn, in some meshes a tool has also been applied to simplify small triangles in a larger one and thus be able to decrease the number of triangles.

7.3 Using Photoshop

In Photoshop the masks of the images have been created, and they have changed the format from ".png" to ".jpg" since it occupies much less space.

The masks are necessary to "hide" those areas of the images that we do not want to be applied in the 3D model, such as dark areas or too bright.

In turn, it should be noted that all images have a common area that is masked. This area refers to a white strip at the bottom of the image (Figure 20). That white stripe is the area that is not scanned by the scanner every time a measurement is taken.

Figure 20. Example of the blank space at the bottom of the picture

7.4 Using Agisoft Metashape

In this software it has been useful mainly to reproject the images and give texture to the 3D model. For this, the following steps have been followed:

- Import HDR images. In this step it is very important not to rename the images.
- Import the masks of each image. Masks created previously with Photoshop.
- Import the model in ".obj" format

Once all the necessary data has been imported, the camera parameters must be changed to spherical. Once this action is performed, a script created by Agageldi samedov must be executed. ("Samedov/Texturizing-FARO-SCENE-model-in-Agisoft-Metashape: Re-projecting the texture of the 3D meshes created in FARO SCENE in Agisoft Metashape,")

The scrip was used to georeferenced each image in the chunk coordinate system by converting the scan locations into the Agisoft Metashape camera transformation matrix.

Since in this project each image will have a different mask from the previous one, a small modification in the script was necessary. In Annexes it could be found the modified script. To be able to execute this script it is necessary to have the file of positions and orientations of each image.

Once the script has been executed, it is possible to see how all the images have been referenced and their positions are seen within the 3D model.

As it can be seen in the following image, there are two options to create the 3D model with the texture "build model" and "build tiled model". The tiled model has been chosen because it can mean an improvement in UNITY in terms of performance, and the configuration of certain parameters that have been applied.

The tiled model is the same as the normal model but let's say it is as if we divided the model into subparts and those parts in turn into other small subparts, thus creating an iterative process that in this case comes up to be repeated 7 times. In this way the texture is in the form of small images with better quality. The following scheme can help better understand the end result of a tiled model:

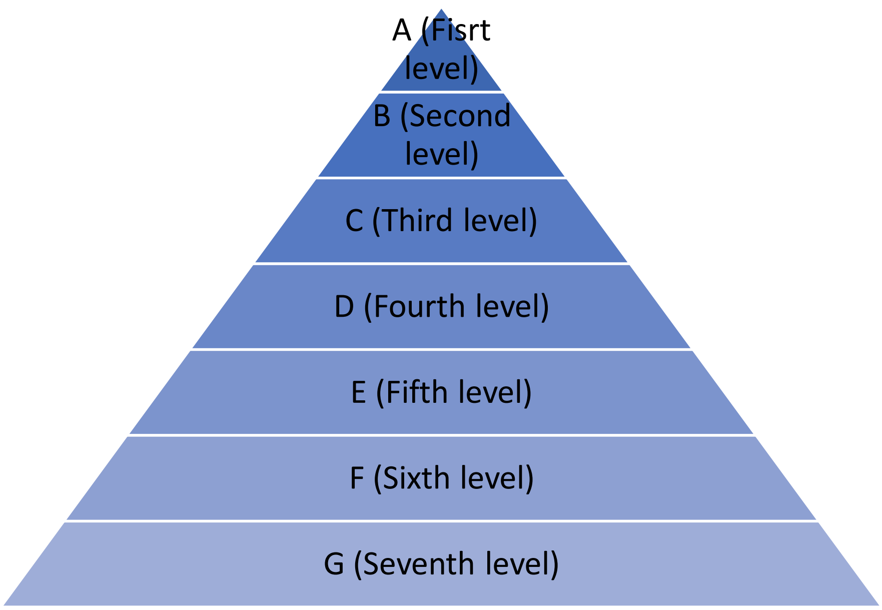


Figure 21. Example of a tiled model structure.

The level of a tiled model can vary depending on how big is and how many details has the mesh. In this project all the meshes have between 6 and 7 levels. The first level has as result one mesh and one image that belong to that mesh and give its texture. The second level is a division of the first level, the third level a division of the second and so on. At the end, the last level could have hundred meshes and pictures. This hundreds of division are useful in UNITY because for example, the mesh of a room is divided in small meshes and only the parts that are close to the view of the user will be charge with more detail, and the ones who are far away will be charge with less detail.

Regarding the steps that has to be follow in Agisoft Metashape, to create a tiled model it is necessary to define a bounding box and place the model inside it.

7.5 Using UNITY 3D

In this program where the final visualization and the result of this master thesis is carried out.

As mentioned above, there is already a project in UNITY. The main things to do is replace the old meshes that are in the project with the new meshes, apply the occlusion culling, improve the texture and create colliders for the configuration of the performance of the walkthrough. The walkthrough script, which is defined in the "MainCharacter" object, has not been modified since it was created to work throughout the whole project. There is a script related to the occlusion culling, since not every mesh has the save level in the tiled model, this script was changed. The script used could be found in the Annexes.

7.5.1 Occlusion culling, frustum culling and backface culling.

Since this project works with a huge amount of meshes, occlusion culling is a useful feature due to this feature is in charge to disable rendering objects that are hidden (occluded) by other objects in the view of the camera. The occlusion culling also include itself the feature frustum culling, which is useful to only render the object that are in the currently camera's viewing area.

In the following images it can see an example of these two features applied (*Figures 22 and 23*):

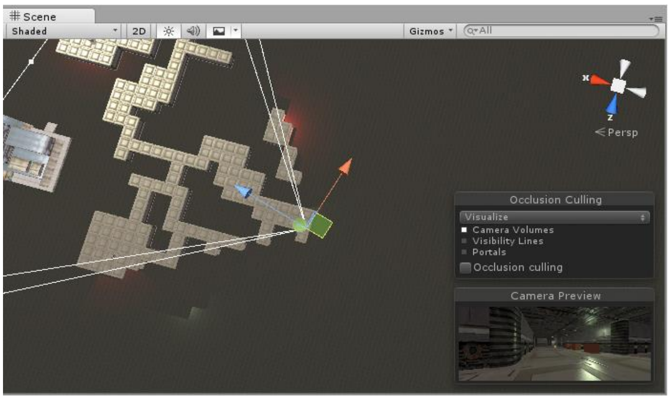


Figure 23. Frustum culling. (Unity – Manual: Occlusion Culling)

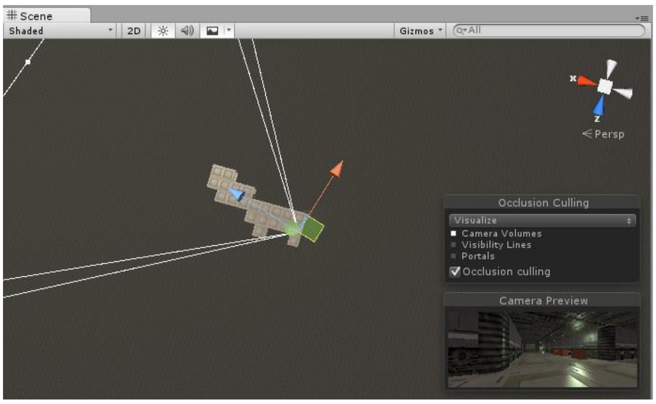


Figure 22. Occlusion culling. (Unity – Manual: Occlusion Culling)

On the right side appears a representation of how looks like the frustum culling, and on the left side the occlusion culling is activated. As it can be seen, the objects that are hidden by other objects disappear, but this not affect to the camera preview.

The backface culling, as was explained in previous chapters, is useful to not render the back face of the meshes. Both backface culling and frustum culling are automatically activated, but occlusion culling has to be set up.

The occlusion culling uses the visibility of the camera to calculate the objects that are visible and the objects that are hidden by other objects, building a hierarchy of potential visible objects. So that the Occlusion culling will only show the object that are visible, and not the objects that are hidden (independently if they are in the view of the camera). As a result, only those objects will be render, so that means less memory consuming, reducing the number of draws calls and increasing the performance of the project.

7.5.2 Colliders

The colliders can be defined as objects or shapes that has physical properties for the purpose of physical collision. For example, in the walkthrough, the user cannot go through an object, wall or column; so, it is necessary to create collider in order to make the user collide with these objects. The collider is invisible so the user cannot see them but can collide with them.

There are different types of colliders. There exist mesh colliders, which are basically the mesh a second time but invisible and with physical properties. These are often more efficient and indistinguishable in gameplay, but more memory consuming. Since, it is wanted to perform the project in an easy and faster way, another type of colliders were used (primitive colliders).

The simplest colliders are primitive colliders such as Box Collider, Sphere Collider and Capsule Collider.

7.5.3 Improving the texture (Shaders and materials).

Some testing has been done to try to improve the texture within the UNITY Project. For this, information about the parameters has been sought when applying an image as texture. In the UNITY project the texture is applied as “albedo” and the characteristic shader is “standard”, however, by changing the shader parameters to “Legacy shaders” and “diffuse” the texture is seen with less brightness, more natural and defined colour.

Then, In the next image it is possible to see the difference by using “standard” shader and “Diffuse”. (*Figure 24 and 25*)

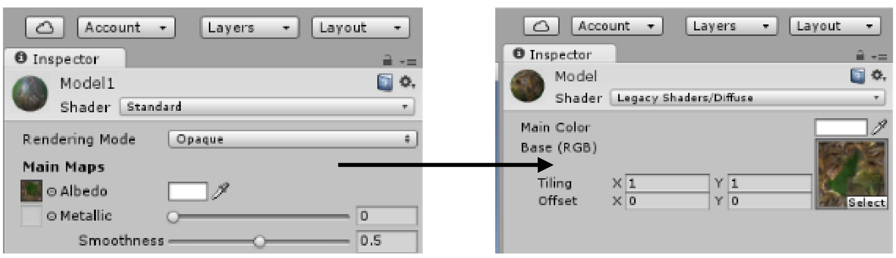


Figure 24. Changing texture parameters

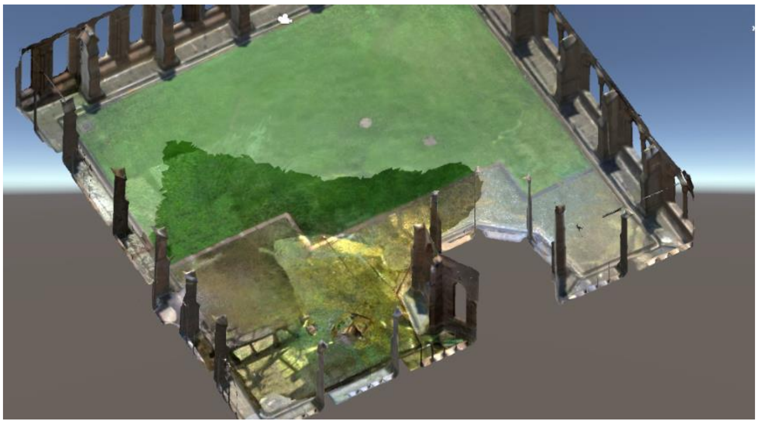


Figure 25. Result of improving texture.

8. Practical part 1: Improving the general project visualization

In this part the process related with the general project, which is involved with the 16 rooms, will be explained.

8.1 Testing parameters for the HDR images

As it was pretended to improve the texture of the 3D visualization, a series of tests were carried out prior to the scans that take place in the monastery. The purpose of the tests was to find out which parameters were the most appropriate to obtain a better image quality, and especially so that the image does not come out too brightly. (Since there were many problems due to the large windows that were in most of the different rooms of the monastery).

For that, a testing was carried out in the photogrammetry laboratory at the university. These are the settings that were used (*Table 1*):

	HDR Mode	**Time (min)**
scan_000	2x	7,30
scan_001	3x	9,09
scan_002	5x	12,00

Table 1. Scan properties.

All the scans were imported in the same project and processed with the same parameters. As was expected, the best resolution was achieved with the level 5. This is because the level 5 takes more pictures with different exposures to create one with better quality. As the same time, as it can be seen, the scanning time increase. As the result image is with a very high quality (compare with the images that were taken with other levels) it can conclude that the scan timing worth it.

Then, the result of the scan will be shown in the following pictures (*Figures 26, 27 and 28*).

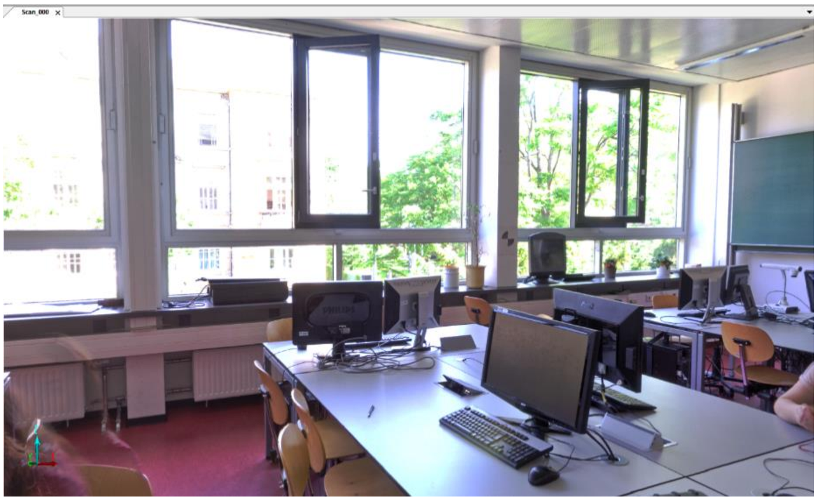


Figure 26. Testing with level 2x.

Figure 27 Testing with level 3x.

Figure 28. Testing with level 5x.

8.2 Analysis of old scans

In this chapter, the different rooms, the scans that were taken and the HDR images are going to be analysed. The numbers of the Scans in the pictures are related with their names in the SCENE project.

8.2.1 Innenhof

The Innenhof is an open space, which is quite problematic in terms of brightness. In fact, the biggest problem with this room is the south facade as it is completely illuminated, and the images of that part are not very good for creating a texture. The problem of this part is that if a scan is repeated, almost every scan should be repeated since at first when the measurements were taken it was winter and there was snow, and these second measurements were made in summer.

In the following picture appears the position of the scans (*Figure 29*):

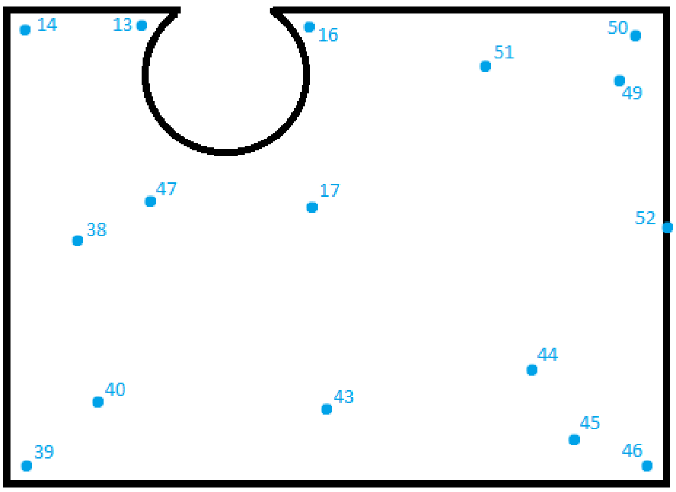


Figure 29. Scan position in Innehof.

The snow was located on the ground and on the roof of some of the facades therefore for these areas new images will be used to give texture, while with the texture of the walls both data can be used (those of winter and of summer). Scans should be taken in the middle of the area to be able to have visibility of the upper part of the facades and the ceiling, and thus solve the problem of luminosity with the facade south.

8.2.2 Laienrefrektorium

It is a very spacious room with 7 central double pillars. This room has a wall completely fully with windows. In the central-right part of the hall there is a stage, since this room is usually used to make concerts.

In the following picture appears the position of the scans (*Figure 30*):

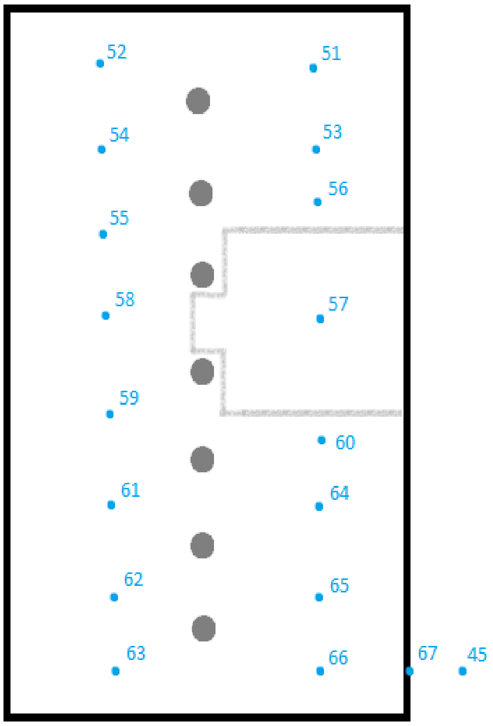


Figure 30. Scan position in Laienrefrektorium.

As previously mentioned, this room is sometimes used to perform concerts, therefore, during the first measurements the room was completely full of chairs. What was done was to remove the chairs from the area to be scanned and pile them up in another part of the room. Subsequently, the part where the chairs had been piled up was scanned and moved to another part of the room. Thus, scanning the entire room "in parts."

This room has double work added because not only it was necessary to mask the windows, but it will be masked the areas where there are chairs. But that is only a problem related to time, it is not necessary to repeat any scanners due to the chairs since all the areas of the room have been scanned correctly. In this room, the scans in front of the windows should be repeated due to an excess of brightness is not possible to get a good texture of that area.

8.2.3 Annexräume

These rooms do not have many windows, but have three large glass access doors, which allows good lighting in the room. It also has two lines of artificial light on the top. These rooms are connected to the room called "Grosser Keller" through a glass door. It also has a showcase with information about the monastery.

In the following picture appears the position of the scans (*Figure 31*):

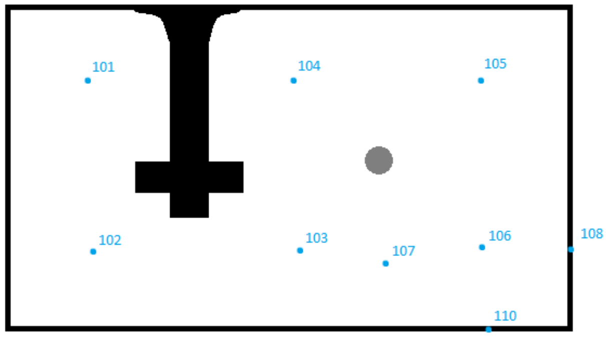


Figure 31. Scan position in Annexräume.

The problem in this room had to do with the points of artificial light located on the ceiling. Due to this, the light affected the walls and part of the ceiling and therefore, the images taken by the laser scanner could not be used to give texture to that area. In turn there was also a bit of excess brightness around the two glazed entrances.

8.2.4 Grosser Keller

This room belongs to one of the oldest parts of the monastery and access to the public is currently restricted. It is a fairly dark room because it has no windows, but the lack of light is compensated by artificial lights located on the floor near the walls. The room has three entrances, two connect with other rooms and the third one connects to the outside. It also has two central pillars.

In the following picture appears the position of the scans (*Figure 32*):

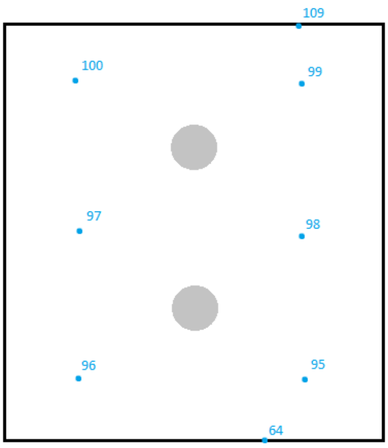


Figure 32. Scan position in Grosser Keller.

Due to the spotlights on the floor, the nearby walls are fully illuminated and therefore do not look good in the images. It would be necessary to repeat the scans that are close to the light bulbs. In general (by passing the areas bathed in the spotlight) the images are dark, but it is possible to work with them.

8.2.5 Frateria

The room contains several large windows, two pillars and a corridor without windows that leads to a door to another room (so it is quite dark). The problem with the texture in the windows is repeated again due to the excess of luminosity.

In the following picture appears the position of the scans (*Figure 33*):

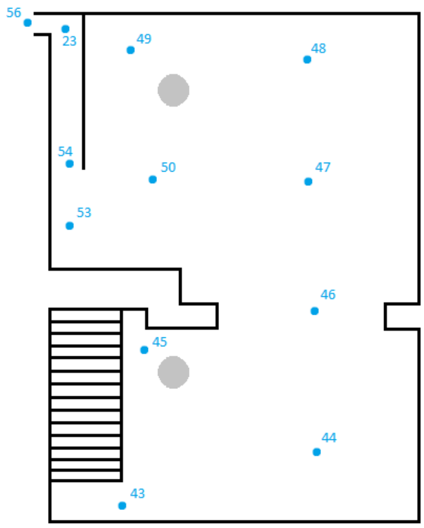


Figure 33. Scan position in Frateria.

The scans that are close to the window should be repeated to improve the texture and quality of the images. Scans number 23, 24 and 50 are too black, while 53 is almost completely white due to excess brightness. In the case of scans 23 and 54, nothing more can be done since it is a corridor without windows, but in relation to scans 50 and 53, they should be repeated.

8.2.6 Ern

Room 5 is the general entrance for visitors, it is a room with three doors. One of the doors is the entrance to the monastery museum. In the middle of the exit is the entrance device to allow access to tourists, it is a metal and glass platform.

In the following picture appears the position of the scans (*Figure 34*):

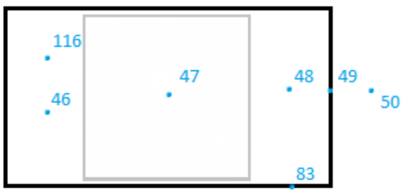


Figure 34. Scan position in Ern.

The scans in this room are pretty good, the only thing that could be improved a bit is the ceiling. In the upper part there are some painted drawings, one of them does not look quite good due to the excess of brightness caused by an artificial light.

8.2.7 Cellarium

This room is the monastery museum; therefore, it is full of "monuments", objects and pieces of the monastery. These pieces are distributed on the floor, but near the walls of the room and some of them are inside stained glass. Physically it has two central columns and two stairs with two doors. One of the doors opens to the visitor's entrance hall.

In the following picture appears the position of the scans (*Figure 35*):

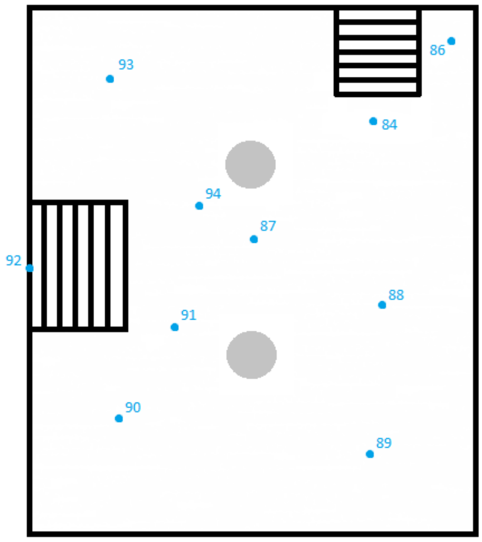


Figure 35. Scan position in Cellarium.

The corners where the 93 and 90 scans are placed, have too much light to be able to fully appreciate the details of the figures and monuments of that part. In both corners there are more than one figure, so if it is wanted to have more detail and cover all angles it should make extra scans to have the full texture. Finally, the scanning of the ladder does not look quite right, it would be advisable to repeat it.

8.2.8 Kalefraktorium

It is a very small room with a window in the background through which not much light enters. In total it has 5 scans. In the following image, the protruding upper part refers to the window, which is shaped like a projection on the wall.

In the following picture appears the position of the scans (*Figure 36*):

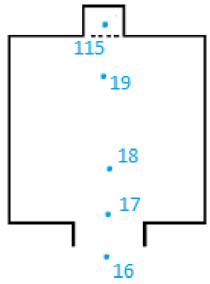


Figure 36. Scan position in Kalefraktorium.

In general, most of the images are quite dark due to the low incoming light in the room, it could be possible to work with them, but it would be preferable to repeat some scanning and try to get it out with a little more light.

8.2.9 AusKreuzgang

This room is an exit or entrance, but only for authorized for the employees that has the key. It is a room with two entrances (one of them glass) therefore it is a well-lit room.

It has 4 holes in the wall, forming two aligned columns. These holes can be seen in the following image on the left side.

In the following picture appears the position of the scans (*Figure 37*):

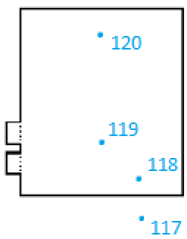


Figure 37. Scan position in AusKreuzgang.

The room's scans are apparently fine, the only thing missing is that there is no texture from the top of the 4 holes in the left part of the room. This may be due to the height of the laser scanner, so it would be advisable to repeat a scan in that area with a lower height.

8.2.10 Kapitelsaal

The room is full of windows, in part west it has windows that overlook the cloister hall. The east facade has windows with coloured windows. On the right side of the facade X there is a double floor 0.5m above the normal floor, leaving a gap below; This can be seen in the following image where the double floor is represented by a dashed line. In turn, the ceiling is filled with paintings and mini sculptures with floral motifs and representing animals.

In this room there are a total of 17 scans. Scans that are greyish mean they are located on the double floor, at the bottom. Therefore, they are not at the same ground level as the others. It could be said that they are with “a negative level in relation to the other scans”.

In the following picture appears the position of the scans (*Figure 38*):

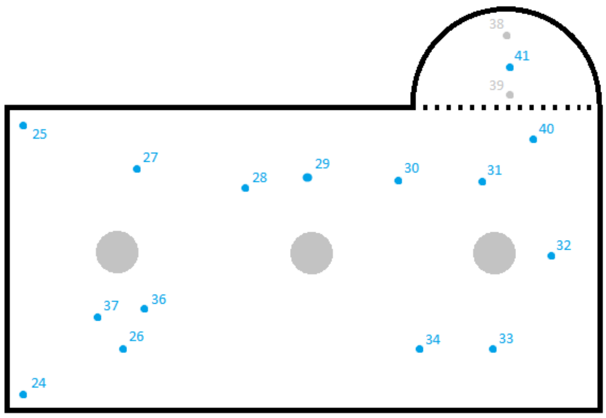


Figure 38. Scan position in Kapitelsaal.

There are some scans that present too much lighting, but as there are other scans that are nearby it is not so worrying. The area that represents the glazed windows has too much presence of light, so much, that hardly see the colours of the windows. Therefore, some scans should be repeated in order to obtain a good texture in the Windows.

8.2.11 Ostern

This room is the “entrance” of Marienkapelle, and does’t have any windows. For that reason this room is quity dark.

In the following picture appears the position of the scans (*Figure 39*):

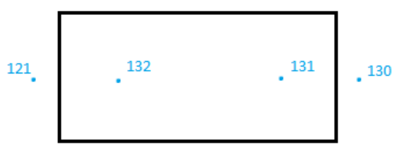


Figure 39. Scan position in Ostern

Since the HDR images could be edited to get more brightness, it is not neccesary to make any measurements in this room.

8.2.12 Marienkapelle

This a long room with a lot of windows in the south facade, that makes that the images of this room are with a lot of brightness. The windows cannot be seen in the images due to sunlight.

In the following picture appears the position of the scans (*Figure 40*):

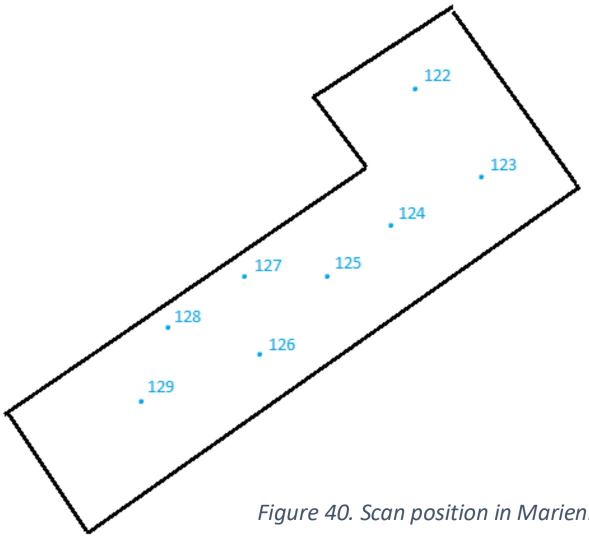


Figure 40. Scan position in Marienkapelle.

Since the images are bright, some measurement had to be taken in this part. Overall infront of the windows to get a good image for the texture.

8.2.13 Kreuzgang

The Kreuzgang is the part of the corridor of the monastery. All images in the hall are correct except those in the east. The images of that area are too bright, and it is difficult to differentiate the floor and walls. There is a small room connected with the Kreuzgang, and in this part the images do not have a good quality.

In the picture of the right, appears the position of the scans (*Figure 41*):

The scans on the east part and inside of the annex room had to be repeated.

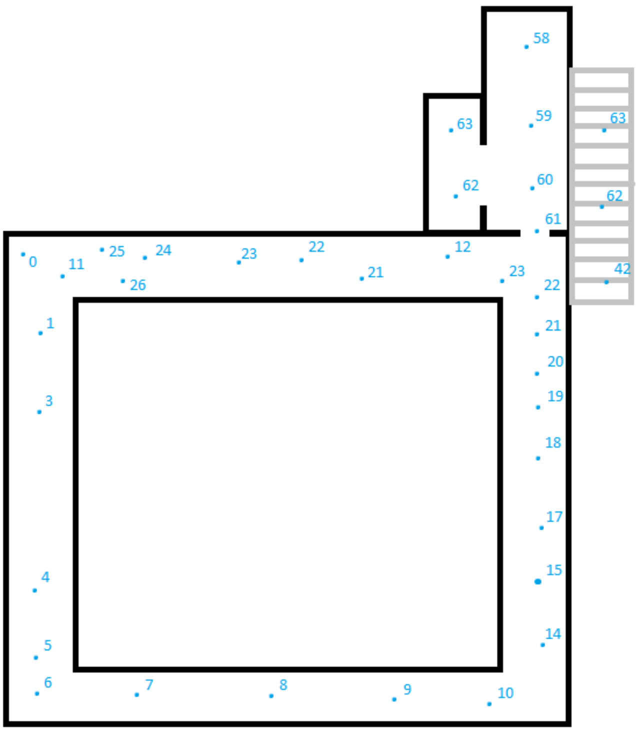


Figure 41. Scan position in Kreuzgang.

8.2.14 Herrenprefektorium

This room is fully completed with big and long windows. Those windows are coloured and in some images this is not appreciated. Furtemore it has a hole in the wall, and the images from that part are quity dark and with no good quality.

In the picture of the right, appears the position of the scans (*Figure 42*):

For that room it would be recommended to scan again overall the part of the windows, and the hole of the wall.

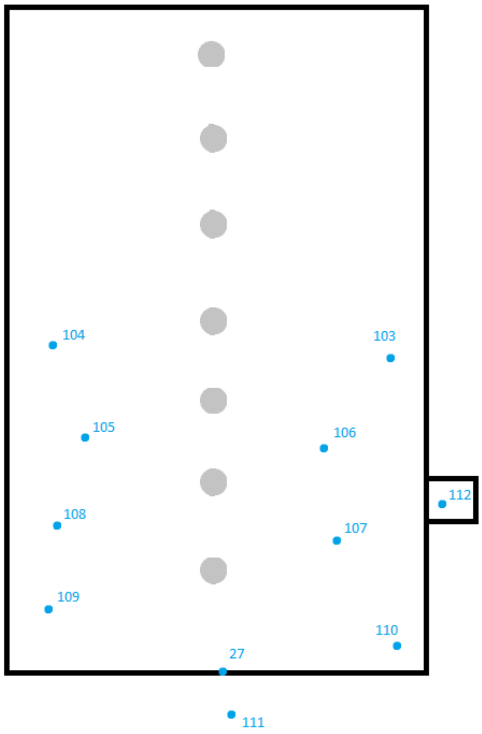


Figure 42. Scan position in Herrenprefektorium.

8.2.15 Brunnenhaus

This room has a circular shape and its walls has big windows. It is well iluminated, and there were taken a lot of scan to have the completely shape of the fountain that is inside of this room.

In the following picture appears the position of the scans (*Figure 43*):

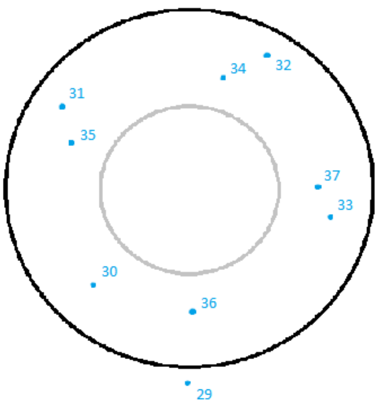


Figure 43. Scan positions in Brunnenhaus.

For that room it is not needed to make new scans due to the images are good enough to work with.

8.3 General conclusion: valid parts and parts to be repeated

In general, in most rooms the problem is related with the brightness and non-brightness in the images. The most problematic areas were:

- Windows and doors
- Rooms with artificial light bulbs
- The rooms without artificial light or natural light.

Next, examples of each of the three cases will be shown.

In the Kreuzgang it is possible to find a great example of how excessive incoming light in the windows does not allow to obtain a good texture of the window itself and its surroundings.

Grosser Keller is a clear example of how artificial light at the top affects the texture in the nearby walls where the light hits.

Kalefaktorium is a small room with a window at the bottom through which there is not too much light, so that the room is usually quite dark, and the texture cannot be appreciated. It is generally better for an image to be darker than to have more brightness, because the dark image can be edited and made lighter, but sometimes if it is too dark or even useful.

To sum up, scans were repeated in all rooms except for Brunnenhaus and Ostern.

8.4 Approach of the scans in Maulbronn

When making the scans, considering that the biggest problem was the light, the best time ranges were in the morning of 7-10. Additionally, it was also necessary to take into account where the sun was rising and the orientation of the windows with respect to it. Finally, the visiting hours of the monastery were from 9 in the morning to 5 in the afternoon, and mostly visiting groups with tourist guides were carried out in the morning from 11. The measurements were carried out for 3 days, mostly in the morning.

The first day the six rooms are scanned, the second day the five rooms, and the third day the three rooms. A total of 69 scans were taken.

As what is pretended in this project is to use the old and new data, the old scans will be used along with the old images (masking the problem areas later, as far as the texture is concerned) and the new data, but these with a lower level of detail in terms of taking points (since that part is covered with old scans), and in turn trying to improve the images in HDR increasing in level in this way. Therefore, summarizing, in the new scans a resolution of 1/10 and a quality parameter of 3x have been used, the distance between points every 10 meters is 15.3 mm and as it is suspected that the highest level was not used of the HDR image mode, in this case level 5 has been used in almost all scans.

Next, each room will be examined one by one. In the image that will appear in each room, the orange scans refer to the new measurements and the blue ones refer to the old measurements.

8.4.1 Innenhof

This particular one was taken early in the morning. Thus, the sunlight did not directly affect to that part of the monastery. Additionally, it has to be commented that that morning was cloudy, which facilitates the taking of the images in the sense of the quality of the light.

As can be seen in the following image, 9 new scans have been taken. (*Figure 44*)

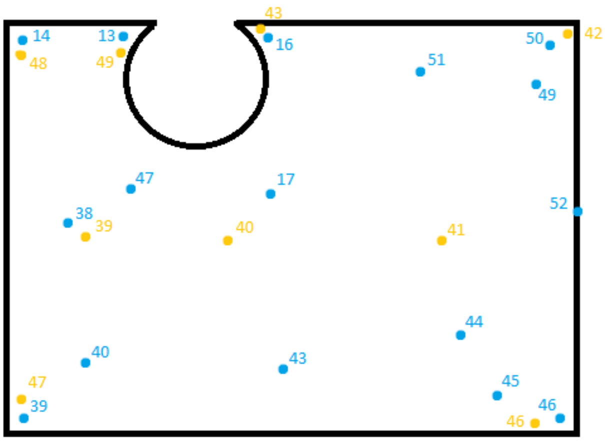


Figure 44. Scan positions in Innehof.

In the chapter 8.2.1 was mentioned that there was snow on the floor in the images of the old scans. Here it is possible to see the snow, the brightness and the difference between the old and new scans using the level 5 of the HDR image mode (*Figure 45 and 46):*

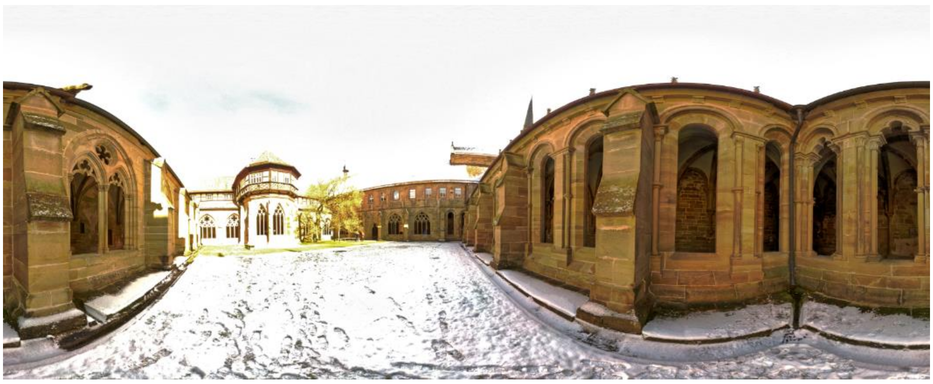


Figure 45. Picture from the old scans

Figure 46. Picture from the new scans

8.4.2 Laienrefrektorium

At the time of making the measurements there were chairs in the room. So for that, all the chairs in that area moved. A total of 8 scans were carried out, focused, as mentioned before, to obtain the best texture for the windows. When rethinking the scans, an attempt was made to have a data collection cover more than one window, in order to minimize the number of scans.

In the following picture appears the position of the scans (*Figure 47*):

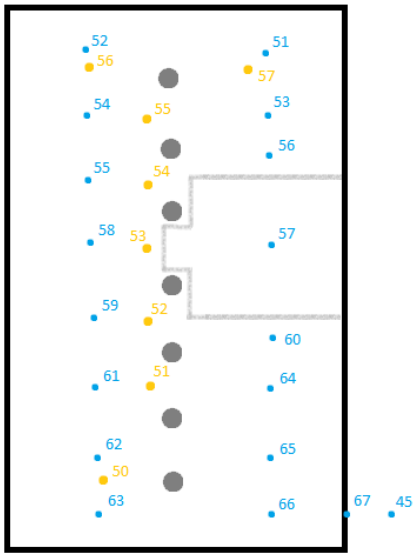


Figure 47. Scan positions in Laienrefrektorium.

8.4.3 Annexräume

In this room in total 7 scans were taken with level 5 of the HDR image mode. We tried to do some scans with the lights switched on and others with the lights switched off.

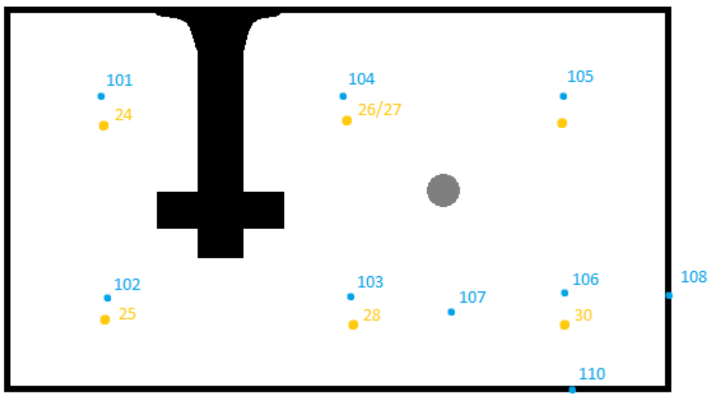


Figure 48. Scan positions in Annexräume.

8.4.4 Grosser Keller

The planning for this room was a bit complicated since the light bulbs could not be turned off, because the room will be completely dark. Therefore, it was decided to cover the spotlights of the ground with white sheets, thus allowing the light to pass through but not as much light as at the beginning since the white leaves attenuated the light, acting as a kind of filter.

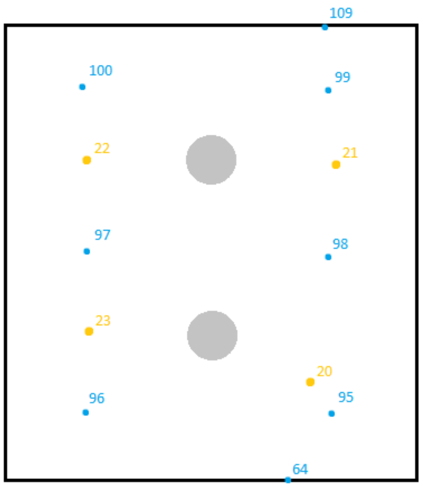


Figure 49: Scan positions in Grosser Keller.

8.4.5 Frateria

Three scans have been made to improve the texture in the windows and improve the brightness in the scan 53. These scans were performed in the early morning so that the sunlight did not affect the windows so much.

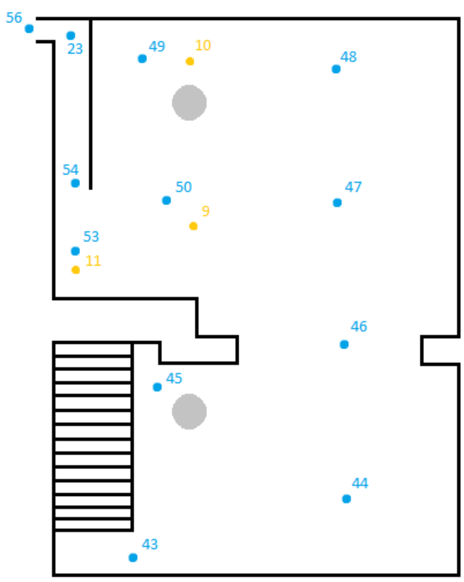


Figure 50. Scan positions in Frateria.

In the following pictures it is possible to see the difference in quality of the images taken in the old and new scans. (*Figures 53, 54 and 55*)

Figure 51. Image from the old scans.

Figure 52. Image from the old scans.

Figure 53. Image from the new scans.

As it can see in the last picture (*Figure 53*) eventhough using the level 5 of the HDR mode, sometimes the brightness still appear. However, the quality of the image is icreassing significativily.

8.4.6 Ern

A single scan has been carried out in this room to obtain a good texture of the ceiling drawings. This scan was done early to avoid the time slot with tourists since that room is the entrance and exit access to the monastery.

Figure 54. Scan positions in Ern.

8.4.7 Cellarium

In total in this room 7 scans have been carried out to cover all the areas that have been previously mentioned. So, 3 scans for each corner and one more for the stairs. Measurements began to be made outside the visitor's schedule, but in the end tourists appeared as it was a very important part of the monastery and visited by tourists. However, this is not very significant since visitors appear in the other part of the room where no scans have been performed; It only implies a small increase in processing time to mask tourists.

So the result is as follows (*Figure 55*):

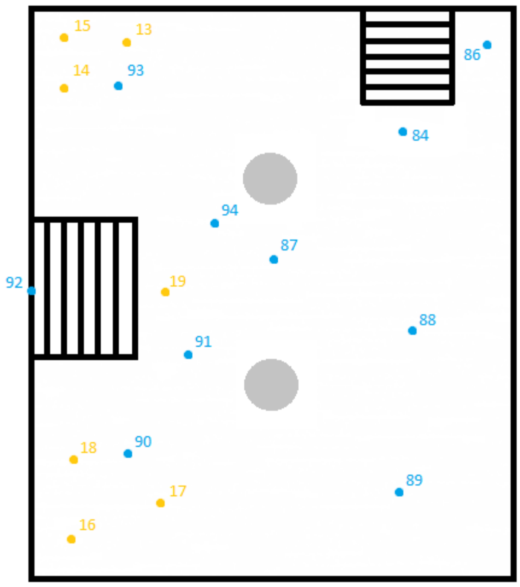


Figure 55. Scan positions in Cellarium.

8.4.8 Kalefraktorium

In this room, 3 different scans have been performed but in the same position. What was done was to perform a first scan with the natural light that enters through the window, trying to see if with the mode 5 of HDR image could get better results and a slightly less dark image. The other two scans were also made with level 5 but the flashlight of a mobile phone located on the ground was used to bring some light to the room.

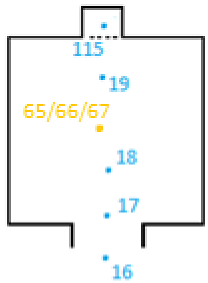


Figure 56. Scan positions in Kalefraktorium.

8.4.9 AusKreuzgang

In this room in total 1 scan was taken. The scan was placed on the floor in order to obtain the top text of the holes in the wall.

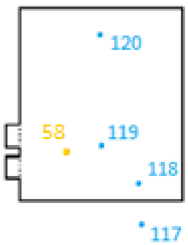


Figure 57. Scan positions in AusKreuzgang.

8.4.10 Kapitelsaal

In total, 2 scans have been carried out in front of the windows of the facade east. The scan was done around noon, when the light incident in that area is not so strong, thus being able to better capture the colors of the windows and not have So many problems with luminosity.

So the scan positions are as follows in the right image (Figure 58):

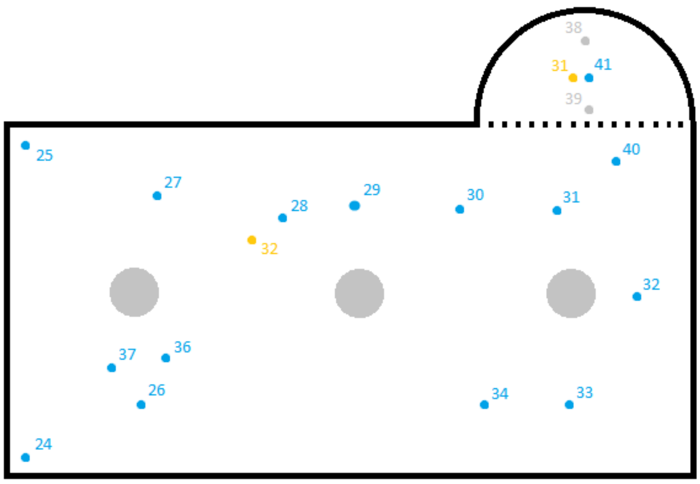


Figure 58. Scan positions in Kapitelsaal.

In the following images is it possible to see how improve the quality in the windows the use of the level 5: (*Figures 59, 60, 61 and 62)*

Figure 59. Image from the old scans.

Figure 60. HDR image from the new scans.

Figure 61. Image from the old scans.

Figure 62. Image from the new scans.

8.4.11 Marienkapelle

In this room, four scans were taken in order to get a better quality for the windows. Since later the images are projected to get the texture, the scans were placed in front of the windows.

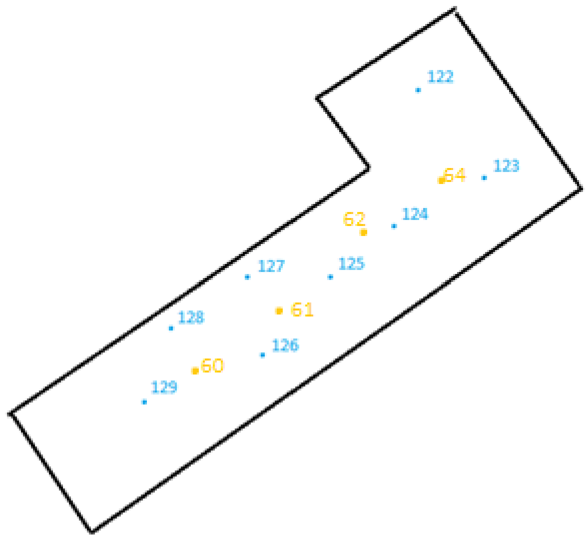


Figure 63. Scan positions in Marienkapelle

8.4.12 Kreuzgang

It this part of the monastery 11 scans were carried out. In the part of the annex room 5 scans were taken, and in the part of the Kreuzgang 6 scans. In the corridor also the scans were placed infornt of the windows to get a better quality image for projecting the texture. This part was made in the early morning when the sunlight does not affect much.

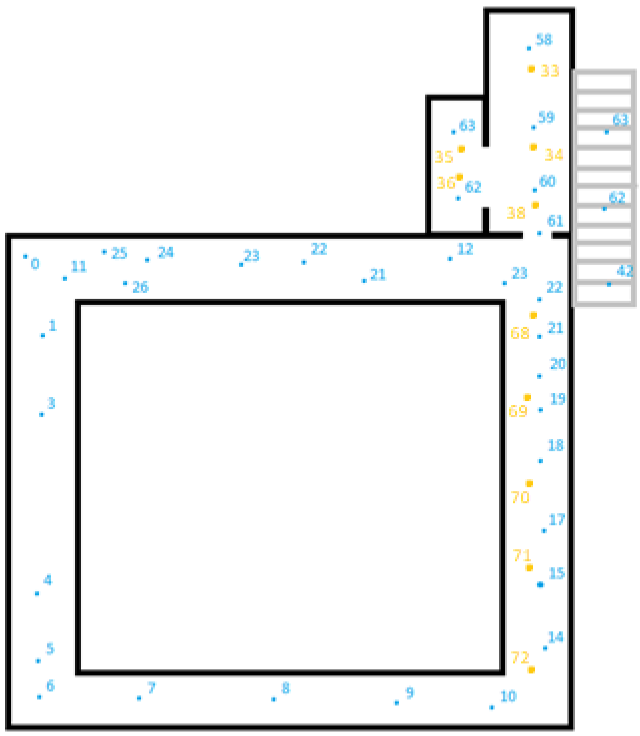


Figure 64. Scans position in Kreuzgang.

8.4.13 Herrenprefektorium

In this room, 7 scans were taken in order to get a nice image for the colored windows. One of the scans was repeated becouse the old one was too dark due to it was placed in hole on the wall.

So the scan positions are as follows in the right image (Figure 65).

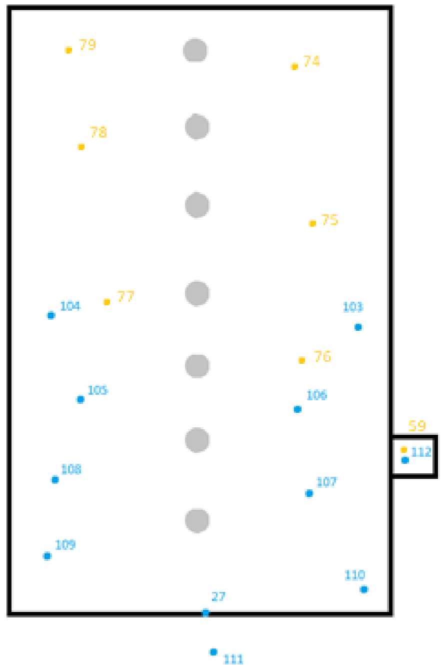


Figure 65. Scan positions Herrenprefektorium

In the following images it is possible to see the difference between the images in the old and new scans. (*Figure 66 and 67*)

Figure 66. Image from the old scans.

Figure 67. Image from the new scans.

8.5 Workflow

In this chapter is going to talk about the Workflow of the Project room by room. AS the process is more or less the same for every room, the process will be explained just once.

8.5.1 Innenhof

This section will follow with the workflow process in “Innenhof”. Then, as shown in Chapter 7, the first software is SCENE.

From this software it is needed to extract the geometry (the mesh), the HDR images and the positions. The way to extract the geometry will be the same for all rooms since all will be created from the existing project in SCENE. Each mesh will have different parameters, since the geometry and size are different, but the procedure will be the same

In order to create the mesh, it is necessary to first create a Clipping Box of the area that interests, in this case it would be “Innenhof”. (*Figure 68*)

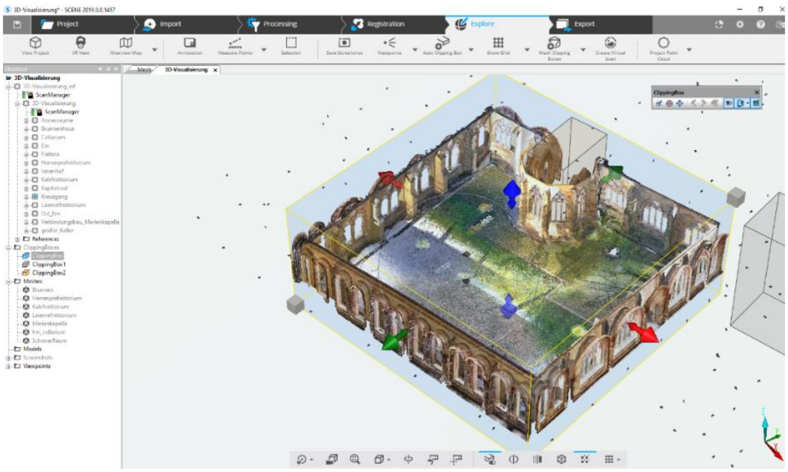


Figure 68. Clipping box of Innenhof.

Once the zone is selected, the mesh can be created, and its parameters defined. After several tests, it has been determined that the best parameters for this area are the following (*Figure 69*):

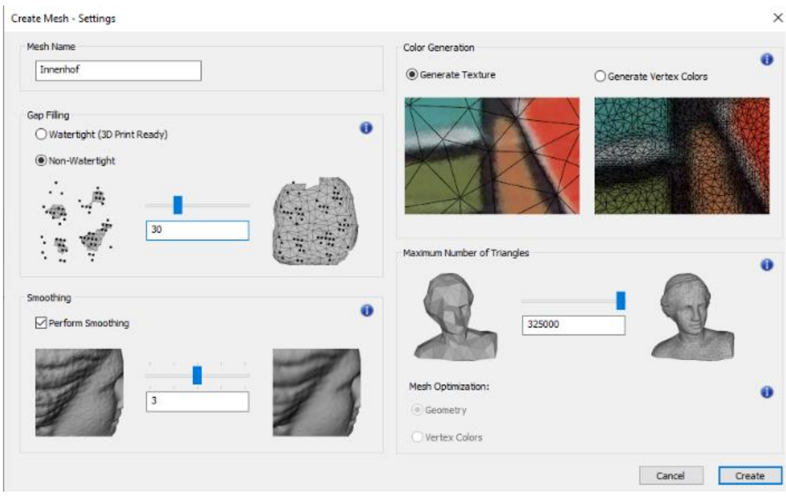


Figure 69. Mesh parameters in Innenhof.

Once the mesh is created, the images and the position file are obtained.

As in this work is not pretended to modify the original project and to obtain the images and positions of the old and new scans (it would be necessary to add the new scans to the project), the old scans will be exported in groups according to the different rooms; Therefore, secondary projects from each room will be created from the original project (with the old scans) where new scans can be imported.

When importing the new scans, these must be processed and registered with the old ones in order to create the point cloud of the project and from there, to be able to take out the positions. Finally, it will be possible to extract the images. It should be noted that the process of extracting HDR images is quite long and may take up to several hours.

Once all the necessary SCENE outputs have been obtained (the mesh, the HDR images and the position file), the next step is to edit the mesh in Geomagic Qualify. Generally, what has been done is to retouch the mesh (so that the windows are in the best possible way) and smoothed the model.

Below it will see an image of what the new exported mesh looks like from SCENE (*Figure 70*):

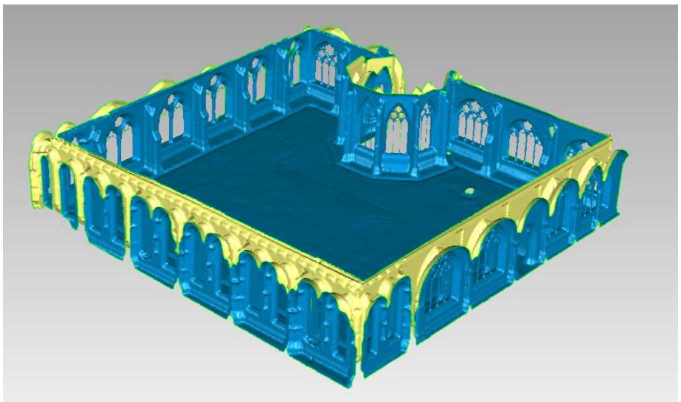


Figure 70: Innenhof mesh.

As previously mentioned, the windows were "cleaned." Below it is possible to see an example of how the mesh looks after removing the remaining parts in the windows (*Figure 71*):

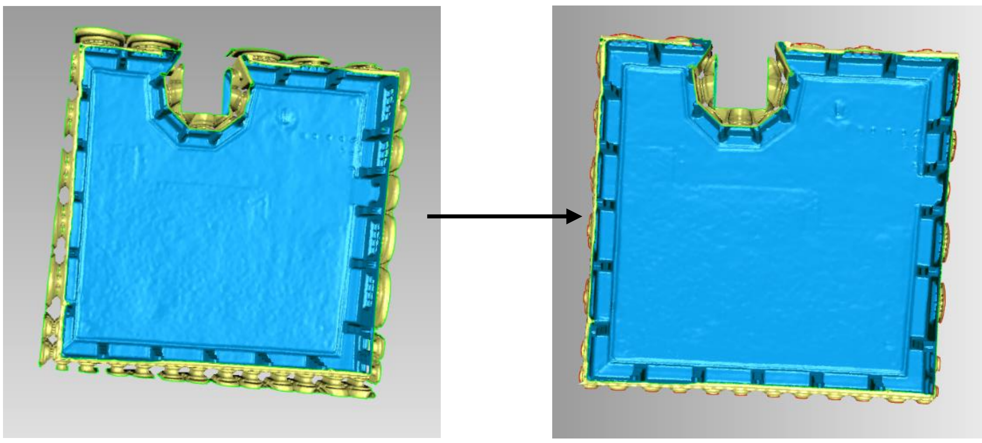


Figure 71. Editing the Windows in the mesh.

Additionally, it was pretended to reduce the number of triangles, using a tool called "Decimate". With this tool, the mesh decreased from 241,457 triangles to 240,008; The "relax polygons" tool was also used to soften the mesh.

Below are the parameters that have been used for both tools (*Figure 72*):

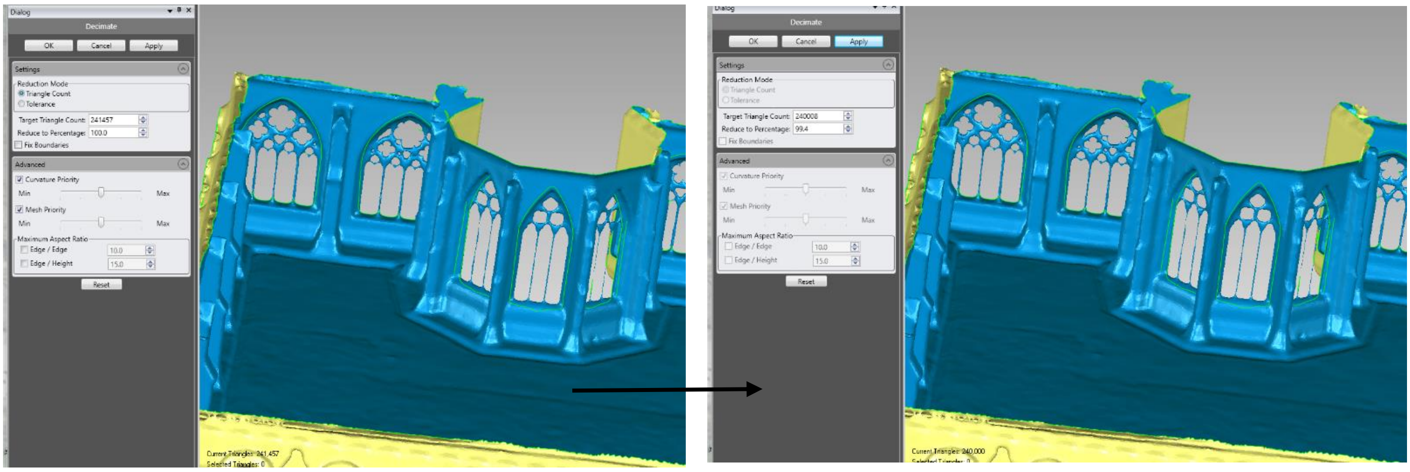


Figure 72. Example Decimate tool.

Below it is possible to see how the model was finally after applying the smoothing tool with the following parameters (*Figure 73*):

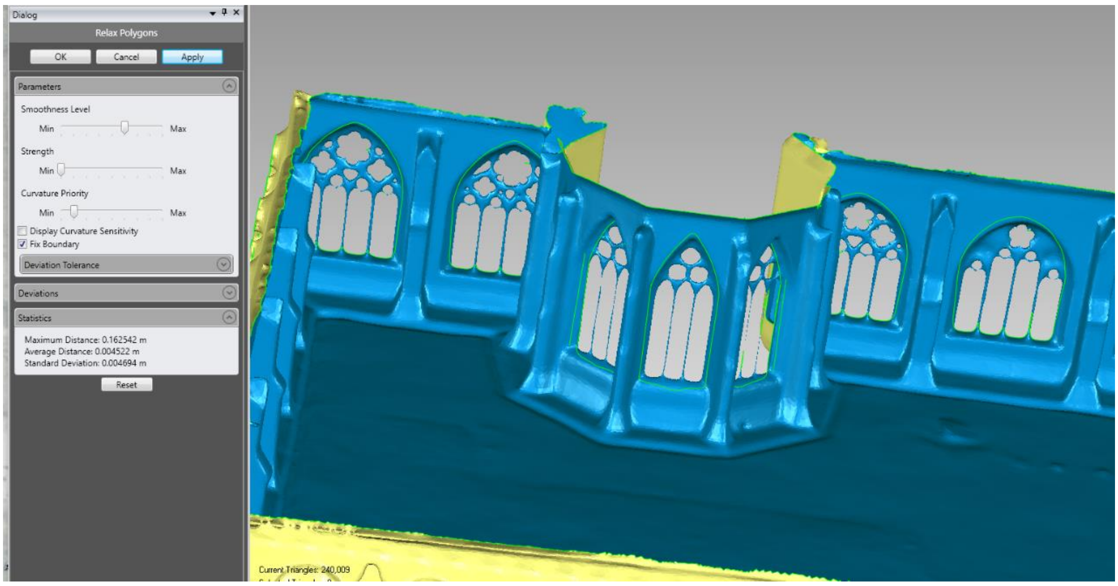


Figure 73. Example Relax polygon tool.

Once the mesh was obtained, the next step was to edit the images with Photoshop. The intensity and brightness of the image were readjusted in most images. Additionally, the new images (since they were with better quality) were edited to erase some wooden benches. When creating the masks, the bright areas, the areas where there was snow, the wooden benches and a tree were camouflaged. Finally, creating the masks also changed the images from “png” to “jpg”.

Finally, in Agisoft Metashape all the data has been imported (the edited images, the masks, the position file and the edited mesh). Once everything is imported, it is necessary to execute the script that it was previously named to be able to reference the images. Finally, the tiled model has been created with the following parameters (*Figure 74 and 75*):

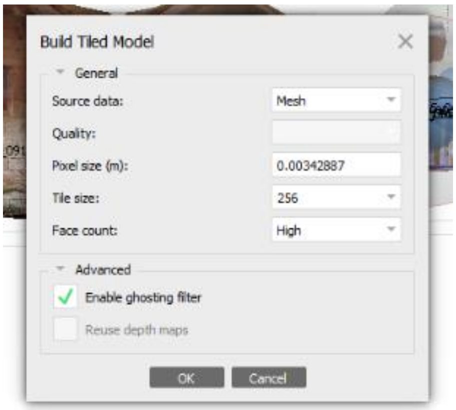


Figure 74. Build tile model parameters.

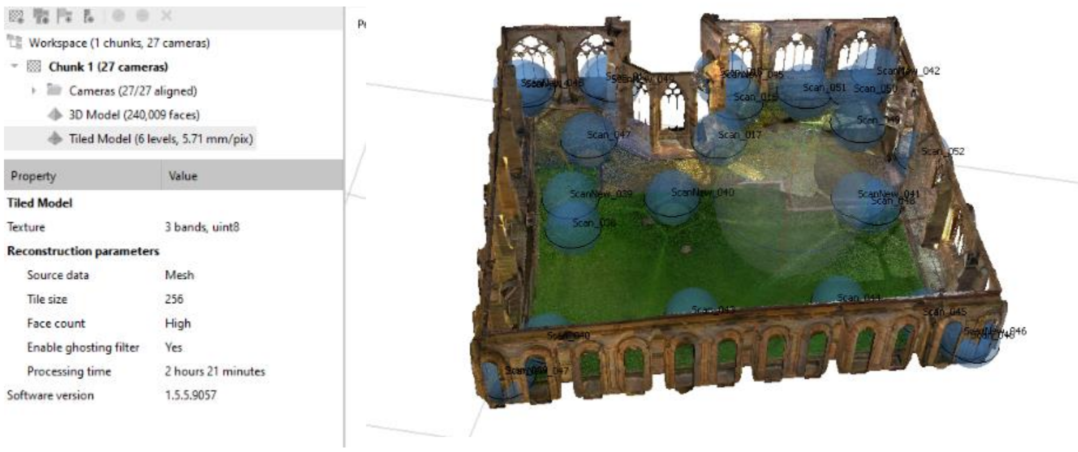


Figure 75. Innenhof tiled model.

8.5.2 Laienrefrektorium

The same steps that have been explained for the previous room will also be repeated in this room.

First of all, the clipping box and the mesh are created with the following parameters, which after several tests have been chosen as the most suitable:

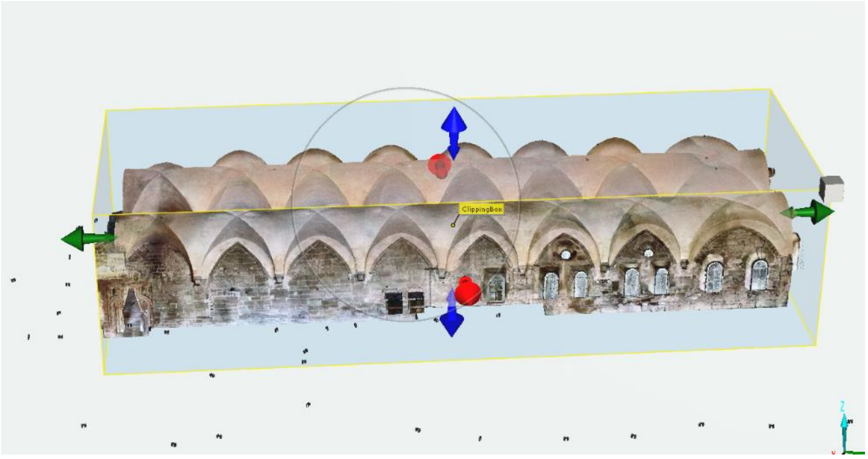


Figure 76. Laienrefrektorium clipping box

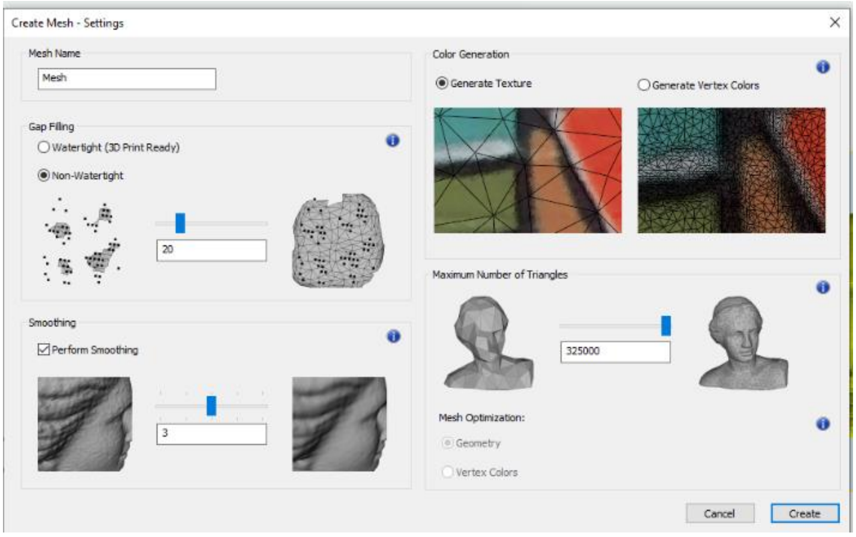


Figure 77. Laienrefrektorium mesh creation parameters.

Then the mesh is imported into Geomagic Qualify for editing purposes. As this room has many windows, it was in those areas where there have been more editions. What was done was to cut the windows, because the mesh created a rough area on the surface of the glass. Since at the time of projecting the images it is better to have a flat area, those areas were cut back and filled in more flatly way.

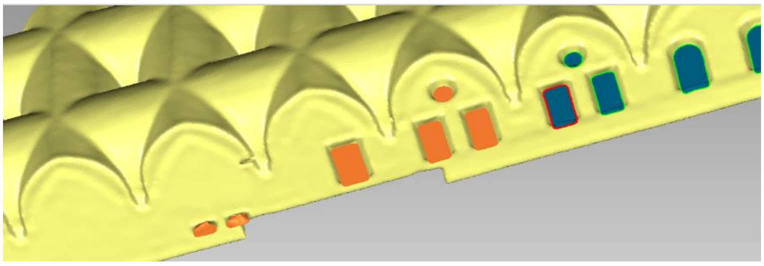


Figure 78. Editing the windows of the Laienrefrektorium mesh.

Then the Decimate tool was tested but the results were not good, therefore this tool was discarded in this room. The Relax polygons tool was also tested. The parameters and the result are shown below. (*Figure 79*)

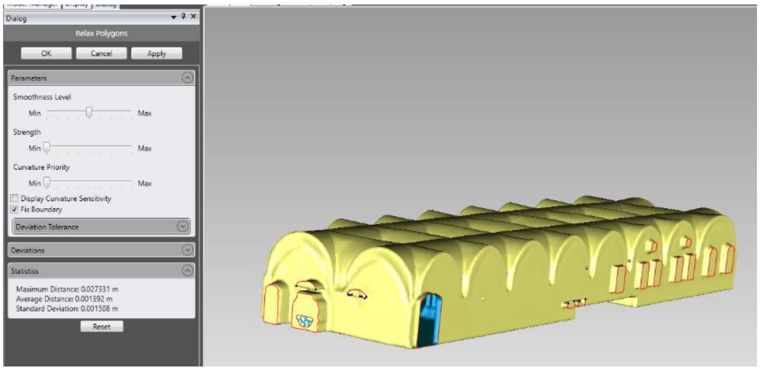


Figure 79. Laienrefrektorium mesh editing tool.

Once that the mesh was edited, it was time for editing the pictures. The images were edited in Photoshop, and at the same time the masks were created. This room was one of the most time consuming regarding the creation of the masking due to there was a lot of chairs at the moment of the measurement.

After getting the masks and the edited images, all the data was imported to Agisoft Metashape to create the tiled model, and this was the result:

Figure 80. Laienrefrektorium tiled model.

8.5.3 Grosser Keller

The workflow in this room was different becouse an error appeared, when adding the new scan to the project in SCENE. The error was that when the scan are registered, the old scan were moved and the positions changed, so at the end the position of the images didn't match with the position of the mesh. To fix this problem, an special workflow in SCENE was followed. This error was also present in the rooms: Kreuzgang, Frateria, Annexräume, and Innenhof.

The workflow is the following; Fisrt, export the old scans as a new project project in SCENE. Then add the new scans in a new cluster. The old cluster with the old scans can be fixed, so then its coordinated won't change.

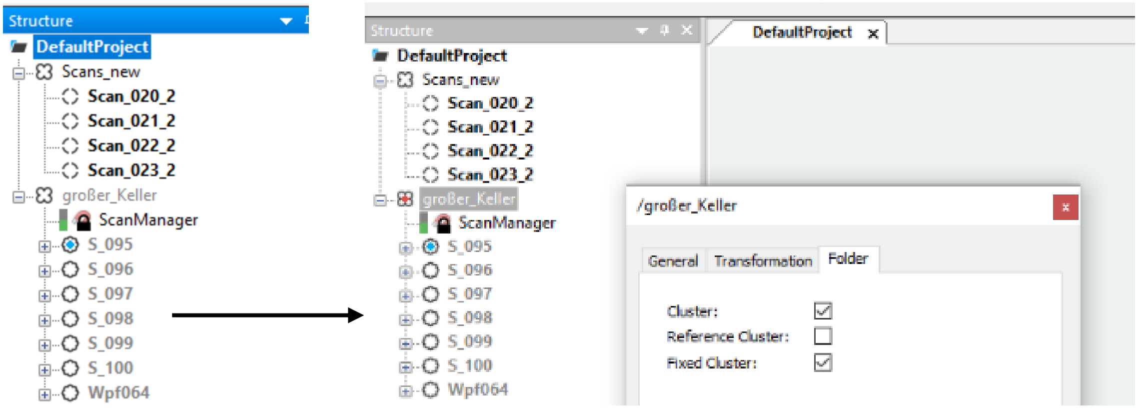


Figure 81. Fixing cluster with the old scans.

In the previous image (*Figure 81*) the cluster called "Grosser Keller" contain the ol scans and the cluster called "Scans_new" contain the new scans.

Once that step is completed, the new scans has to be processed in a normal way. After that a General cluster is created, and the two clusters (Scans_new and Grosser_Keller) are placed inside by maintaing the position, with the option "Move hear and keep position".

Then an visual registration must be done, like in the next image:

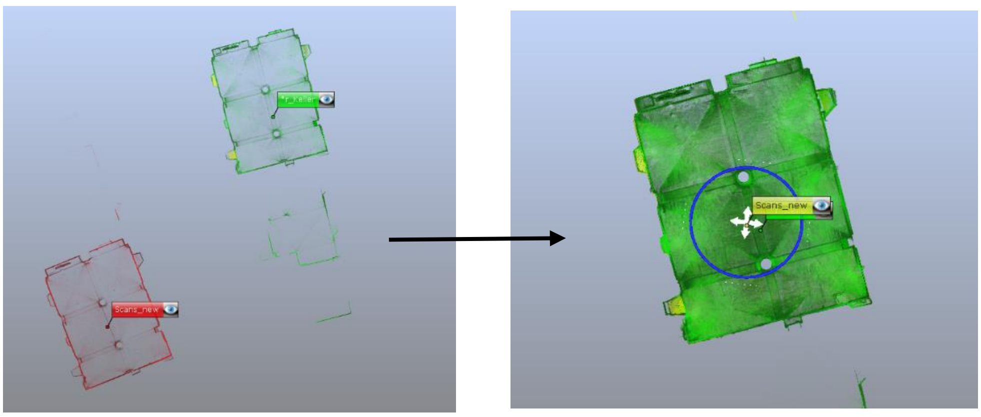


Figure 82. Visual registration process.

Once that the two cluster are in the same place, the next step is to move one scan from the "grosser_keller" scan to the "Scans_new", and choose the option of keep the position while doing this step. After that the cluster called "Scans_new" has to be registred again, by using automatic registration. When the previous step is done (*Figure 82*), the two cluster can be move out from the total cluster, and the everything is match without changingthe original positions.

Figure 83. Final registration.

Once this problem is solve, the workflow of this practical part is the same than the other rooms.

In this room the procces of editing the images in Photoshop was very important since the old and the new scans has a different intensity and saturation values.

So the final result will be shown in the next image:

Figure 84. Grosser Keller tiled model.

8.5.4 Annexräume

As it was mentioned before this room also had the problem with the position, and as the workflow is the same, the final result will be shown in the following picture:

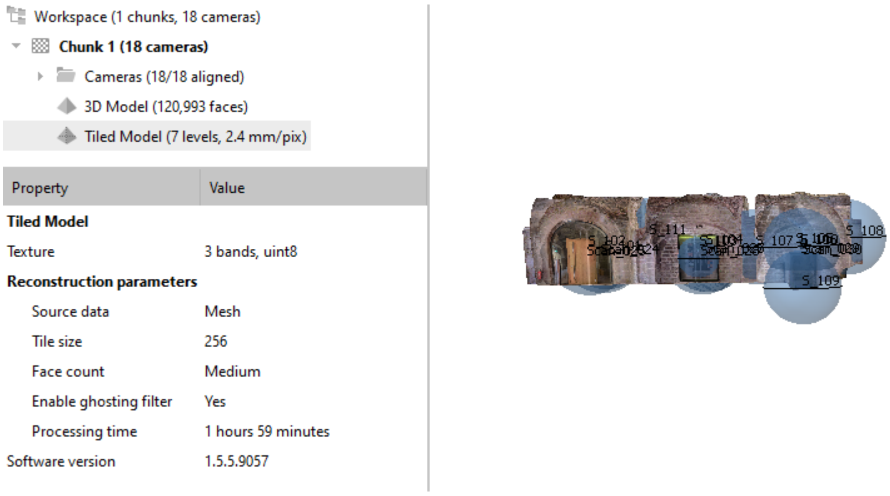


Figure 85. Annexräume tiled model

8.5.5 Frateria

As it was mentioned before this room also had the problem with the position, and as the workflow is the same, the final result will be shown in the following picture:

Figure 86. Frateria tiled model

8.5.6 Ern and Cellarium

Since the Cellarium is the part of the museum of the monastery and has a lot of details, and maximum triangles in the creation of the mesh in SCENE is fixed, it was carrying out some testing in Geomagic Qualify to create a mesh with more quality. Nevertheless, the results of the testing were not better than the mesh obtained with SCENE. Then in the next image is possible to see how looks like the inside of this room:

Figure 87 . Cellarium tiled model.

The mesh has to be edited and some stairs railings were removed since a good shape was very difficult to obtain.

In addition, it has to be mention that in Photoshop some images were edited to remove parts of the mechanism of entrance in the part of the Ern, and also one object place near to the entrance. (*Figure 66 and 67*)

Figure 88. Original image.

Figure 89. Edited image.

As it was mentioned before, as the workflow is the same, the final result will be shown in the following picture:

Figure 90. Ern and Cellarium tiled model.

8.5.7 Kalefraktorium

The principal work in this room was in Photoshop trying to obtain good quality images from a place with no good illumination, but the results of the editing were satisfactory. Then, the final result of this room will be shown in the next image:

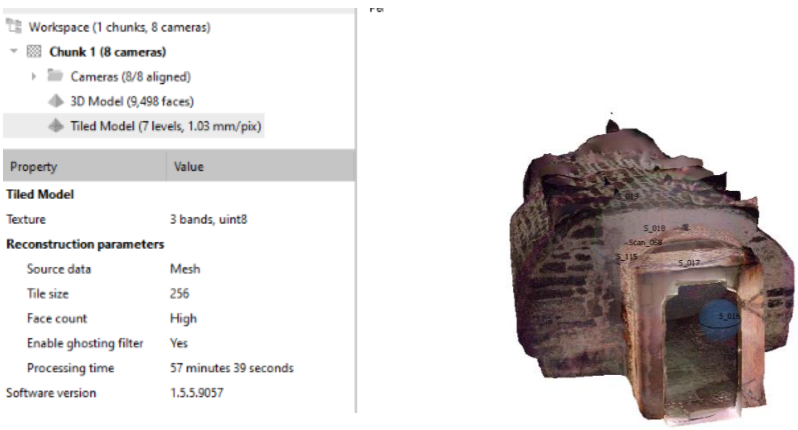


Figure 91. Kalefraktorium tiled model.

8.5.8 AusKreuzgang

As it was mentioned before, as the workflow is the same, the final result will be shown in the following picture:

Figure 92. AusKreuzgang tiled model.

8.5.9 Kapitelsaal

It has to be mentioned that, the main work with this room was in Photoshop becouse some scan markers had to be camouflage.

As it was mentioned before, as the workflow is the same, the final result will be shown in the following picture:

Figure 93. Tiled model Kapitelsaal.

8.5.10 Ostern and Marienkapelle

As it was mentioned before, as the workflow is the same, the final result will be shown in the following picture:

8.5.11 Kreuzgang and Kreuzgangraum

As it was mentioned before this room also had the problem with the position, and as the workflow is the same, the final result will be shown in the following picture:

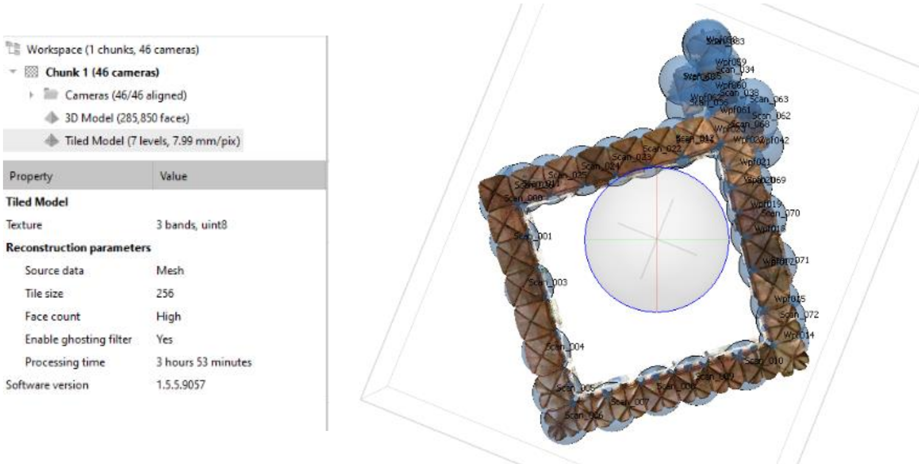


Figure 94. Kreuzgang tiled model.

8.5.12 Brunnenhaus

It has to be mentioned that, the main work with this room was in Photoshop becouse some scan markers had to be camouflage.

As it was mentioned before, as the workflow is the same, the final result will be shown in the following picture:

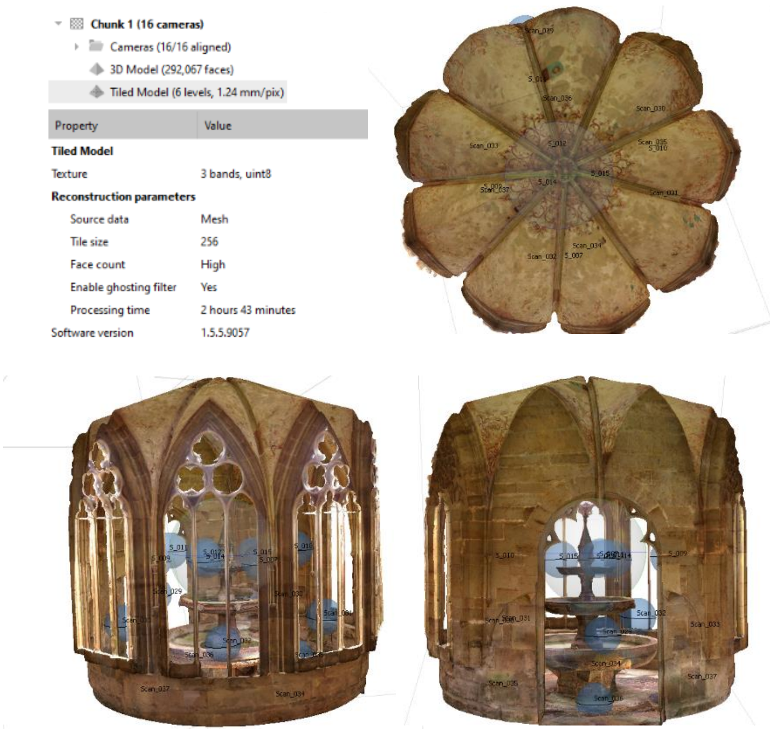


Figure 95. Brunnenhaus tiled model.

9. Practical part 2: Church Project

In this chapter the process for thr project of the church will be explained.

9.1 SCENE

As it was explained before, the images, the positions and the mesh are necessary to obtain. In this case, as there were no new scans in this part, the original Project in SCENE was the only Project that it was used.

9.1.1 Clean the geometry

First of all, it was necessary to remove tome object from the project before to create the mesh. All the objects that were not originally from the monastery were removed manually. Using the 3D view of the project point cloud is it possible to detect which objects has to be removed.

The first step is to detect the object: (*Figure 94*)

Figure 96. Detecting objects church.

Then a clipping box for this object is created in order to deleted only the point that it is wanted. (*Figure 95)*

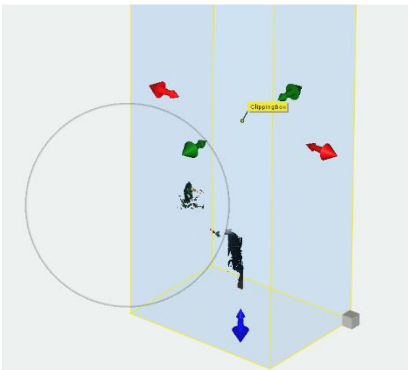


Figure 97. Selecting object church.

Then the selection mode is used, and the object deleted by using the option that appear in the following image:

Figure 98. Removing object church.

There were also some points that didn't belong to an object, but it was necessary to remove them because they were noise points. In the following picture it is possible to see an example:

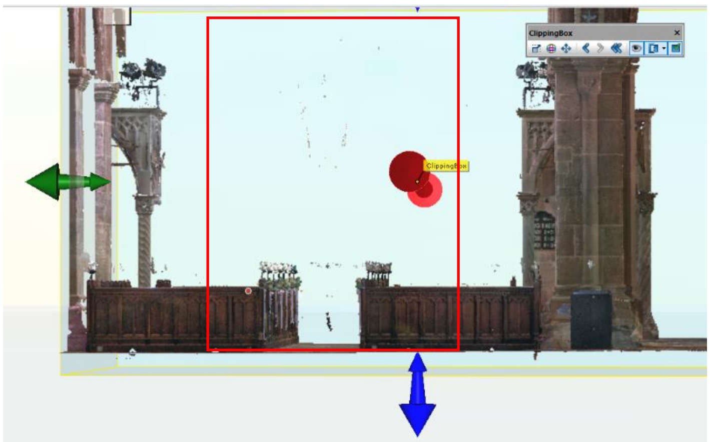


Figure 99. Noise Points.

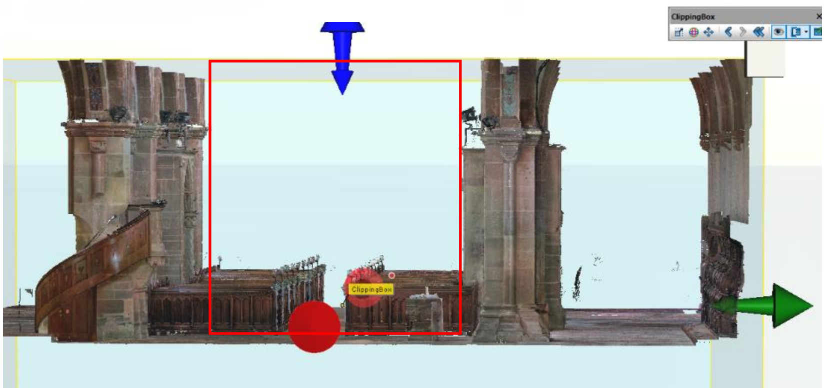


Figure 100. Deleting noise points.

9.1.2 Obtain the geometry, the HDR images and the position

Once the geometry is cleaned, the next step was to create the mesh of the church. Several testing parameters were used, and also the Geomagic software was proposed as software to get the mesh due to, the processing of the creation of a mesh in SCENE has a limit of triangles.

Nevertheless, it was impossible to create the mesh in Geomagic due to the huge amount of points (the software crashed). For that reason, the process to obtain the mesh was done in SCENE and the church was divided in 3 parts. One mesh was created for the front part of the church, another mesh was created for the part of the back side of the church, and then one last mesh was created in the part of the monk's chairs since this part has a lot of details.

This were the parameters for the front part of the church:

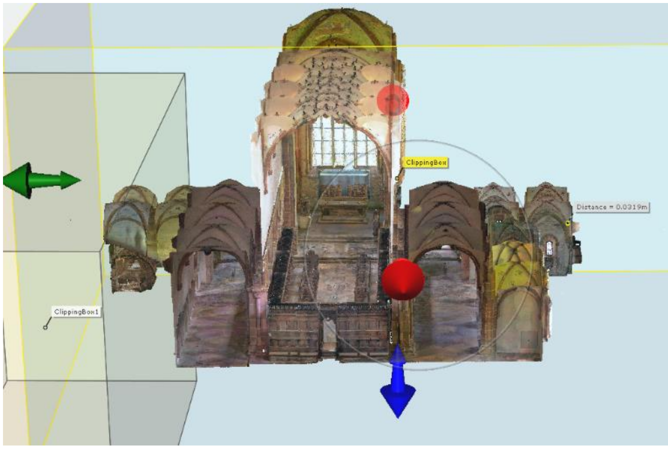


Figure 101. Front part of the church.

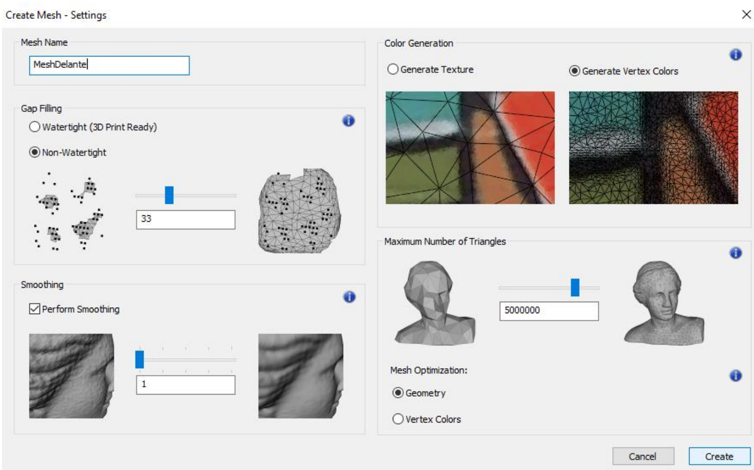


Figure 102. Mesh parameters front part.

This were the parameters for the back part of the church:

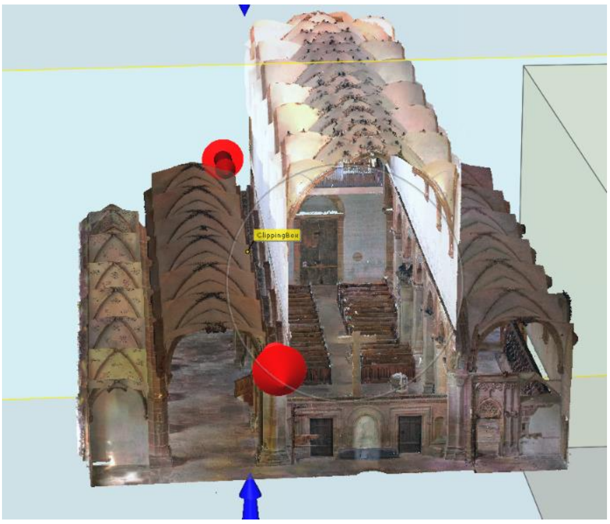


Figure 103. Back part of the church.

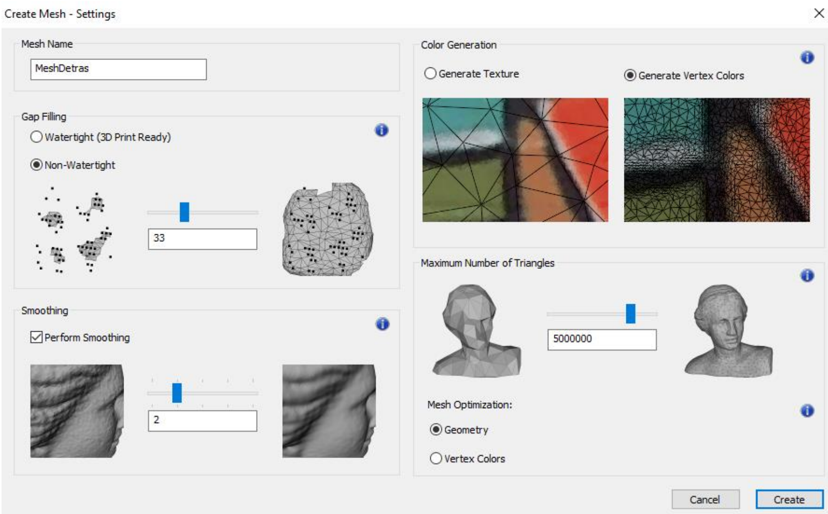


Figure 104. Parameters back part.

This were the parameters for the monk's chairs part of the church:

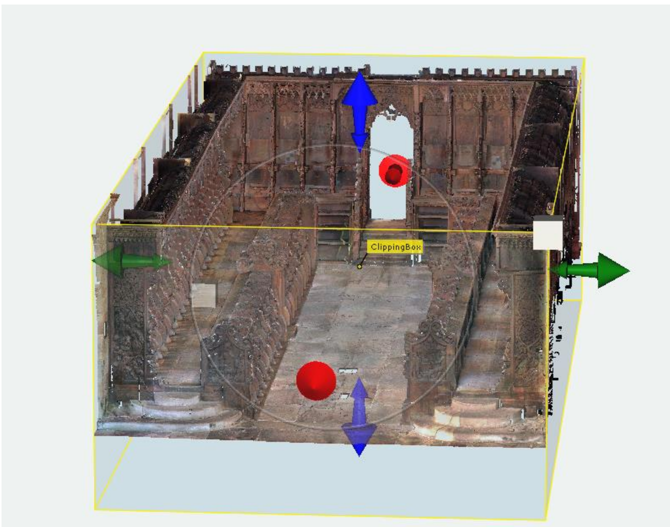


Figure 105. Chairs part of the church.

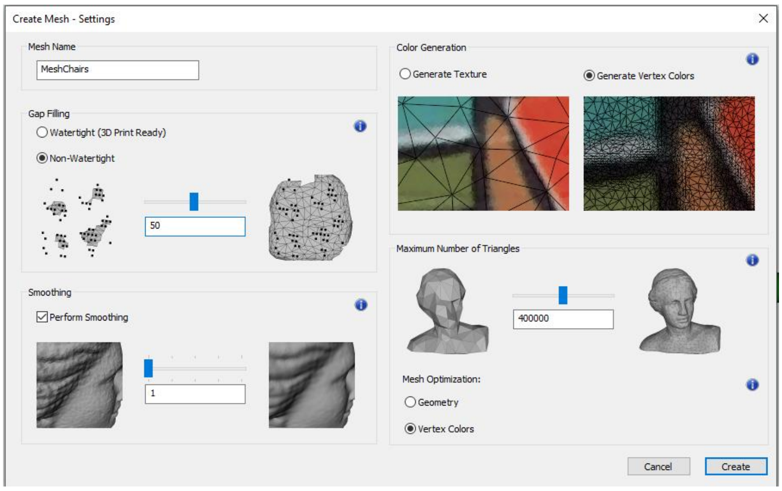


Figure 106. Parameters chairs part.

The position file and the images were obtained also from SCENE.

9.2 Geomagic Qualify

The software Geomagic Qualify was used to join the three meshes of the church, to fill gaps, and to edit the mesh.

The first part that was edited was the mesh of the chairs. In this case the tool "Relax Polygon" was used to smooth a little bit the mesh, and some gaps on the chairs were filled manually.

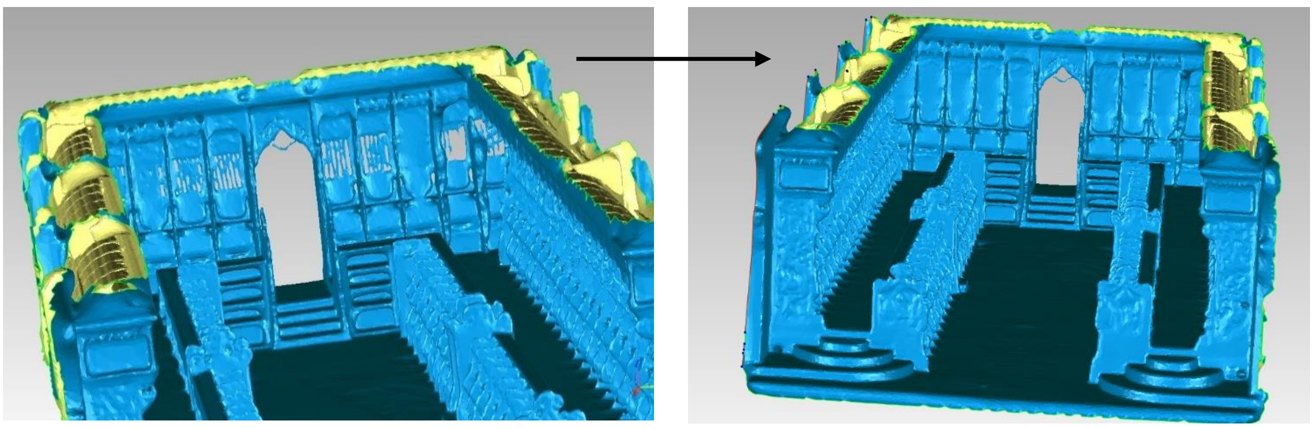


Figure 107. Edition of the chairs mesh.

Additionally, all the Windows from the meshes were edited, due to the shapes was not so flat and when projecting the images to give texture it is better to have a flat shape in this areas.

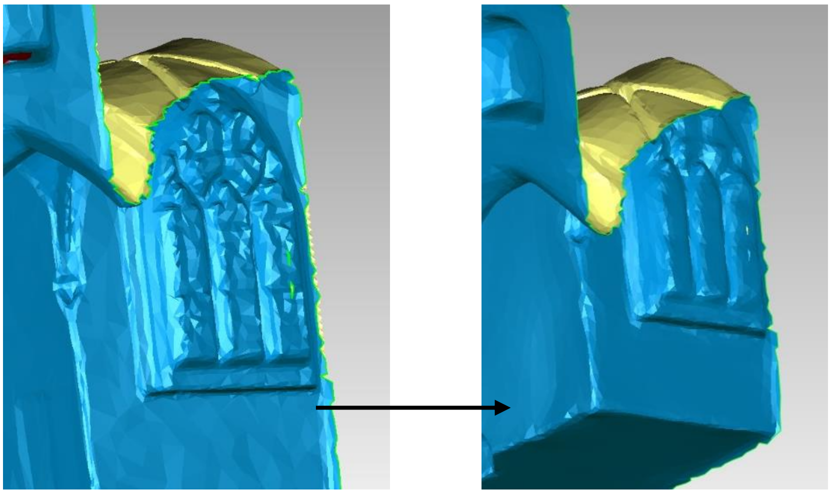


Figure 108. Windows edition.

Once that every mesh was edited, the next step was to join the meshes in one big mesh. The mesh was exported with format ".obj".

9.3 Photoshop

Since the Project point cloud was edited in SCENE to remove the objects, the HDR images has to be edited also to camouflage these objects (Figures X and X). Additionally, some scan marks were removed, and some images were modified to get a better quality regarding the brightness. Finally, all the masks were created.

Figure 109. Original image (right) and Edited image (left.)

Figure 110. Original image (right) and Edited image (left).

9.4 Agisoft Metashape

The final result was created in this software. The inputs were the mesh, the HDR images and the position file. The process to build the tiled model of the church was the most time-consuming since it is the biggest mesh of the project.

Then the result will be shown:

Figure 111. Church tiled model.

Figure 112. Church tiled model.

Figure 113. Church tiled model.

10. Practical part 3: Project in UNITY

10.1 Importing the meshes

As it was explained in previous chapters, the new tiled models have to be imported in the project replacing the old models. From the new tiled models only were used the last level, in some meshes was the level 6 and in others the level 7. The result of a tiled model is also an ".xml" file in which the coordinate of every piece of mesh that are in every level in describe. For that reason, the meshes are georeferenced between them.

In the following image there is a picture about how looks like the structure of this meshes:

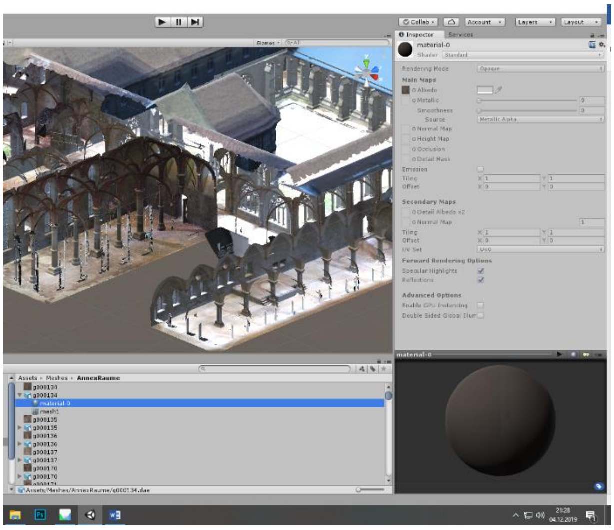


Figure 114. Structure of a tiled model.

The meshes were imported in a new folder inside the "Resources" folder, one folder for one room. As the same time the Xml file was imported in another folder called "Tile". It has to be mentioned that every tiled model has thousand of meshes so it takes between 5-10 minutes to import all of them.

10.2 Creating the colliders

The creation off colliders is very useful since helps the walkability of the virtual reality of the project. In the current project there already exists some colliders in the part of the "Kreuzgang". These colliders are related to the walls of the Kreuzgang. Before it was impossible to enter in some rooms of the monastery because the colliders in these walls didn't have a hole in the door to allow the user to come through. Additionally, there exists also some colliders for the columns of some rooms.

So, for that part the missing colliders for the columns, and the colliders from the church had to be created. In addition, the colliders of the walls of the "Kreuzgang" had to be modified in order to let the "First person" (the avatar) enter to every room of the monastery.

Now, the description about how to create a collider and its properties will be shown. The first step is to create an 3D element. It this case it is wanted to create a collider for a column, so the type of the collider will be "cylinder".

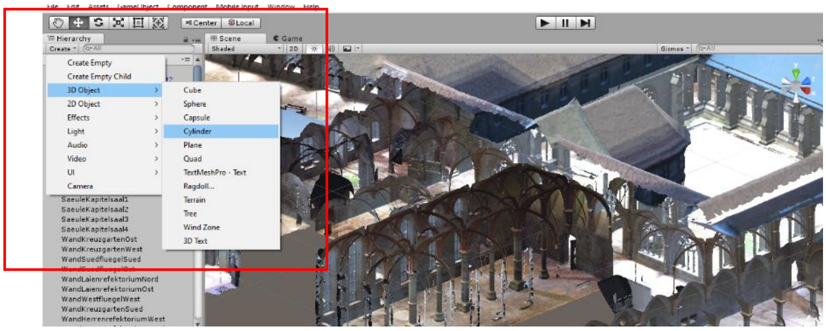


Figure 115. Step to create a cylinder in UNITY.

When the cylinder is created, it is necessary to place it in the position that it is needed.

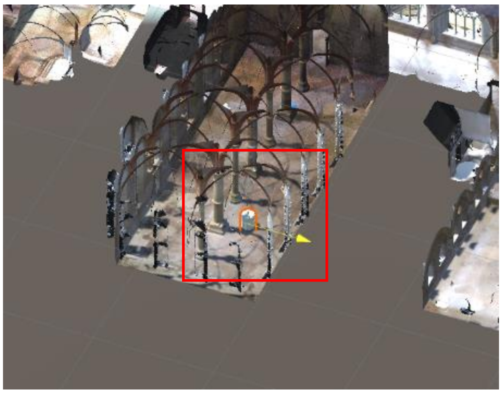


Figure 116. Appearance of a cylinder.

And then the collider had to have these parameters:

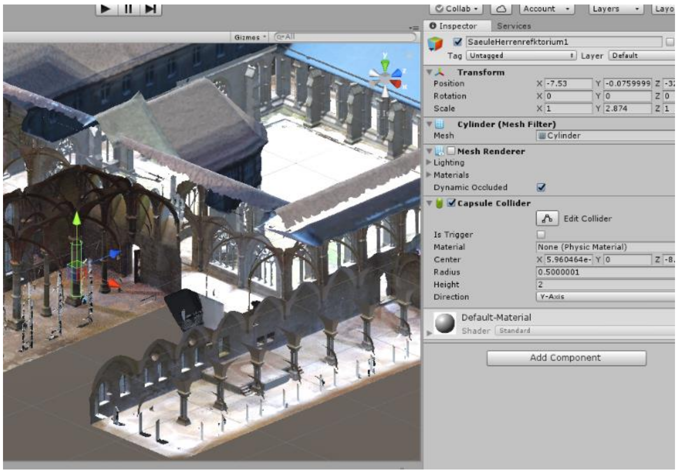


Figure 117. Collider parameters.

This process was repeated for all the parts from the monastery in which the colliders were missing. So that every room has a collider in the walls, in the columns and in the objects that could be inside (as the case of the cellarium).

In the following picture it is possible to see all the colliders of the project:

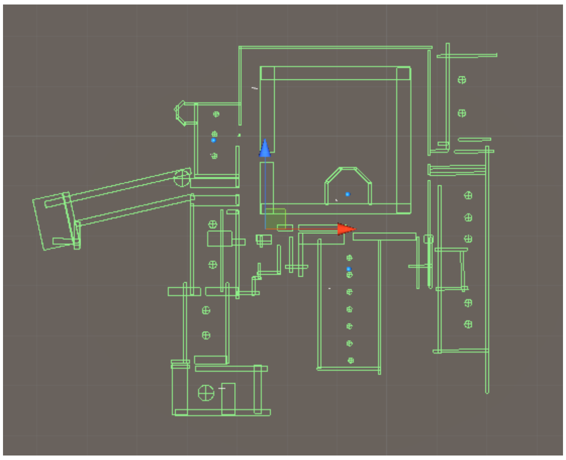


Figure 118. All the colliders in the Project.

10.3 Adding the script to load the tiled models

Since every room has its own tiled model, and every tiled model has thousands of little meshes, it is necessary to load the meshes in run-time. That means that only the meshes are imported when the script is running, when exit from the “game mode” the meshes are not loaded any more. So, in that way the project in not so memory consuming and works faster. Only the last level of every tiled model was load. Some meshes has 6 level and others 7.

To be able to execute the script, an empty parent object has to be created. Inside this pared object an empty child object has to be created. In this empty child object, the thousands of meshes from the last level of one tiled model will be imported. Therefore, for every tiled model it is necessary an empty parent object.

The parent object and the child object have to have a specific position, due to exist and offset of coordinates between the existing object from the original project, and the new tiled model. (Since the tiled model has a local system of coordinates and coordinate system of the objects from the original project were forced to (0,0,0)).

These are the positions:

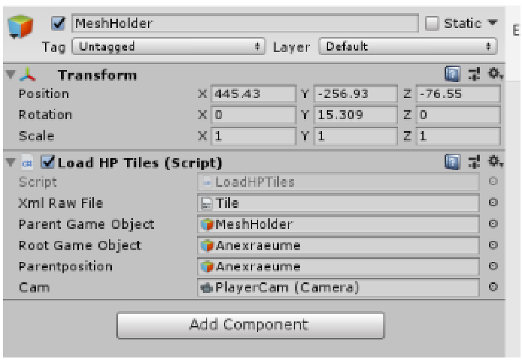


Figure 119. Position for the parent object.

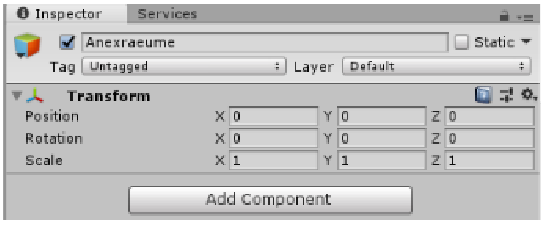


Figure 120. Position for the child object.

Three scripts are used to load the meshes in run-time, but only one in linked to an object. The one who is linked is called “LoadHPTiles.cs”, and it is linked to the parent object (Figure 115). The other scripts are called “ModelTile.cs” and “TileAtributtes.cs”. All scripts are in C#.

As it can be seen the script has seven parameters that has to be filled:

- The xml file: This xml file is obtained when exporting the tiled model from Agisoft Metashape. It has the coordinates for every little mesh from every level of the tiled model. So, it is needed to georeference the meshes of the last level.
- Parent game object: is the empty parent object.
- Root game object: is the child object.
- Parent position: is the position from the child game object.
- Cam: is the camera from the player.
- Level: is the level that every mesh has. (In that case it could be six or seven)
- Folder: is the folder where the last level is imported. It has to be inside a folder called “Resources”

When the script is executed, the meshes that are inside the angle of the view of the camera are loaded into the scene (This is called Frustum culling). The script generated a box collider for every mesh in order to know is they are inside of the view of the camera. In the next picture it is possible to see an example (Figure 119). The player is located in front of the door that connect Grosser Keller with Annexräume. As you can see the green boxes are the box colliders, and as the player is looking into the Annexräume, the meshes from that part that are in the angle of the view of the camera are loaded also.

Figure 121. Running the script example.

In the pre-processing of the Occlusion culling, the objects (in which has to be applied the occlusion), have to be baking. Then for that reason, the occlusion culling was not possible to apply since the tiled models are loaded in run time.

10.4 Teleportation script

Since some rooms has the door close, and the script create a collider, the player could not go through different rooms. To fix this problem, another script was applied. Two different cube objects have to be created and placed in both sides of the door. The script works as the following way, when the player collide with this cube is teleported to the other side of the door.

In the following picture it can be seen the cube:

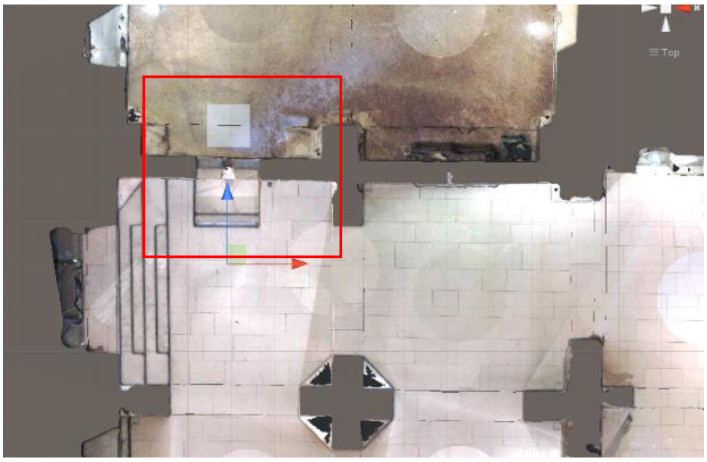


Figure 122. Example of a teleport cube.

In order to know exactly where it has to be placed the cube, since the tiled model are loaded in run time, the old model of the monastery was used. In the case of the church was different because the church was not yet implemented into the old model. So for that case, the script was executed, the tiled models loaded and the cubes were created. The positions of the cubes were taken, because when exiting from the "game mode" everything created inside this mode is removed. So the cubes were created again but with the coordinates that were taken into the "game mode".

10.5 Creation of the new terrains

In the existing project only one terrain was created, but since the rooms are in different levels (*Figure 121*), more terrains were needed. The terrain does not have to be placed at the bottom

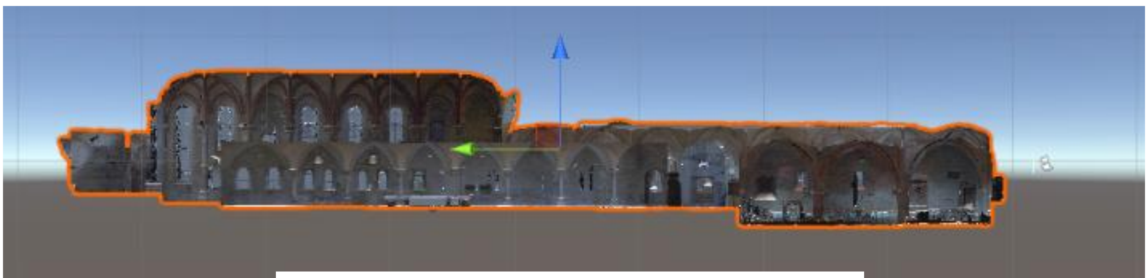


Figure 123. Example of the different levels of the rooms.

of every room, because the player has a collider in its head and this collider is the one which interact with the terrain, so the terrain has to be defined a little above the ground.

10.6 Improving the texture

The method to improve the texture that has been explained in the chapter 7.5.3 was implemented into the script that is used to load the tiled models. It was necessary to create a loop to check how many materials has every mesh.

In addition, another action was carried out to improve the texture. The texture type from the images was changed from “Standard” to “Sprite (20 and UI)” manually. This process takes a lot of minutes because it has to change the texture type of thousands images.

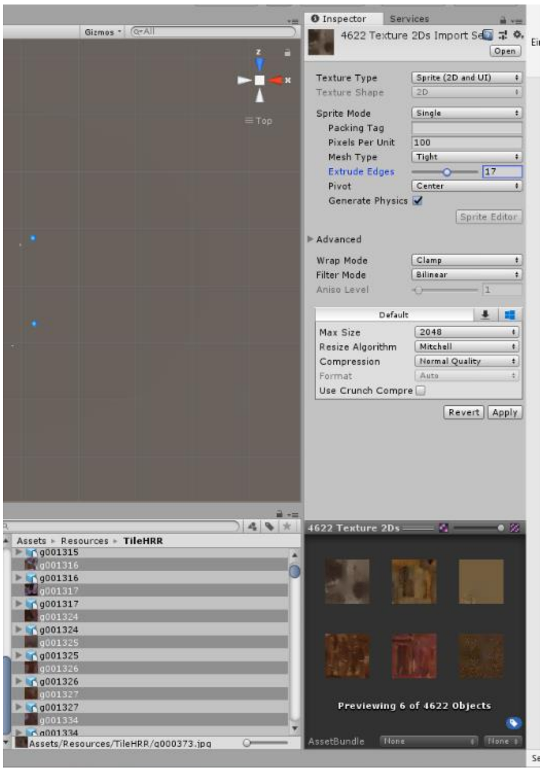


Figure 124. Improving the texture parameters.

10.7 Final result

In this chapter the final result and the old models from the old UNITY project will be compared. For that some images form the game view will be shown:

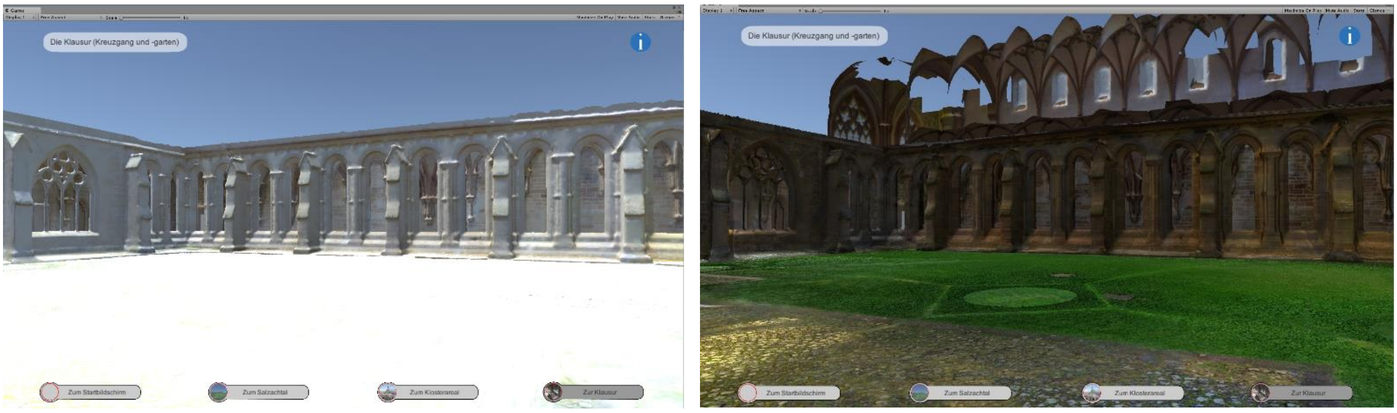


Figure 125. Innenhof difference between old model (left)and new model (right))

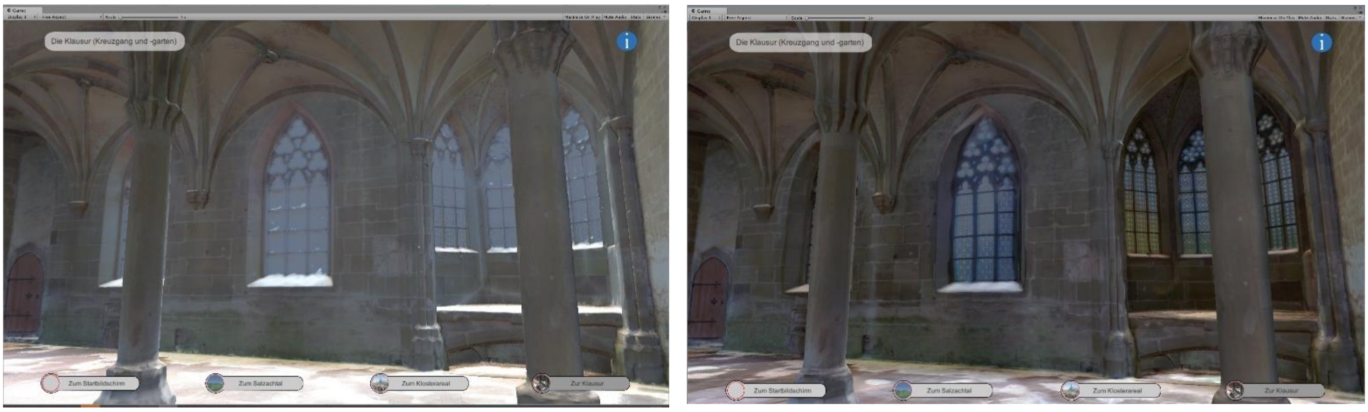


Figure 126. Kapitelsaal difference between old model (left)and new model (right))

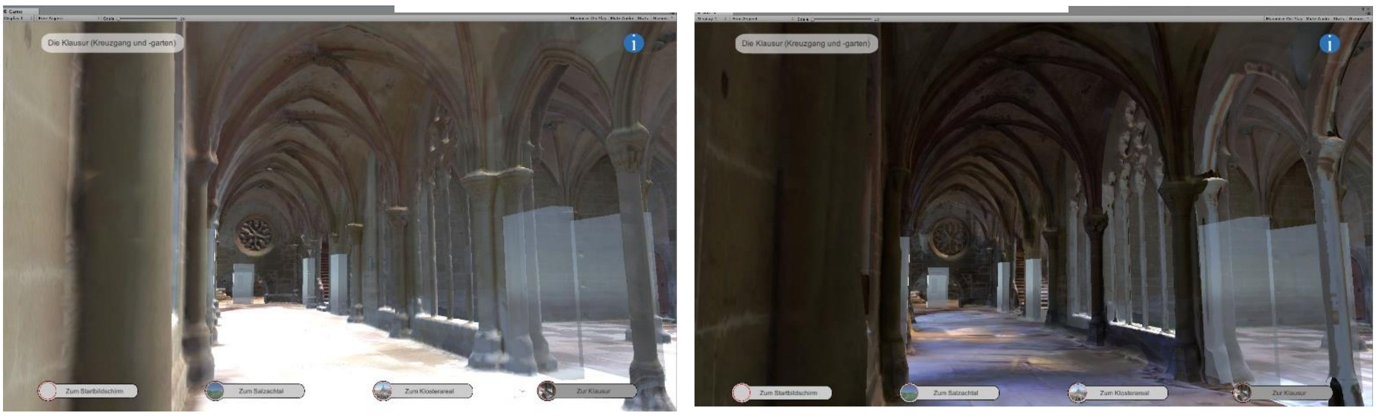


Figure 127. Kreuzgang difference between old model (left)and new model (right))

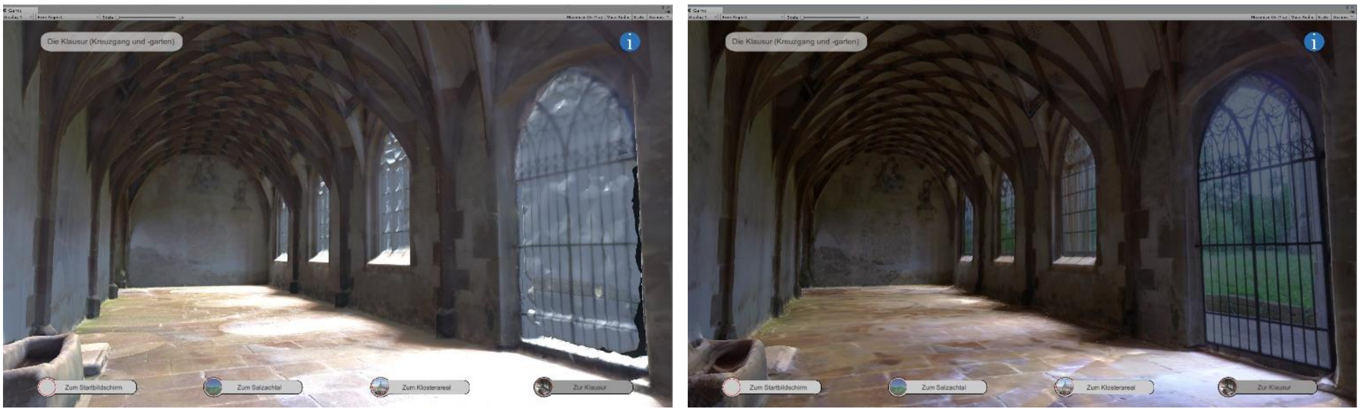


Figure 128. Marienkapelle difference between old model (left)and new model (right))

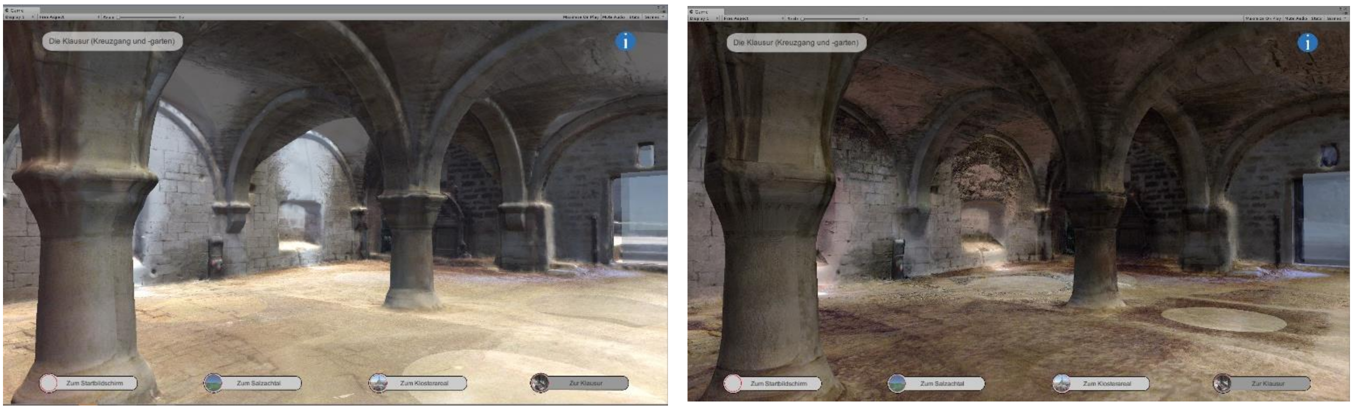


Figure 129 . Grosser Keller difference between old model (left)and new model (right))

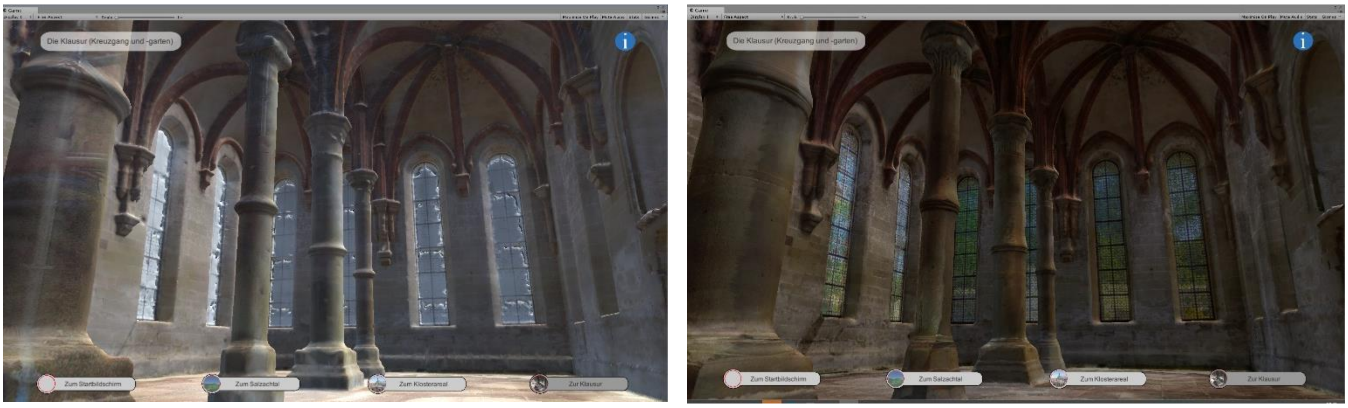


Figure 130. Herrenprefektorium difference between old model (left)and new model (right))

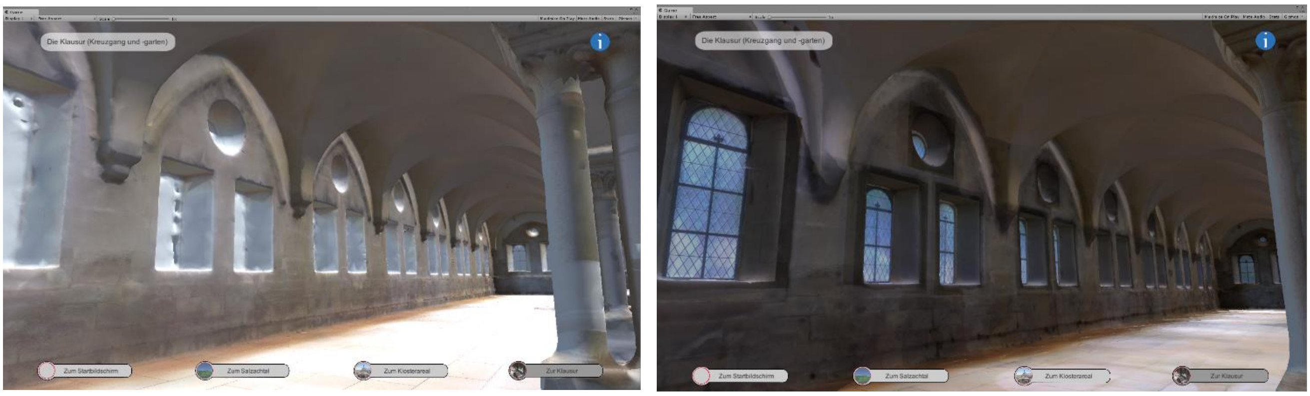


Figure 131. Laienrefrektorium difference between old model (left)and new model (right))

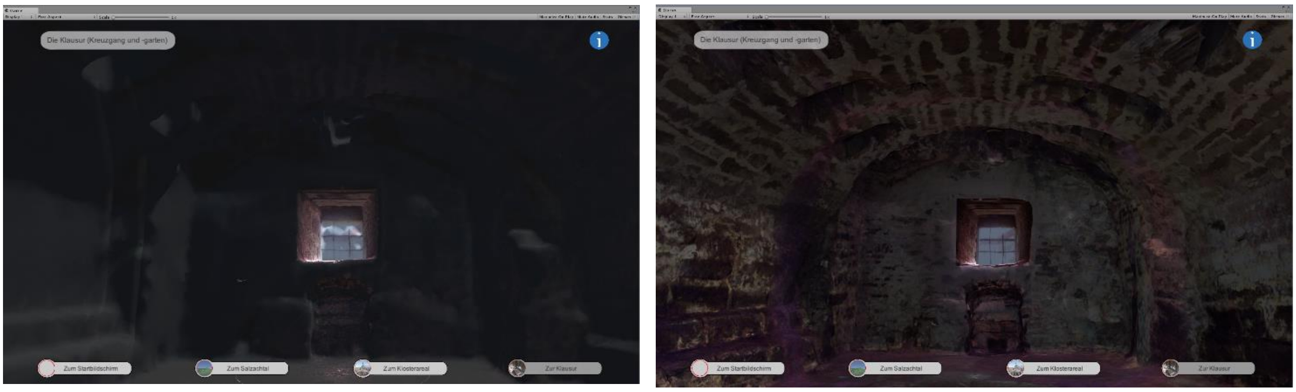


Figure 132. Kalefaktorium difference between old model (left)and new model (right))

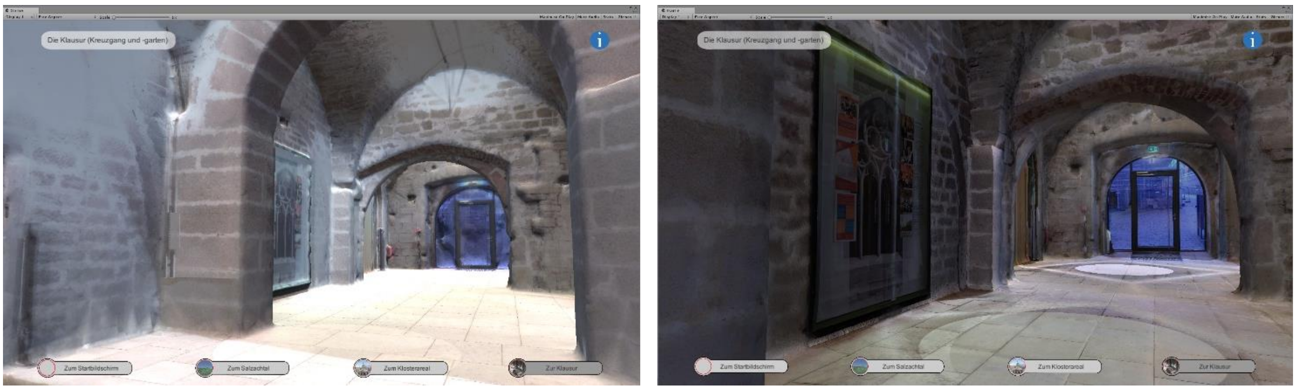


Figure 133. Annexräume difference between old model (left)and new model (right))

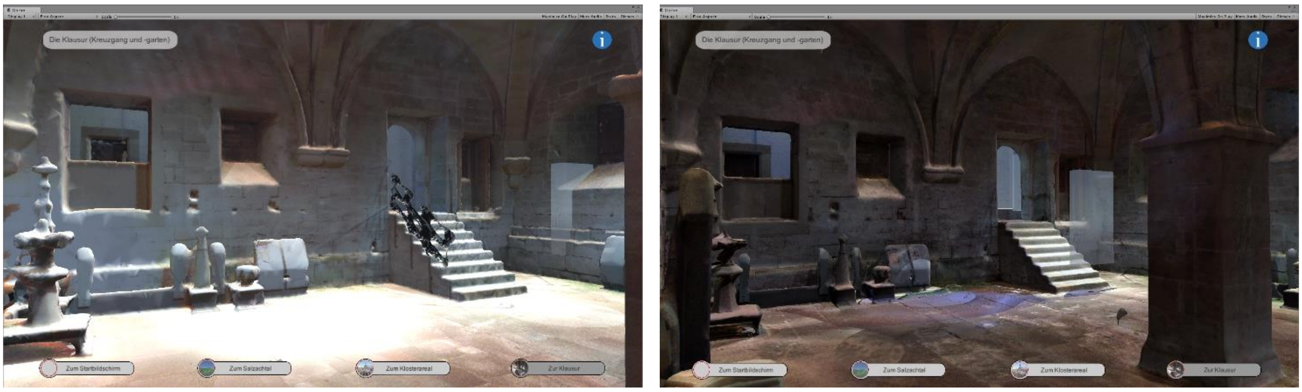


Figure 134 Cellarium difference between old model (left)and new model (right))

As it could be seen in the previous images, the texture of all rooms has been improved. The most notable parts are the windows and the walls or floors with too much light. The white boxes that appear in some pictures are the portal that connect the different rooms.

11. Conclusions and Future works

This Project has covered different states, from the analysis of the original data, testing, location approaches and measurements with the scanner, data processing, creation of the final results, to the incorporation into an existing Project. Through all these steps different disciplines acquired throughout the master and degree have been used and have been implemented in a real Project.

Therefore, it could be said that this Master Thesis is a fairly complex and complete work due to the different parts and disciplines it covers.

Thanks to new technologies and Geomatics, a Project to visualize a historical element can be approached to the user in an easy and direct way, as this project has demonstrated.

The main objective in general was to improve the texture. Considering that the monastery of Maulbronn has many rooms with a lot of windows, the problem of luminosity became clear. But since the use of the laser scanner with the maximum level in HDR images activated (level 5x), this problem decreased. Although it was tried to take measures at times when sunlight did not affect the windows, it is normal that some little reflections or areas with some lighting still appear. However, most of these reflections or areas with a bit of luminosity were not a problem since the images were edited in Photoshop to be able to have a better quality. Therefore, the part of masking the images is very important, because is in that part when the worst part of every image has to be mask out. That step is needed to warrantee good results. Although the process of the masking was good, some circles appear in the floor due to the difference between the intensity of the images. However, the texture in general was improved with satisfactory results.

Regarding the memory-consuming in UNITY, the used of the tiled model increase the quality of the visualization, since only the meshes that are close to the player are loaded with high definition. Of course, the use of the backface culling and frustum culling make the project run faster and not to be smoothly when the player moves.

As future views, some performance improvements in UNITY can be included. As for example, in the museum there are panels with information on the pieces and architectural elements found within the museum itself. These panels cannot be read correctly in the 3D model. An incorporation that could be very useful would be that when the user approached the panel, a pop up appeared with the panel information or with additional information about that particular piece of the museum.

Another possibility to improve the appearance and texture of the 3D model could be the incorporation of external pictures to the laser scanner. Such images could be used to obtain more details of small objects, paintings on the walls and ceilings of some rooms.

12. Bibliography

Samedov/Texturizing-FARO-SCENE-model-in-Agisoft-Metashape: Re-projecting the texture of the 3D meshes created in FARO SCENE in Agisoft Metashape. (n.d.). Retrieved July 4, 2019.

URL: https://github.com/Agageldi/Texturizing-FARO-SCENE-model-in-Agisoft-Metashape

Agisoft LLC. (n.d.). *Agisoft PhotoScan User Manual*.

URL: https://www.agisoft.com/pdf/photoscan-pro_1_4_en.pdf

Blackman, S. (2013). Beginning 3D Game Development with Unity 4: *Beginning 3D Game Development with Unity 4.*

URL: https://doi.org/10.1007/978-1-4302-4900-9

Cinco cosas buenas y cinco no tan buenas de Unity 3D – La leyenda de Darwan. (n.d.). Retrieved December 16, 2019.

URL: https://laleyendadedarwan.es/2017/12/21/cinco-cosas-buenas-y-cinco-no-tan-buenas-de-unity-3d/

El escaneado láser revoluciona el sector de la arquitectura y la construcción | FARO SPAIN, S.L.U. (n.d.). Retrieved December 16, 2019.

URL: https://www.faro.com/es-es/case_study/el-escaneado-laser-revoluciona-el-sector-de-la-arquitectura-y-la-construccion/

García-Gómez, I., Fernández de Gorostiza, M., & Mesanza Moraza, A. (2012). Láser escáner y nubes de puntos. Un horizonte aplicado al análisis arqueológico de edificios. *Arqueología de La Arquitectura*.

URL: https://doi.org/10.3989/arqarqt.2011.10019

Lammers, K. (2013). *Unity_Shaders_and_Effects_Cookbook_2013*. Retrieved from papers2://publication/uuid/C438A3C9-7D25-4027-BC0A-E5B20544770A

Laser Escaner 3D - Tecnología Bim. Retrieved December 16, 2019.

URL: http://www.tecnologiabim.es/laser-escaner-3d/

Manual de Faro Scene (2019). *Scene 2019.*

URL:https://knowledge.faro.com/Software/FARO_SCENE/SCENE/User_Manual_for_SCENE

Muñoz, M. J. G., Ruiz, A. J. R., Sánchez, R. J. S., Anguita, C. J. O., Esteban, A., & Lara, J. (2010). Uso de sistemas basados en escáner 3D para digitalización y estudio del patrimonio arqueológico. Virtual Archaeology Review.

Monasterio de Maulbronn - Wikipedia, la enciclopedia libre. Retrieved December 15, 2019.

URL: https://es.wikipedia.org/wiki/Monasterio_de_Maulbronn

SCENE FARO SPAIN, S.L.U. Retrieved December 16, 2019.

URL: https://www.faro.com/es-es/productos/3d-design/faro-scene/

Simon Baumer. (2019). *Hochschule Karlsruhe – Technik und Wirtschaft University of Applied Sciences Simon Bäumer Erzeugung großer Punktwolken aus TLS und Visualisierung im Internet am Beispiel des Klosters Maulbronn Erzeugung großer Punktwolken aus TLS und Visualisierung im Int*.

Tutorial 5 : Un cubo con textura. (n.d.). Retrieved December 15, 2019.

URL: http://www.opengl-tutorial.org/es/beginners-tutorials/tutorial-5-a-textured-cube/

Unity - Manual: Occlusion Culling. Retrieved September 5, 2019.

URL: https://docs.unity3d.com/Manual/OcclusionCulling.html

Winkler, F. (2013). Introduction to Unity3D (vers. 4.2). *Computer Games*, 1–13.

URL:https://cla.purdue.edu/academic/rueffschool/ad/etb/resources/AD41700_Unity3D_workshop01_F13.pdf

¿Qué es el Láser escáner terrestre?. Retrieved December 16, 2019.

URL: http://detopografia.blogspot.com/2012/11/que-es-el-laser-escaner-terrestre.html

13. Annexes

The script that was used in Photoscan is the following:

```
import PhotoScan, math, sys

#path to the position.txt which was exported from Faro Scene
#path = 'C:/Users/saag1011/Desktop/Test_Scans/Positions.txt'
path = sys.argv[1]
path = path.replace('\\','/')

doc = PhotoScan.app.document
chunk = doc.chunk
cams = chunk.cameras    #list of the cameras

#opening the file
file = open(path, 'r')

print("Reading the file '"+ path +"'...")
txt = file.read()
txt = txt.splitlines()

#internal variables
camNum = len(cams)
rowNum = len(txt)
coordinates = {}
angles = {}
I= PhotoScan.Matrix([[1,0,0],[0,1,0],[0,0,1]])

#find line of scan coordinates
i=0
for row in txt:
    i+=1
    if row.find('Scans {')>=0:
        break

#extract scan coordinates from the lines into dictionaries 'coordinates' and 'angles'
print("Reading coordinates from the file...")
while(i<rowNum):
    row = txt[i]
    if row.find('}')>=0:
        break
    row = row.replace('"','')
    row = row.replace('  ',' ')
    row = row.replace('  ',' ')
    row = row.split(' ')

    coordinates[row[0]] = [float(row[1]),float(row[2]),float(row[3])]
    angles[row[0]] = [float(row[4]),float(row[5]),float(row[6]),float(row[7])]

    i+=1

#creates rotation matix from list of angles
def RotEuler(Rot):
    a,b,c=Rot[0],Rot[1],Rot[2]

    Rx = PhotoScan.Matrix([[1,0,0],[0,math.cos(math.radians(a)),-math.sin(math.radians(a))],[0,math.sin(math.radians(a)),math.cos(math.radians(a))]])
    Ry = PhotoScan.Matrix([[math.cos(math.radians(b)),0,math.sin(math.radians(b))],[0,1,0],[-math.sin(math.radians(b)),0,math.cos(math.radians(b))]])
    Rz = PhotoScan.Matrix([[math.cos(math.radians(c)),-math.sin(math.radians(c)),0],[math.sin(math.radians(c)),math.cos(math.radians(c)),0],[0,0,1]])

    return Rx,Ry,Rz

# offset rotation matrix
Rx1,Ry1,Rz1 = RotEuler([-90,0,90])

#Tranformation of the cameras
i=0
while i<camNum:
    #reading/finding coordinates and orientations from the dictionary
    cam0 = coordinates[cams[i].label]
    rot  = angles[cams[i].label]
```

```python
    #creating rotation matrix from axis rotation angles
    n=[rot[2],rot[0],rot[1]]
    J=PhotoScan.Matrix([[0, -n[2],n[1]],[n[2],0,-n[0]],[-n[1],n[0],0]])          #J or n*J in belove reference
    R=I+J*math.sin(math.radians(rot[3]))+J*J*(1-math.cos(math.radians(rot[3])))      #Rotation matrix ref:https://de.mathworks.com/help/physmod/sm/mech/gs/repr
    #backward calculation of Eular angles out of Rotation matrix which was created from axis angle
    EAngle = PhotoScan.utils.mat2opk(R)
    EAngle = [EAngle[1],EAngle[2],EAngle[0]]
    Rx,Ry,Rz = RotEuler(EAngle)
    #Create new Rotation matrix by defining rotation orders
    R=Rz*Rx*Ry
    R*=Rz1*Rx1  # offset angles
    #Transformation matrix ref:master thesis p.36
    # 4x4 location/coordinates reference matrix  ref:master thesis p.36
    T = PhotoScan.Matrix([[R.row(0)[0],R.row(0)[1],R.row(0)[2],cam0[0]],[R.row(1)[0],R.row(1)[1],R.row(1)[2],cam0[1]],[R.row(2)[0],R.row(2)[1],R.row(2)[2],cam
    #changing Transformation of the camera
    cams[i].transform = T
    print("Transformation for the camera '"+cams[i].label+"' was set")
    i+=1

print("Camera transformations are successfully set!")

#Masking
if(cams[0].mask is None):
    print('No mask was found on the frist camera!')
# else:
    # i=1
    # myMask = cams[0].mask
    # while(i<camNum):
        # cams[i].mask = myMask
        # i+=1
    # print('Mask of the first camera was inserted to the all cameras!')

print("Script successfully finished!")
```

The Script used in UNITY for the teleport is:

```csharp
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System;

public class ModelTiles {

    public string name;
    public Vector3 center;
    public string parent;
    public float radius;
    public int level;

    public ModelTiles(string newName, Vector3 newCenter, string newParent, float newRadius, int newLevel){
        name = newName;
        center = newCenter;
        parent = newParent;
        radius = newRadius;
        level = newLevel;
    }


}
```

The scripts used in UNITY to load the tiled models are:

TileAttributes.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;


public class TileAttributes : MonoBehaviour {

    //public ModelTiles Attributes;

    public string tileParent;
    public int tileLevel;
    public string tileName;
    public float tileRadius;
    public Vector3 tileCenter;


    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {

    }
}
```

ModelTiles.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System;

public class ModelTiles {

    public string name;
    public Vector3 center;
    public string parent;
    public float radius;
    public int level;

    public ModelTiles(string newName, Vector3 newCenter, string newParent, float newRadius, int newLevel){
        name = newName;
        center = newCenter;
        parent = newParent;
        radius = newRadius;
        level = newLevel;
    }


}
```

LoadHTPTiles.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System.IO;
using System.Xml;
using UnityEngine.UI;


public class LoadHPTiles : MonoBehaviour {

    // XML Datei als Variable
    public TextAsset xmlRawFile;
    public GameObject parentGameObject;
    public GameObject rootGameObject;
    public GameObject parentposition;
    //public GameObject Abstandsholder;
    public Camera cam;
    public int Levels;
    public string tileSubFolder;
    Plane[] planes;
    Collider objCollider;
    Shader shader;
    Renderer rend;


    //Material material;
    //  private ScriptableObject  instAttr;
    // Use this for initialization
    void Start () {

        string data = xmlRawFile.text;
        parseXmlFile (data);

    void Update(){


        setVisibility_Tile (cam, rootGameObject);
        //print (parentGameObject.GetComponent<TileAttributes> ().tileName);

    }


    void setVisibility_Tile (Camera camVision, GameObject rootObject)
    {

        // AnfangsNode überspringen
        //if (rootObject.name == "root(Clone)") {
        //  rootObject = rootObject.transform.GetChild (0).gameObject;
        //}


        // Anzahl der Kinder
        int numChilds = rootObject.transform.childCount;


        // Jedes Kind
        for (int i = 0; i < numChilds; i++) {

            // Kamerakegel berechnen
            planes = GeometryUtility.CalculateFrustumPlanes (cam);

            // Abstand berechnen
            float dist = Vector3.Distance (rootObject.transform.GetChild (i).transform.position, camVision.transform.position)
```

```
if (rootObject.transform.GetChild (i).GetComponent<BoxCollider> () == null) {

    //Reichweite überprüfen rootObject.transform.GetChild (i).GetComponent<TileAttributes> ().tileRadius
    if (100 > dist)
        objCollider = rootObject.transform.GetChild (i).gameObject.AddComponent<BoxCollider> ();

    else
        // Boxcollider löschen falls ausserhal der Reichweite
        Destroy (rootObject.transform.GetChild (i).GetComponent<BoxCollider> ());
}

// Wenn innerhalb der Reichweite:
if (100 > dist) {

    // Mutter ausschalten
    //rootObject.GetComponent<MeshRenderer> ().enabled = false;

    // CHecken ob im Kegel
    if (GeometryUtility.TestPlanesAABB (planes, rootObject.transform.GetChild (i).GetComponent<BoxCollider> ().bounds)) {

        // Wenn ja, einschalten
        rootObject.transform.GetChild (i).GetComponent<MeshRenderer> ().enabled = true;


    } else {

        //Ausschalten wenn nicht im Kegel
        rootObject.transform.GetChild (i).GetComponent<MeshRenderer> ().enabled = false;

    }

    } else {

        // Auschalten wenn zu weit
        //
        rootObject.transform.GetChild (i).GetComponent<MeshRenderer> ().enabled = false;
        // Mutter einschalten
        //rootObject.GetComponent<MeshRenderer> ().enabled = true;
    }


}
}

// Update is called once per frame
void parseXmlFile (string xmlData) {

    // Liste wird erstellt
    List<ModelTiles> tileList = new List<ModelTiles> ();

    // XML Doc wird geöffnet
    XmlDocument xmlDoc = new XmlDocument ();
    xmlDoc.Load (new StringReader(xmlData));
    float count = 0;
    string parentName = "";
    int Level = 0;


    // für jedes Node Objekt:
    foreach (XmlNode node in xmlDoc.SelectNodes("//Node")) {

        //Zähler für die anzahl der Knoten
        count++;

        // Radius wird abgefragt
        float radius = float.Parse(node ["Radius"].InnerText);

        // Modelname wird gesucht
        XmlNode ModPathNode = node.SelectSingleNode ("ModelPath");

        //POsition wird gesucht
        XmlNode center = node["Center"];

        string model = "";

        //wenn ein Model existiert:
        if (ModPathNode != null) {
```

```
model = node ["ModelPath"].InnerText;

//Modelcentrum wird abgefragt
float x = float.Parse (center.Attributes.GetNamedItem ("x").Value);
float y = float.Parse (center.Attributes.GetNamedItem ("y").Value);
float z = float.Parse (center.Attributes.GetNamedItem ("z").Value);


//Wenn es sich nicht um den Hauptknoten (erster Knoten, gehört nicht zu Modelbaum) handelt:
if (count > 1) {

    // Mutterknoten wir ermittelt
    if ( node.ParentNode.SelectSingleNode ("ModelPath") != null)
        parentName = node.ParentNode.SelectSingleNode ("ModelPath").InnerXml;
    else parentName = node.ParentNode.ParentNode.SelectSingleNode ("ModelPath").InnerXml;

} else {
    parentName = "root.dae";
}

//Endung wird entfern vom Namen
parentName = parentName.Remove (parentName.Length - 4);



// Ladepfad wir zusammengesetzt
string loadModel = tileSubFolder + model;
loadModel = loadModel.Remove (loadModel.Length - 4);

// Leeres Ladepfad wir ausgeschlossen
if (loadModel != "T") {

    // Name des Muuterobjektes wird angepasst
    string parentGameObjectName = parentName + "(Clone)";

  //Anfangsbuchstabe ermittelt
  char Anfangsbuchstabe = model[0];
  Level = char.ToUpper (Anfangsbuchstabe) - 64;



  // Elemet in zusätzliche Liste geschriben zur optionalen verwendung)

  tileList.Add (new ModelTiles (model, new Vector3 (x, y, z),parentName,radius,Level));


  //nur Level 7 wird geladen
  if (Level == Levels) {

      GameObject instance = Instantiate (Resources.Load (loadModel, typeof(GameObject))) as GameObject;
      //... auf position gestetzt,....
      instance.transform.parent = parentposition.transform;
      instance.transform.localPosition = new Vector3 (-x, z, -y);
      instance.transform.Rotate (0f, 0f, 15.309f);
      //... als Kindobjekt des Mutterobjektes geschoben,....
      //instance.transform.parent = rootGameObject.transform;

      shader = Shader.Find ("Diffuse");
      Material[] tileMaterials;

      //tileMaterials =
      int range = instance.GetComponent<Renderer>().materials.Length;

      for (int m = 0; m < range; m++) {
          instance.GetComponent<Renderer>().materials[m].shader = shader;
          Shader.WarmupAllShaders ();
      }
      instance.GetComponent<MeshRenderer>().enabled = false;

      //scghliesslich wird das Script für die Attribute an das Objekt gehängt
      instance.AddComponent<TileAttributes> ();


      //Objekt wird auf dem Levelzugehörigem Layer gesetzt
      instance.layer = Level + 8;

      //Attribute werden gesetzt
      instance.GetComponent<TileAttributes> ().tileParent = parentGameObjectName;
      instance.GetComponent<TileAttributes> ().tileLevel = Level;
      instance.GetComponent<TileAttributes> ().tileName = loadModel;
      instance.GetComponent<TileAttributes> ().tileRadius = radius;

    }
  }

        }

      }

    }
  }
```