

AALBORG UNIVERSITY COPENHAGEN

MSc MEDIALOGY

2ND SEMESTER

Mesh Reconstruction Using The Point Cloud Library

Authors:

Andrea KEISER

Ann-Marie H. B. BECH

Frederik V. BÆRENTSEN

Supervisor:

Georgios TRIANTAFYLLIDIS

28 May 2015

This page was intentionally left blank.

Contents

Abbreviations	v
1 Introduction	1
2 Initial Problem Statement	3
3 Investigation	5
3.1 Point Cloud Library	5
3.2 Autodesk ReCap	6
3.3 VisualSFM	6
3.4 Meshlab	8
3.5 C3 Technologies	8
3.6 Drone 55	8
3.7 Octofilms	9
3.8 Reconstruction projects	9
4 Final Problem Statement	11
5 Methods	13
5.1 Point Cloud Data	13
5.2 Structure from Motion	14
5.2.1 The Structure from Motion Task	15
5.2.2 Greedy Triangulation	15
5.2.3 Grid Projection	15
5.2.4 Poisson	16
5.2.5 Delaunay Triangulation	17
5.2.6 Success Criteria	17
6 Implementation	19
6.1 Image Acquisition	19
6.2 Point Cloud Library	21
6.2.1 Preparing the cloud	22
6.2.2 Smoothing	24
6.2.3 Reconstruction	26
6.2.3.1 Greedy Triangulation	26
6.2.3.2 Grid Projection	27
6.2.3.3 Poisson	29
6.2.4 Normals	30

6.2.5	ReCap	31
6.3	Comparison	32
7	Use Case	37
7.1	Introduction	37
7.2	Collaboration	37
7.3	Design	38
7.3.1	The Manuscript	38
7.3.2	The Storyboard	38
7.3.3	The Models	39
7.4	Implementation	40
7.4.1	The Props and Accessories	40
7.4.2	From Polygons to Animation	41
7.4.2.1	The Models	41
7.4.2.2	The Rigging	42
7.4.2.3	The Animation	43
7.4.3	ReCap Mesh	44
7.4.4	UnityEngine Implementation	45
7.4.4.1	3D Models and Animation	46
7.4.4.2	Cameras and Canvas	47
7.4.5	Sound	48
7.5	Test	49
7.5.1	Interview results	49
7.5.1.1	The Computer Science Students	49
7.5.1.2	The Audience	50
7.5.1.3	The Client	50
7.6	Sub-Conclusion	51
8	Discussion	53
9	Conclusion	57
A	The Manuscript	59

Abbreviations

IPS	I nital P roblem S tatement
FPS	F inal P roblem S tatement
PCL	P oint C loud L ibrary
VTK	V isualization T oolkit
SfM	S tructure from M otion
UAV	U nmanned A erial V ehicle
GT	G reedy T riangulation
GP	G rid P rojection

This page was intentionally left blank.

Chapter 1

Introduction

The following report will have in focus 3D reconstruction based on video footage from a drone. Interest in computer graphics was a great motivation for wanting to learn about 3D reconstruction and in what type of scenarios this could be used for in the real world. To gain this knowledge, the chapter investigation will go through a variety of programs used for this purpose and an introduction of the drone. It will shortly cover professional drone companies and other reconstruction projects. In the method chapter theory point cloud data and structure from motion will be introduced. Both topics are techniques used to turn 2D images into 3D reconstructed models. The different techniques used for making 3D reconstruction will be discussed and tested from different aspects, later in the implementation chapter. This is done in order to find the most optimal reconstruction technique for each of the object sizes selected for the test.

The report will continue with a use-case made as an example of how 3D reconstruction can be used in a real world scenario, in this case, in an application for a museum. The collaboration with Mosede Fort makes it possible to develop an application with 3D reconstruction as a focus point and at the fulfill the client's needs. The experience of working with a client was used as a motivation for expanding the project to more than just learning about theory but also learning how to implement it for everyday use. It should be stated that the main focus for this project is to find the most optimal technique for 3D reconstructions which therefore leads to an in depth comparison between the different techniques. The use-case will therefore be most used to prove that improving and working with 3D reconstruction can in fact be used in real life scenarios. The development and creation of this use-case can be read in the chapter of the same name. The findings from the use-case will not be concrete information for statistic purposes and with the goal of proving a theoretical point, but instead used to get a an insight of whether or not this 3d reconstruction is applied in a usable scenario, in this case an application for a museum. The report will end with a discussion including a list of improvements that could be implemented in the future and lastly to round of a conclusion.

This page was intentionally left blank.

Chapter 2

Initial Problem Statement

Is it possible to use video live feed from a drone to create a similar 3D environment?

This page was intentionally left blank.

Chapter 3

Investigation

This chapter will go in depth with information regarding drones and 3D environment related subjects. Firstly it will start out by describing programs that allow users to create reconstructions using point cloud data followed by reconstruction projects and cases where drones are used professionally. This section's focus is to give an insight on the field this project is based on and guide the report towards a final problem statement.

3.1 Point Cloud Library

The Point Cloud Library (PCL) is a free library which allows the user to work with point clouds and 3D geometry and is mainly used as a tool to allow robots to see the world in a 3D manner. The library is written in C++ programming language and is available on multiple operating systems such as Mac, Windows and Linux. [Rusu and Cousins, 2011]

PCL contains a various range of libraries that perform processing algorithms which focus on point cloud data, some of these libraries are;

- `libpcl_filters`: Implements data filters such as downsampling, extraction, etc
- `libpcl_features`: Implements features such as surface normals, curvatures, integral images, etc.
- `libpcl_surface`: Implements techniques that reconstruct surfaces like meshing, convex hulls, Least squares.
- `liblevel_set_segmentation`: Implements cluster extraction, model fitting, polygonal prism extraction, etc.

PCL also uses a personalized visualization library based on the visualization toolkit (VTK) which offers a platform for rendering 3D point cloud and surface data, and also allows texturing and volumetric methods. The combination of the point cloud information and the visualisation tool allows the user to set visual properties to the point cloud dataset or draw basic 3D shapes on screen and create histograms for 2D plots. [Rusu and Cousins, 2011].

3.2 Autodesk ReCap

Autodesk ReCap is a software that focuses on both capturing point cloud data from a wide range of images and creating a 3D model using the images from different angles. There are two different parts of the software, which are used to acquire different actions. [Autodesk, 2015].

One part of the software is called Autodesk ReCap 360, which is a website allowing you to upload images of the objects and after a certain waiting time it creates a point cloud data and a 3D model using this data, if the images were not clear enough or proper and the software fails to create the data, it notifies the user that the model cannot be created via email. Figure 3.1 shows what kind of features ReCap 360 allows.

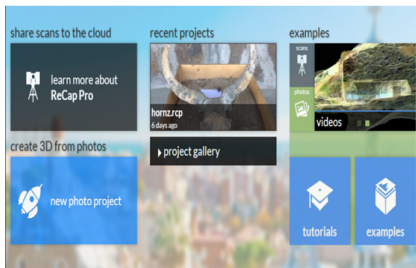


Figure 3.1: ReCap 360's online interface.

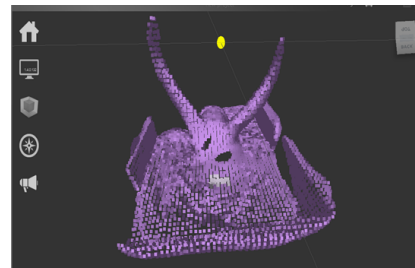


Figure 3.2: Example of a point cloud from ReCap.

The program requires full resolution photos and they need to provide information of the object from different angles. At least 6 images are required to be added to the software in order for this to be able to create the 3D model while the maximum amount of images is 250. Once the images are uploaded and the project has finished being created, it is possible to export the file of the model as an Obj file to be imported in Autodesk Maya or 3ds Max, however you can also export them as an .rcs file which can then be imported in ReCap Pro which then allows the user to increase the accuracy of the point clouds.

Some of the features that ReCap Pro offers are; photorealistic data visualisation using the point cloud data, a detailed 3D design and also allows the user to edit and clean the point cloud data creating a higher accuracy in the representation of the point cloud. However, the ReCap Pro is mainly aimed at data information gathered from 3D laser scanning and not photography, therefore it is possible to combine the usage of the two programs ReCap Pro and ReCap 360 to achieve point cloud data and alter it afterwards using the Pro version. An example of how the program uses the data can be seen in Figure 3.2.

3.3 VisualSfM

VisualSfM is a program that uses Structure from Motion (SfM) to reconstruct 3D objects using point cloud data. VisualSfM is a combination of multiple tools [Wu,

2011], which combined, can quickly create a point cloud based on multiple images and be exported as a .ply file. The interface is simple and consists of a simple window with a menubar (see Figure 3.3).



Figure 3.3: VisualSFM interface

To use the program, you must first load in the images needed for creating the point cloud, then Compute Missing Matches algorithm and after that perform Compute 3D Reconstruction.

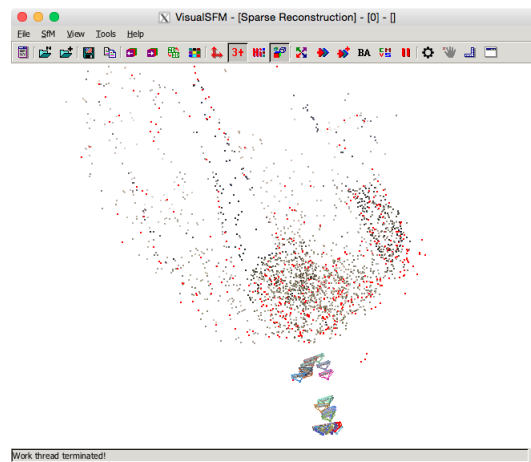


Figure 3.4: Example of the Compute 3D Reconstruction phase.

Last action to perform is to Run Dense Reconstruction, which gives a .ply file. This file can then be loaded into a point cloud viewer such as ccViewer. (see Figure 3.5) [Girardeau-Montaut, 2013].

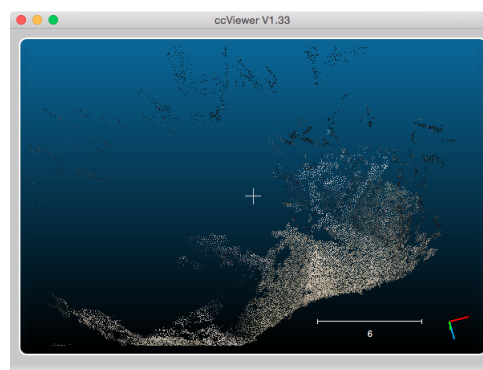


Figure 3.5: Point cloud in ccViewer

VisualSFM is fast at processing images even when having a lot of details and many images, however the built in viewer is not accurate at displaying the point cloud. Instead

of using the build-in viewer, using a program like ccViewer or Meshlab, gives a better overview and control.

To use the constructed point cloud it needs to be converted into a polygon mesh. This can be done using Meshlab or PCL.

3.4 Meshlab

Meshlab [Meshlab, 2014] is a software for creating polygon meshes out of point clouds which is achieved using various filters and cleanup methods. When importing a point cloud into Meshlab, it usually has to be cleaned up. This means removing the points that are not part of the point cloud that is wanted. Using the Select Vertexes and Filters→Selection→Delete Selected Vertexes it is easy to remove the unwanted vertices before moving on to creating the polygon mesh. If the mesh contains any faces, it is also possible to delete those to clean up the cloud. When the cloud is cleaned up it is possible to apply different filters to connect the cloud into a mesh. To reconstruct the mesh, filters such as Pall Pivoting, Poisson or VCG can be used.

3.5 C3 Technologies

One of the popular 3D reconstruction technologies is C3 Technologies. This company based in Sweden has been using declassified missile targeting methods to provide highly detailed 3D maps based on aerial photos [Gurman, 2011]. The maps are created by taking multiple photos at specified angles from a plane flying 1.600 feet above the ground. The plane flies multiple flyovers of a city to get photos from all the required angles, these photos are then combined and analysed to create stereoscopic depth and reconstruct buildings. These buildings are not modeled in 3D (like a building in Maya would be) but instead projected stereoscopic as 3D. This makes the software very fast and requires no user interaction. C3 Technologies are currently providing all the 3D details to Apple's Maps software, which is available on OS X and iOS.

3.6 Drone 55

Drone 55 is a company that specialises in using drones to create aerial cinematography. The company hires professional pilots to fly the drones and has the equipment required capturing aerial footage that would otherwise be expensive to achieve while working on your own. They, not only, have professional pilots, they also have employees who specialize in films and cinematography, guiding the pilots flying to drone to capture useful and visually pleasing footage. Their goal is to offer high quality videos to their clients and have so far worked with; real estates, events, marketing, architecture and construction and also have experience with achieving high resolution photos for computer generated renderings. Their equipment involves not only GoPro cameras but also really high quality cameras and various drones to fly these high quality cameras. One example

of the Cinema rig equipment, the Heavy Lift Octocopter, which has around 8 minutes of flight time, the option for real time video broadcast and also GPS auto-pilot system offering position holding and stable hovering. To it attached is one of the high quality cinema cameras RED Epic Dragon, C300. [Drone 55, Inc., 2015].



Figure 3.6: Image of Drone 55's Aerial Cinematography drone.

3.7 Octofilms

Octofilms is a crew of professional pilots situated in Burbank, California who offer aerial footage captured by themselves as well as teaching building and flying your own drone. Octofilms also builds drones to suit their clients needs, for example having a drone circle a building or to check whether a certain place is safe to access. Octofilms also provides university classes such as general introduction to drones, building unmanned aerial vehicles (UAV), maintenance and flying. Their introduction subject regarding drones starts out with giving information regarding UAV aircrafts assembly, maintenance and safety. Octofilms also prides themselves in being able to acquire cinematographic aerial footage and having suitable drones to carry high quality cameras similarly to Drone55. [OctoFilms, 2014]. For examples of drone footages achieved by Octofilms access <http://octofilms.com/#!portfolio-item/safeway/>.

3.8 Reconstruction projects

Both Microsoft [Izadi et al., 2011] [Newcombe et al., 2011] and Google [Google] are working on 3D reconstruction projects based on mobile or indoor use. Microsoft has been doing a lot of research in computer vision and reconstruction using their Kinect devices and has various libraries available for the Kinect. Stanford also has had some interesting reconstruction projects [Saxena, Sun, and Ng, 2007a] [Saxena, Sun, and Ng, 2007b] focusing on single image reconstruction. This leaves a lot to desire for a proper reconstruction, but the result is quite interesting and gives a proper 3D effect. Another interesting project is done at Zuse Institute Berlin [Zuse Institute Berlin, 2006-2015] where they have undergone a lot of research about 3D reconstruction of bones from 2D X-ray images.

Autodesk also has a mobile application for 3D reconstruction: Autodesk 123D Catch [Autodesk, 2014]. Like ReCap (see Section 3.2 Autodesk ReCap), the application focus on uploading pictures and outputs a 3D model. The difference with Catch compared to ReCap is that it is a mobile application, so the user can take images with their phone or tablets and immediately see a model. When using the application, it is simple to capture images and a simple widget keeps track on where images are positioned around the object and guides the user. The downside of the application is the time it takes to reconstruct the model, a small toy figure takes about 20 minutes to 3D reconstruct.

An interesting project using a drone as image acquisition method is the reconstruction of Rio de Janeiro's Christ the Redeemer [Chen, Betschart, and Blaylock, 2015]. The project used a drone to capture 3.500 images of the statue and later used a combination of Pix4D and manual modeling to complete a textures 2.5 million triangles mesh (see Figure 3.7).



Figure 3.7: Model of Christ the Redeemer[Chen et al., 2015].

Chapter 4

Final Problem Statement

What would the ideal reconstruction technique for big scale outdoor environments be when using a drone to gather point cloud data and what possible application can a reconstruction of this be used for?

This page was intentionally left blank.

Chapter 5

Methods

This chapter will introduce concepts regarding 3D reconstruction which will lead to the process of implementation. Firstly, an introduction to Structure from Motion and how it works will be presented, followed by a section about Surface Reconstruction using Point Cloud Data. Finally, a short discussion of the criteria leading to a successful reconstruction in the implementation will be described.

5.1 Point Cloud Data

A point cloud represents a collection of various points set in different dimensions which are commonly used to represent 3 dimensional data. In point clouds the points refer to the X, Y and Z geometric coordinates of a surface. Point clouds are acquired by devices such as laser scanners, stereo cameras or can be generated from a computer and can be applied in sciences as geoscience, medicine, manufacturing, etc. The data acquired from the scanners can be used to recreate certain objects digitally, represent shapes or detect particular surfaces and their topology. [Mémoli and Sapiro, 2005].

Stereoscopy is one way of acquiring 3D surfaces, the data can be gathered by creating a disparity map from two images viewed in two distinct positions and using the difference between matching points to achieve a depth map. Another way to gather 3D data is by using *Time of flight* laser scanners which measure the elapsed time between a light emission and sensor detection. This method is mainly used for large objects and measures the surface one point at a time. Triangulation laser scanners are also widely used in acquiring data points in a 3D coordinate system. [Digne, 2010, p 30] The data gathered from devices such as the ones presented above is given as a set of point with either a good or bad precision of the object surface. With the point data it is possible to create a mesh of the surface. It is also possible to represent the surface using splines or NURBS surfaces. [Digne, 2010, p 33].

There are four main steps to be considered when using the point cloud data to reconstruct a surface;

1. Once the point cloud data has been gathered it is important to eliminate irrelevant

data and sample points to reduce the computational time.

2. Finding the relations between the data points their connectedness, continuity and boundaries.
3. Start the process of generating the surface using polygons and create the meshes.
4. Edit the model created to perfect the polygonal surface resulted.

[Fabio, 2003, p 4].

Triangulation or mesh generation is the most important part in the reconstruction of the model from the data. Methods that aim to recreate detailed object from the point data use as output either a discrete surface or an implicit function. Implicit functions are based on an underlying grid while the reconstructed surface will be achieved via isocontouring. Other methods use grids such as octrees or adaptive 3D triangulations for the reconstruction; however contouring octrees can bring up difficulties in creating models with completely closed meshes. [Berger et al., 2014, p 7].

Many methods of surface reconstructions require normal associated with the point clouds, the normal can either be oriented or unoriented. Normals that do not have a direction are called unoriented normal, which means it is to be expected the normal to be pointing either on the inside or outside of the surface. This sort of information is useful to determine planar regions in a point cloud, projecting a point onto an approximated surface or achieving covariance matrix. An oriented normal on the other hand have consistent directions and it is known which point outside or inside of the surface.[Berger et al., 2014, p 5].

5.2 Structure from Motion

The world is a multitude of 3D objects that are perceived as 2D projections. Researchers today work in order to find a way to recover a 3D representation of what is visible in order to use it for recognizing and identifying objects. Looking from a computer aspect, a tractable and theoretically well-posed problem that is occurring is the computation of 3D geometry based on the 2D geometry that has been retrieved, based on this Structure-from-Motion (SfM) can be introduced. [Jebara, Azarbajejani, and Pentland, 1999].

There are many different techniques regarding the issue of scanning real-world objects in order to recreate them as 3D models on a computer. Some examples of these techniques could be 3D laser scanning to depth from defocus estimation. Another example could be the Structure from Motion that is in focus here. SfM is considered a useful alternative that are able to construct 3D coordinates and 3D images based on 2D information of real-world objects in a flexible manner. The framework of SfM can also include parameters of 3D motion, which can be useful for animations. [Jebara, Azarbajejani, and Pentland, 1999].

5.2.1 The Structure from Motion Task

When working with SfM tasks, there are multiple simplifying assumptions involved when referring to the problem of creating 3D models based on 2D information gained from imagery. One of the important assumptions is that the movements of the objects remain rigidly and equivalently, and that the only moving object in the environment is the camera. A second simplification is that the cameras images are pre-processed at all times to extract and locate 2D features while labeling them. 2D features consist of e.g. corners, edges, such as lines or curves indicating the contours of the objects, etc. The information given about each 2D feature in the frame is then checked to correspond and match with other 2D features in other frames. This information about the 2D objects is what the SfM problem takes as input. [Jebara, Azarbayejani, and Pentland, 1999].

5.2.2 Greedy Triangulation

The focus of Greedy Triangulation (GT) is to keep a list of possible points that can be connected to create a mesh. Greedy triangulation is obtained by adding short compatible edges between points of the cloud, where these edges cannot cross previously formed edges between other points

GT can, however, over-represent a mesh with triangles when these are represented by planar points. GT contains accurate and computationally efficient triangulations when the points are planar. However, a big difference can be noticed when the triangulations are non-planar, this can be fixed by increasing the number of boundary vertices or linearly interpolating between the boundary vertices. [Ma et al., 2013, p 6].

An approach at computing the GT is to compute all distances, sort them and examine each pair of points in length and compatibility with edges created. The Greedy Triangulation algorithm is simple at its core but also not very reliable. With a point cloud, S , of n points, the algorithm looks for the closest point where a compatible edge can be created between the two. A compatible edge can be described as an edge between two points that does not intersect with any other edge [Dickerson et al., 1994], [Goldman, 1989], [Levcopoulos and Lingas, 1992].

In the Point Cloud Library, greedy projection triangulation works locally, however it allows the specification of the required features to be focused on. Such parameters are neighbourhood size of the search, the search radius and the angle between surfaces. [Hyvärinen, 2012, p 20].

5.2.3 Grid Projection

A different approach to the 3D reconstruction is to use Grid Projection (GP) [Li et al., 2010]. GP is an extremal surface reconstruction algorithm that tries to create seamless surfaces that lack boundaries or proper orientation. One of the defining points for GP is that a continuous surface is created, as opposed to GT where the surface is not continuous and can contain holes.

5.2.4 Poisson

There are a multitude of surface reconstruction techniques, however each one of them poses a number of difficulties when applied to data points. Some schemes can be classified in global and local fitting methods. Global fitting methods are represented in implicit form and can be represented as the sum of radial basis functions (RBFs) at the centre of the points while local fitting methods use the distance from tangent planes to the nearest points. Poisson surface reconstruction combines both the local and global methods; however the RBFs functions are associated with the ambient space and not the data points. The Poisson reconstruction uses the Poisson equation, which is also known to be used in systems that perform tone mapping, fluid mechanics and mesh editing. The Poisson reconstruction approached by Michael Kazhdan, Matthew Bolitho and Hugues Hoppe has set as goal to reconstruct a watertight model by using the indicator function and extracting the isosurface. The indicator function is a constant function whose computation would result in a vector with unbounded values. The challenge these three researchers were faced with was to accurately compute the indicator function from the sampled data points. They approximated the surface integral of the normal field using a summation of the point samples and lastly reconstructed the indicator function using its gradient field as a Poisson problem.



Figure 5.1: Buddha statue reconstruction using Poisson.

One limitation of the Poisson reconstruction is dependent on the acquisition of the data points. For example in Figure 5.1, two reconstructions of a Buddha statue can be seen, the left reconstruction using Volumetric Range Image Processing (VRIP) while the right one using Poisson. At the feet it can be noticed that the Poisson reconstruction connects the regions, which poses some limitations in the use of Poisson reconstruction.

The time taken to perform the Poisson reconstruction increases by adding octree depth while the number of output triangles increases by a factor of four. The Poisson reconstruction uses the indicator function that works best with noisy data points and can therefore recover fine details. [Kazhdan, Bolitho, and Hoppe, 2006].

5.2.5 Delaunay Triangulation

As with the other reconstruction methods, the idea behind Delaunay Triangulation (DT) is to reconstruct a surface S from a finite set of points P . While Greedy Triangulation only looks at the closest point where a compatible edge can be created, the DT also looks at how pleasing the points are.

“ AB is an edge of the Delaunay triangulation, if there is a circle passing through A and B so that all other points in the point set, C , where C is not equal to A or B , lie outside the circle.” [Peterson].

This means that the mesh created by DT is a lot cleaner and more accurate than GT.

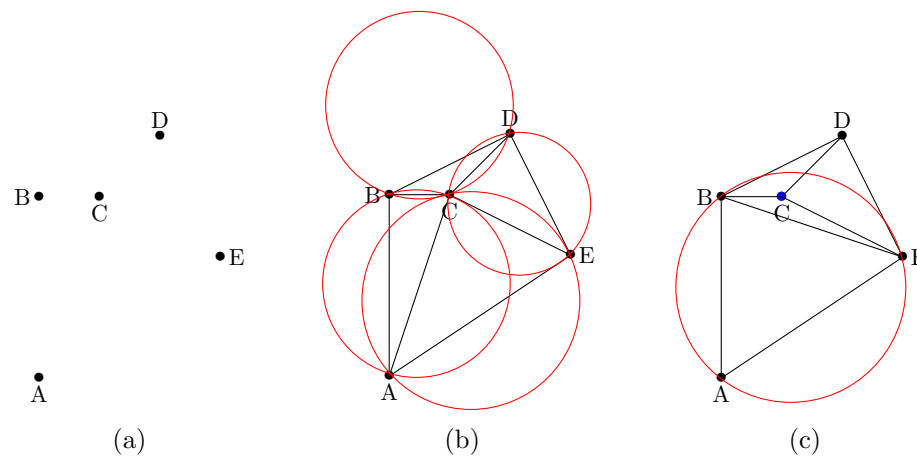


Figure 5.2: Example of Delaunay Triangulation on a point cloud (b) and Greedy Triangulation on the same cloud (c).

When looking at Figure 5.2 it is possible to see two different examples of triangulation methods. In Figure 5.2a a very simple five points point cloud is shown. In Figure 5.2b four circles are drawn to pass through the points ACB, ACE, CDE and BCD. These circles do not contain any excess points and all the different lines are enclosed in a circle. On the other hand, in Figure 5.2c the AC line from Figure 5.2b is replaced with a BE line as would be the result of a GT. This makes it impossible to create a circle around the ABE points without having C inside the circle.

When looking at the two triangulations it is more pleasing to look at (b) because connecting the points AC gives a better representation than connecting the points from BE in Figure (c). [Cazals and Giesen, 2006].

5.2.6 Success Criteria

A criteria for the success of the various reconstruction algorithms is that they are fast and reliable. For that purpose, the computation time is recorded for each algorithm with various settings and each mesh is manually looked at for checking how reliable

and accurate the mesh is. For computation time, a simple C++ code snippet is added around each algorithm.

For the reliability and accuracy of the mesh, three different topics are looked at; holes, amount of details and noise.

- Holes: how many holes are there and how do they affect the finished mesh?
- Details: is the mesh too smooth or are there enough details to see the smaller changes in mesh?
- Noise: how much noise is present in the final mesh? Noise is defined as polygons that manually need to be removed in another software package.

These criteria combine both subjective measurements with objective observations, which in the end should have a proper conclusion on what algorithms to use on what types of meshes.

Based on initial theory the following is proposed:

- Greedy Triangulation works best for smaller meshes with few details and closed point cloud devoid of noise.
- Grid Projection works best for larger meshes that are somewhat flat and without too much noise.
- Poisson works best for enclosed meshes that do not contain any edges or larger holes.

The three above mentioned methods were chosen because they are part of the Point Cloud Library and allows the reconstruction of the point clouds gathered. Delaunay Triangulation is not going to be implemented as it is not part of the Point Cloud Library and it was deemed too ambitious to implement our own version for this project.

A final point to keep in mind when using a reconstruction technique is the time it takes to record images, compute the point cloud and reconstruct the mesh should not take significantly longer than creating the mesh in a 3D software such as 3ds Max or Maya. To sum up, what is aimed for with this project is for people that are not familiar with such applications or not skilled in 3D modeling, to create accurate meshes in as little time as possible.

Chapter 6

Implementation

This chapter will go through the whole implementation of the three reconstruction techniques and the point cloud acquisition. It starts out with explaining the Image Acquisition process, what drone and camera was used for the process, before moving on to the PCL implementation with code examples and finally ending with a comparison section, where the different reconstructions will be compared and discussed.

6.1 Image Acquisition

In order to get aerial footage, a DJI Phantom was used together with a GoPro Hero 3 camera (see Figure 6.1). The drone was very easy to fly with and contains a GoPro mount for easy attachment of the camera. All footage was recorded as 1080p video at 24fps and then imported into photoshop, where it was exported as images. The images were then imported into VisualSFM to create a point cloud.



Figure 6.1: Image of the DJI Phantom drone iwth a GoPro camera attached [Corto, 2013].

The DJI Phantom was very simple to operate with and has a decent stabilization. It is possible to get a better camera mount that includes gyroscope so that the camera will

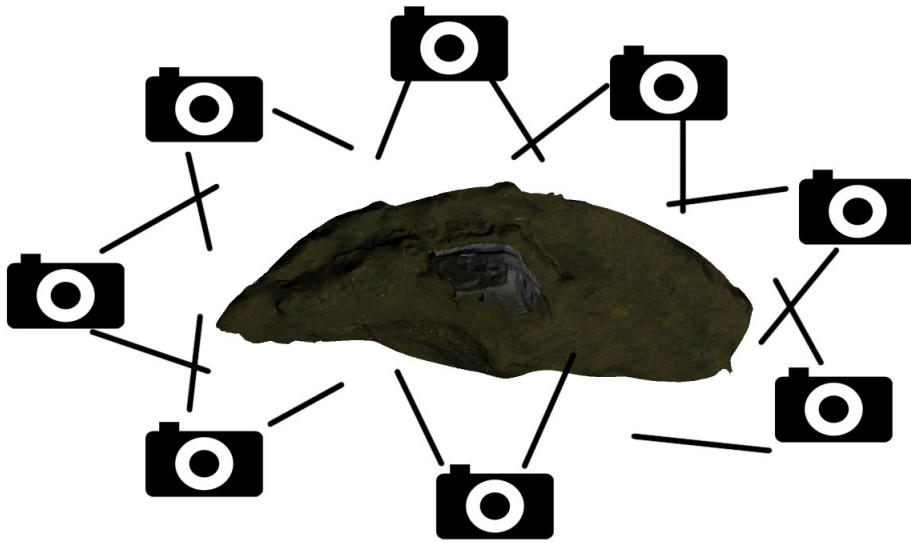


Figure 6.2: Example of how the camera was placed when the data was gathered.

always record from the same angle, but when looking at the footage it was not deemed necessary to do any post stabilization. The Phantom's battery has approximately 15 minutes of flight time and when the battery is low the Phantom will try and land itself. This happened a few times during the footage acquisition, however the auto landing mechanism does not always work perfectly.

While recording, the GoPro was controlled remotely by streaming its video feed to a Google Nexus 7 tablet where the camera could also be stopped. The GoPro creates a simple Wi-Fi connection that the Nexus can join and then remotely control the camera from.

The images could also be acquired without a drone. The photographer would then need to either take pictures of smaller objects or be able to get high up and take some overview images.

Figure 6.2 shows how the camera was placed when the images for the point cloud were taken.

Table 6.1 represents the three types of reconstruction; Greedy Triangulation (GT), Grid Projection (GP) and Poisson methods graded based on their accuracy level according to the sizes of the reconstructed objects. The assessment table shows the techniques' performance level based on accuracy, in this case, by accuracy it is referred to how well the reconstruction is represented compared to the original object and how detailed it is. Accuracy in these assessments also refers to the amount of unwanted holes the mesh presents, therefore a high amount of holes is considered as low accuracy. These assessments are based on the theory gathered in Chapter 5.2.6 and represent the accuracy levels for three different object sizes.

Three different sets of images have been acquired and will be used for testing reconstruction. The images can be seen in Figure 6.3.

Accuracy Level			
Object size	GT	GP	Poisson
Large scale	Low	High	Low
Medium scale	Medium	Medium	Medium
Small scale	High	Low	High

Table 6.1: Assessment table of the accuracy level

Figure 6.3: Examples of three different objects that will be used for reconstruction. 1. House in Sweden, 2. Caponiere at Mosede Fort, 3. Stanford Bunny².

These three objects/places have been chosen because of their unique features. The house is a representation of an outdoor object that is almost closed off (no open areas) with flat ground surrounding it. The Caponiere¹, is also an outdoor structure but it is a lot more open while the surrounding area is shaped around it and not flat. The last object is the Stanford Bunny. This is a classic object in computer graphics (like the Utah Teapot) and is a small object without anything surrounding it. The bunny is an example of an indoor object, as opposed to the other two objects.

6.2 Point Cloud Library

The open source point cloud library, PCL, will be used as the foundation for the 3D reconstruction (see Section 5.1 Surface reconstruction Using Point Cloud Data) [Point-Clouds.org, 2014b]. This library provides a simplified approach of extracting data from a point cloud and visualize it.

PCL provides different algorithms that can be used for 3D reconstruction such as; Greedy Triangulation, Grid Projection and Poisson. Many of those algorithms are also available in Meshlab (see Section 3.4 Meshlab) but what this project wants to accomplish is a complete software solution for 3D reconstruction and therefore Meshlab's GUI solution is not useful for this type of approach.

The next sections will go through the implementation of the different algorithms used from PCL and show the results achieved and discuss the encountered difficulties of creating a 3D reconstruction.

¹A fortification structure, the name comes from the French word caponnière

²The Stanford bunny point cloud has been downloaded from the Stanford 3D Scanning Repository [Stanford University Computer Graphics Laboratory, 1994].

6.2.1 Preparing the cloud

Before going into the implementation of greedy triangulation, grid projection and Poisson, it is necessary to look at the input data that is provided to these algorithms and how it is processed.

The basis for 3D reconstruction is a proper point cloud. While researching into PCL, it became clear that the library does not provide a way of creating a point cloud from a range of images. To make the point cloud, VisualSFM was used (see Section 3.3 VisualSFM). This program provides a point cloud, in the .ply format, based on a series of images. The cloud generated by VisualSFM is generally decent looking but the problem that arise is that VisualSFM exports the data as .ply format. PCL, however, has a support for .ply but it is a known bug that the `pcl::io::loadPLYFile()` function gives errors on certain operating systems. Therefore, the preferred format for PCL is a .pcd file. To convert the .ply to a .pcd file it was necessary to understand the header information in both of these files. When looking at the header files in Code 6.1 and Code 6.2 it is possible to see some similarities.

Code 6.1: .ply header

```
1 ply
2 format ascii 1.0
3 comment PCL generated
4 element vertex 35291
5 property float x
6 property float y
7 property float z
8 element camera 1
9 end_header
```

Code 6.2: .pcd header

```
1 VERSION .5
2 FIELDS x y z
3 SIZE 4 4 4
4 TYPE F F F
5 COUNT 1 1 1
6 WIDTH 35291
7 HEIGHT 1
8 POINTS 35291
9 DATA ascii
```

Both files define basic variables in form of x, y and z information as floats and give information about the amount of vertices in the file. To get a .pcd file that can be used in PCL, all the data points are copied to a new file and a new header is created.

With the .pcd file ready to use, it is possible to start on the filtering and smoothing of the point cloud. Before showing how this is done, a few of the important variables used will be explained. Code 6.3 shows the initial setup of the variables.

Code 6.3: Setup of the most used variables.

```
1 pcl::PointCloud<pcl::PointXYZ>::Ptr cloud (new pcl::PointCloud<pcl::
   ↪ PointXYZ>);
2 pcl::PointCloud<pcl::PointNormal>::Ptr cloud_with_normals (new pcl::
   ↪ PointCloud<pcl::PointNormal>);
3 pcl::search::KdTree<pcl::PointXYZ>::Ptr tree;
4 pcl::search::KdTree<pcl::PointNormal>::Ptr tree2;
5
6 int main(int argc, char** argv) {
```

```

7     std::cerr << ">> Loading file ...";
8     pcl::PCLPointCloud2 cloud_blob;
9     pcl::io::loadPCDFile(argv[1], cloud_blob);
10    pcl::fromPCLPointCloud2(cloud_blob, *cloud);
11    std::cerr << "Done.\n";
12 }

```

Code 6.3 shows the initial setup of the variables. Line 1 and 2 defines the two clouds that are used. `cloud` stores the x, y and z information in a pointer and `cloud_with_normals` stores the information of the normals for each point. Line 3 and 4 represent two k-dimensional trees which are used to organize the cloud points in k dimensions [PointClouds.org, 2014a]. These four variables will be used throughout the whole program.

Next thing is to create a search tree and perform a normal estimation (see Code 6.4).

Code 6.4: Setup search tree, do a normal estimation and concatenate the variables

```

1  std::cerr << ">> Create search tree ...";
2  tree.reset(new pcl::search::KdTree<pcl::PointXYZ>(false));
3  tree->setInputCloud(cloud);
4  std::cerr << "Done.\n";
5
6  std::cerr << ">> Normal estimation ...";
7  pcl::NormalEstimation<pcl::PointXYZ, pcl::Normal> n;
8  pcl::PointCloud<pcl::Normal>::Ptr normals (new pcl::PointCloud<pcl::
    ↪ Normal>());
9  n.setInputCloud(cloud);
10 n.setSearchMethod(tree);
11 n.setKSearch(20);
12 n.compute(*normals);
13 std::cerr << "Done.\n";
14
15 std::cerr << ">> Concatenate XYZ and normal information ...";
16 pcl::concatenateFields(*cloud, *normals, *cloud_with_normals);
17 std::cerr << "Done.\n";

```

The search tree is based on the previously created `KdTree` and takes the `cloud` as input. The normal estimation is stored in a new variable called `normals`. These two variables, the sorted cloud and the normals, need to be stored together in a new variable. The `cloud` only contains information about x, y and z while the `normals` only contains information of the normals. In order to use the cloud properly for the reconstruction, a cloud with both x, y and z values and normal information is needed. Line 16 in Code 6.4 shows how it is possible to take the `cloud` data and the `normals` data and combine it and store it in `cloud_with_normals` [PointClouds.org]. The cloud is now ready for use and the next section will go into details with the smoothing process.

6.2.2 Smoothing

The first step taken after preparing the cloud is to smooth it using Moving Least Squares (MLS). As explained by [Alexa, Behr, Cohen-Or, Fleishman, Levin, and Silva, 2003] the idea behind MLS is to have a data set of points $P = \{p_i\}$ that constructs a smooth surface S_P , however, instead of using the original data set of points P , a reduced set ($R = \{r_i\}$) is created and a new surface S_R is defined.

Figure 6.4 visualises the idea behind MLS. Looking at Figure 6.4a, it is possible to see the cloud (P_i) and the line fit the cloud (S_P). In Figure 6.4b the line (S_P) has been sampled and new points (R_i) have been added. The new points can now have a line smoothed to them using MLS and the result can be seen in Figure 6.4c. In the last figure, Figure 6.4d, a comparison of the two lines can be seen.

The smoothing implementation using PCL can be seen in Code 6.5.

Code 6.5: MLS smoothing

```
1 std::cerr << ">> Smoothing with MLS...";
2 pcl::PointCloud<pcl::PointNormal>::Ptr mls_normals (new pcl::
   ↪ PointCloud<pcl::PointNormal>());
3 pcl::MovingLeastSquares<pcl::PointXYZ, pcl::PointNormal> mls;
4
5 mls.setInputCloud (cloud);
6 mls.setComputeNormals (true);
7 mls.setPolynomialFit (true);
8 mls.setSearchMethod (tree);
9 mls.setSearchRadius (0.03);
10 mls.process(*mls_normals);
11 std::cerr << "Done.\n";
```

The MLS smoothing is done before any reconstruction algorithm is applied. Another smoothing method is also used, however only applied after the reconstruction is complete.

In greedy triangulation and Poisson a new variable called `triangles` is used. It is initialized as seen in Code 6.6.

Code 6.6: Initialization of `triangles` variable

```
1 boost::shared_ptr<pcl::PolygonMesh> triangles (new pcl::PolygonMesh)
   ↪ ;
```

The `PolygonMesh` data type contains information about vertex position, edge data and face data. This data is required for PCL to save the new reconstructed mesh as a file that can be read by another 3D software.

The second smoothing method used is `MeshSmoothingLaplacianVTK`. This is a laplacian smoothing method from the VTK library. As described in the documentation for the original VTK smoothing method: “*The effect is to “relax” the mesh, making the cells better shaped and the vertices more evenly distributed.*” [Visualization Toolkit, 2015].

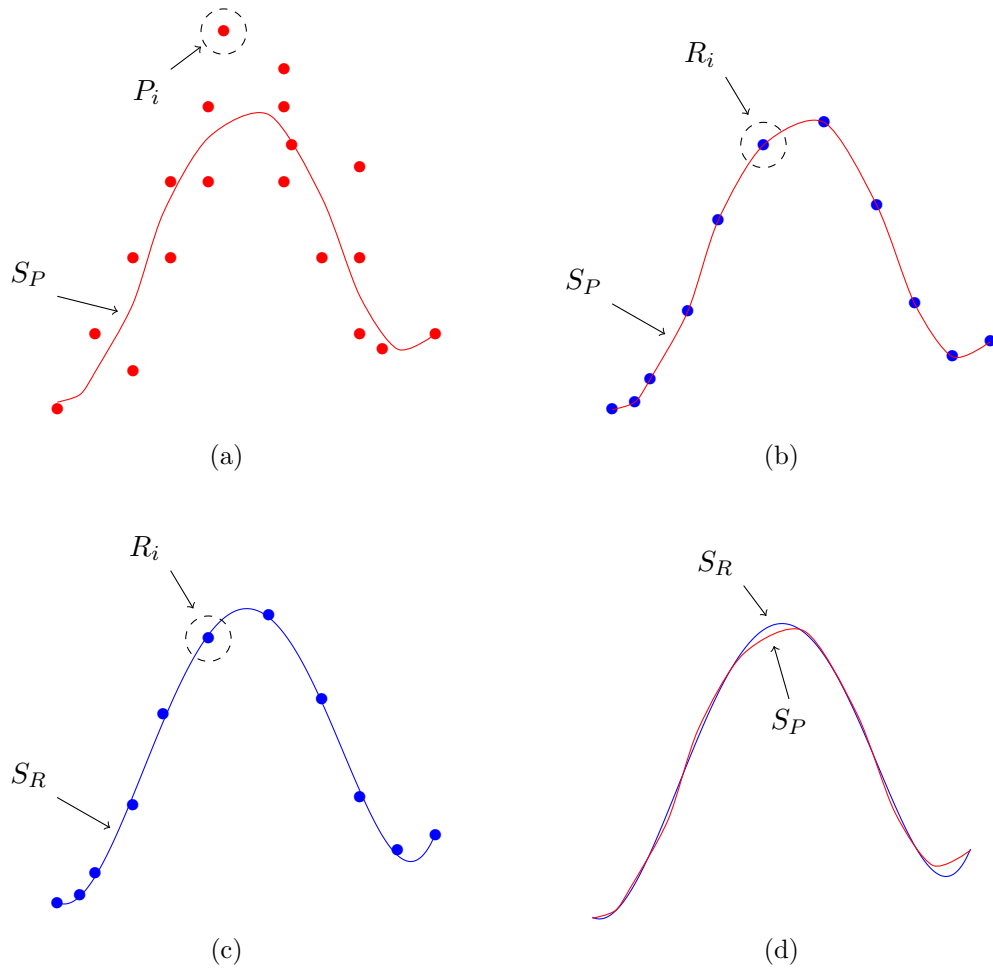


Figure 6.4: Visualization of how MLS works.

Code 6.7: Setup of Laplacian smoothing.

```

1 std::cerr << "Begin Laplacian Smoothing...";
2 pcl::PolygonMesh output;
3 pcl::MeshSmoothingLaplacianVTK vtk;
4 vtk.setInputMesh(triangles);
5 vtk.setNumIter(20000);
6 vtk.setConvergence(0.0001);
7 vtk.setRelaxationFactor(0.0001);
8 vtk.setFeatureEdgeSmoothing(true);
9 vtk.setFeatureAngle(M_PI/5);
10 vtk.setBoundarySmoothing(true);
11 vtk.process(output);
12 std::cerr << "Done." << std::endl;

```

The smoothing algorithm works by starting out looking at all the vertices and for each vertex checks which other vertices are connected to it. An array of connected vertices is created for each vertex and for each vertex this process is repeated multiple times. Code 6.7 shows the setup of the different variables used for the smoothing algorithm. These variables, such as iterations, edge smoothing and boundary smoothing, helps

determine what the algorithm has to do next. Not all vertices are smoothed the same way. Corner vertices are normally not smoothed, edge vertices are smoothed only along the edge and only if the angle of the edge is less than the angle defined in the code. All other vertices are smoothed by taking the array of connected vertices and averaging the values of all connected vertices.

Before running the Laplacian smoothing from Code 6.7 the mesh has to be constructed. To do this, three different algorithms are used and in the next sections, a comparison of the three will be presented in the end.

6.2.3 Reconstruction

The Reconstruction section will introduce; Greedy Triangulation, Grid Projection and Poission. These three reconstruction methods will be used to reconstruct the point clouds acquired in Section 6.1 Image Acquisition and a comparison will be made. The end of this section will shortly describe decisions made regarding ReCap 360.

6.2.3.1 Greedy Triangulation

As mentioned, the idea behind Greedy Triangulation (GT) is straightforward. To accomplish GT using the PCL a new variable `gt` is created which can be seen in Code 6.8.

Code 6.8: Greedy Triangulation implementation.

```
1 boost::shared_ptr<pcl::PolygonMesh> triangles (new pcl::PolygonMesh)
   ↪ ;
2 pcl::GreedyProjectionTriangulation<pcl::PointNormal> gt;
3
4 gt.setInputCloud(cloud_with_normals);
5 gt.setSearchMethod(tree2);
6 gt.setSearchRadius (R);
7 gt.setMu (mu);
8 gt.setMaximumNearestNeighbors (D);
9 gt.setMaximumSurfaceAngle(M_PI/4); // 45 degrees
10 gt.setMinimumAngle(M_PI/18); // 10 degrees
11 gt.setMaximumAngle(2*M_PI/3); // 120 degrees
12 gt.setNormalConsistency(false);
13
14 gt.reconstruct(*triangles);
```

The variables `R`, `mu` and `D` are passed from the main method and are set from the command line via `argv` and `M_PI` are defined as π^3 from `math.h`. The angles are described by how big or small the angles in the created triangles are allowed to be. The result is stored in the pointer `triangles` which can be used in the Laplacian smoothing method mentioned in Code 6.7. After the smoothing process is complete the mesh can be saved as an `.obj` file.

³ $\pi = 3.14159265358979323846$

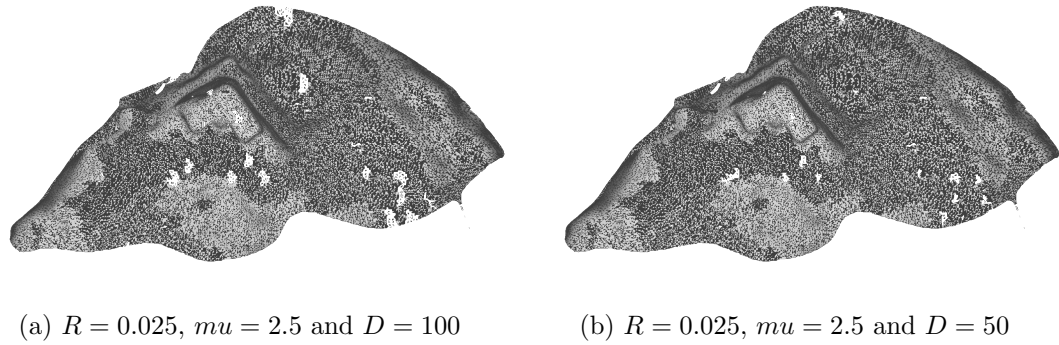


Figure 6.5: Greedy Triangulation with change of D .

As seen in Figure 6.5 lowering the D value (the maximum number of nearest neighbors which the algorithm should search for) will make the mesh have a bit fewer holes (b). Lowering it more than 50 will not have any substantial effect because not enough neighbors are considered and the library automatically increases D until it finds enough neighbors.

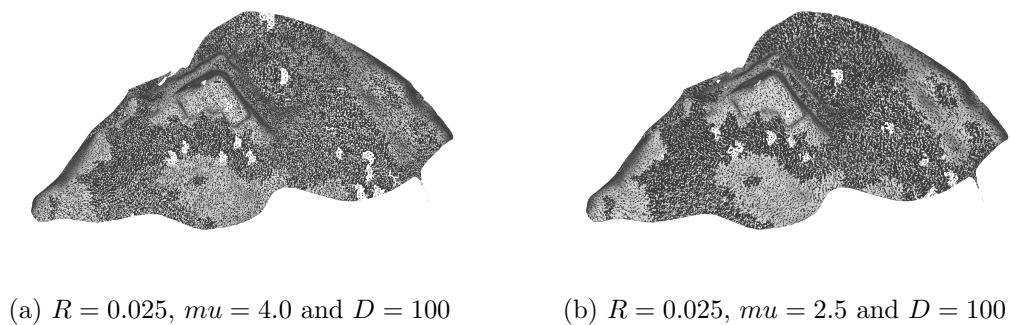


Figure 6.6: Greedy Triangulation with change of μ .

Figure 6.6 shows another variance where the μ values has been increased from 2.5 to 4.0. This change is not notable other than the way the triangles are ordered. On Figure 6.6a the triangles in the mesh are unordered and seemingly randomly created whereas Figure 6.6b shows nice and even triangles.

One problem that occurs with both Figure 6.5 and Figure 6.6 is that the normals are flipped for half the triangles. This is supposedly a known problem with PCL and can be fixed by loading the mesh in Maya and manually fixing the normals by applying Normals→Set to Face and Normals→Conform.

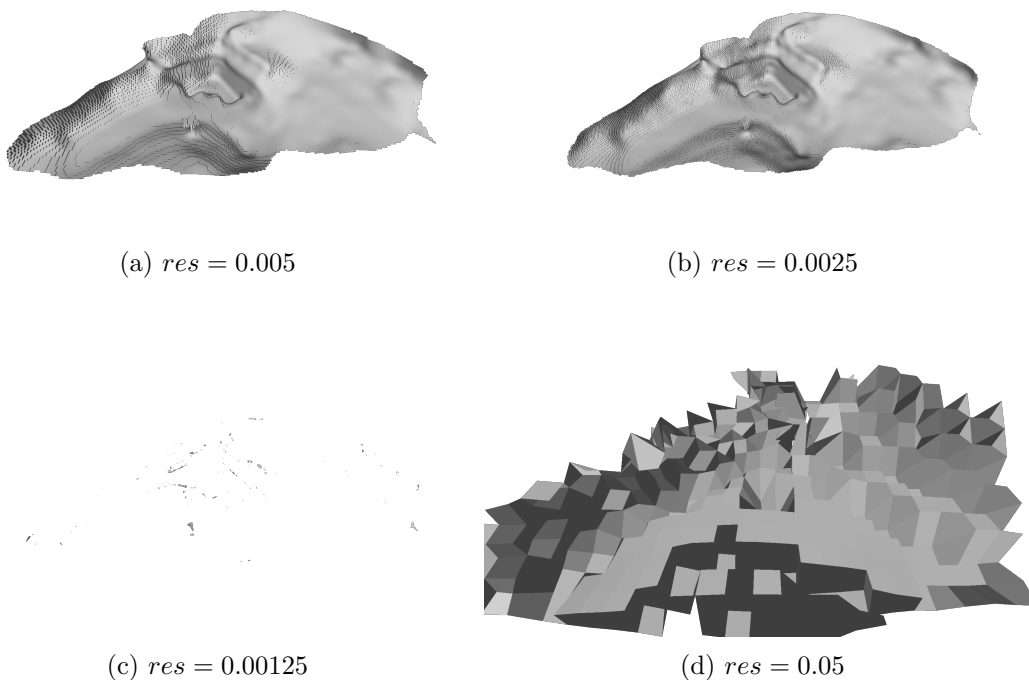
6.2.3.2 Grid Projection

With PCL it is possible to implement Grid Projection using Code 6.9.

Code 6.9: Implementation of Grid Projection.

```
1  pcl::PolygonMesh  grid ;
2  pcl::GridProjection<pcl::PointNormal> gp ;
3
4  gp.setInputCloud (cloud_with_normals) ;
5  gp.setSearchMethod (tree2) ;
6  gp.setResolution (res) ;
7  gp.setPaddingSize (3) ;
8  gp.reconstruct (grid) ;
```

The code is very similar to the greedy triangulation code (see Code 6.8) where a polygon mesh and a grid projection variable are created. The `gp` variable is then used to set the input cloud, search method, resolution and padding properties before the reconstruction begins. For the resolution variable `res` is used. This variable is set via `argv` at run-time and different examples can be seen in Figure 6.7.

Figure 6.7: Grid Projection with change of `res`.

Looking at Figure 6.7a and Figure 6.7b, these two images are smooth and without holes. (a) is a bit more rough and the normals skewed in some places, whereas (b) shows a lot more clear and smooth mesh. The problem with these two reconstructions though is that they are so smooth that a lot of details are lost.

On the other hand, Figure 6.7c and Figure 6.7d have very low accuracy and the meshes are unusable. The idea behind (c) was to try and make the grid a higher resolution than (b) and that way avoid some of the smoothing, however that did not lead to the results expected. Instead, the grid projection could not find enough points

and the results are small scattered lumps of mesh. In the other end of the `res` scale is (d), where `res` is set to 0.05 in order to see what is produced. The result is just as unusable as (c) since the topography is so deformed that it is impossible to see what it is supposed to resemble.

As mentioned, all four representations of the grid projection are not optimal. Figure 6.7a and Figure 6.7b are smooth without holes but too many details are lost, whereas Figure 6.7c and Figure 6.7d have too few details.

6.2.3.3 Poisson

The implementation of Poisson in this project is done as described in Code 6.10.

Code 6.10: Implementation of Poisson

```

1  pcl::Poisson<pcl::PointNormal> poisson;
2  poisson.setDepth(D);
3  poisson.setInputCloud(cloud_with_normals);
4  boost::shared_ptr<pcl::PolygonMesh> triangles (new pcl::PolygonMesh)
    ↪ ;
5  poisson.reconstruct(*triangles);

```

Code 6.10 shows the implementation where `D` is a variable passed using `argv` at runtime. The variable defines how detailed the mesh is going to be and with the given mesh works from 1 to 14.

One problem with Poisson is that it is prone to errors if the point cloud contains noise.

Figure 6.8 shows four different reconstructions using Poisson. It should be noted that the figures have their normals inverted and is represented from below (compared to Figure 6.7, Figure 6.5 and Figure 6.6), because the Poisson algorithm closes all borders to the models which become spherical and with no openings. (a) shows the smallest `D` value that gives a meaningful result. It is possible to see parts of the main shape in the middle of the figure but it is very rough and unusable. (b) on the other hand is very smooth and looking at the compute time it is very low. (c) and (d) are nearly identical when comparing them in both smoothness and in the amount of details, but their compute time is vastly different. With a compute time of over 11 minutes (d) is by far the slowest and it still has a lot of mesh leftover that shouldn't be a part of the model.

The biggest problem with Poisson and the reason why it is not usable for this project, is the fact that so much extra mesh is created. The point cloud is pretty clean and without too much noise but the mesh is still not clean and too much manual cleanup is needed for Poisson to be a valid implementation method.

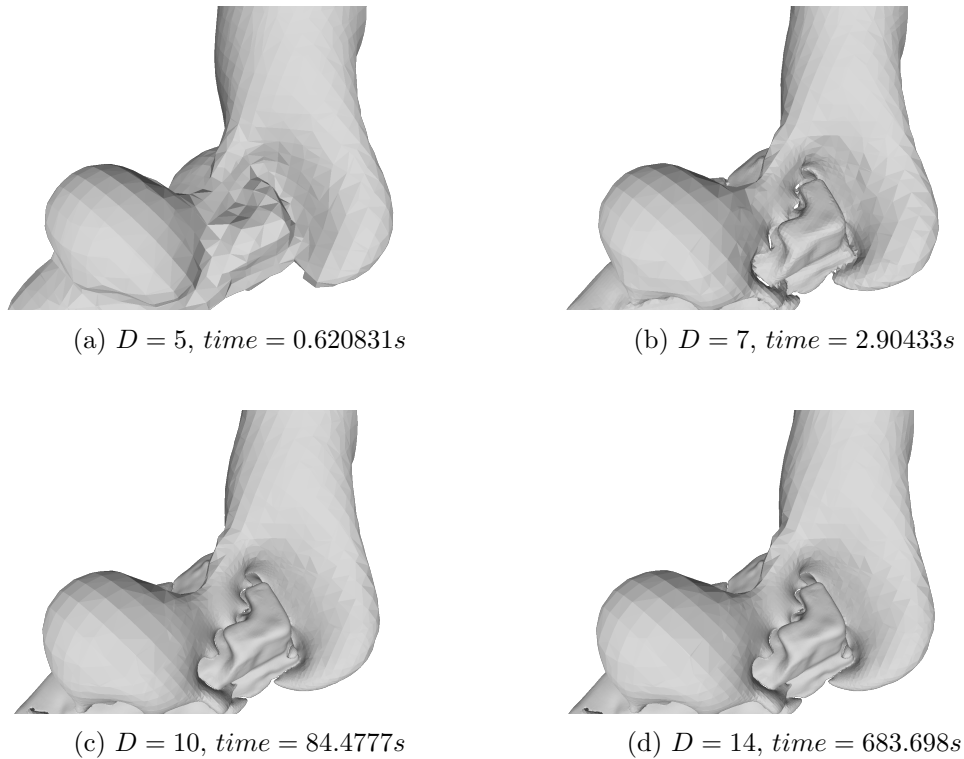


Figure 6.8: Poisson with change of D.

6.2.4 Normals

One of the problems mentioned earlier is the normals. In Figure 6.9a and Figure 6.9c it is possible to see that about half of the normals are reversed. Even though the normals are supposed to be fixed in the code this is not the case when using Greedy Triangulation.

There is a very simple way to average out the normals so they are facing the same way, but it is not doable in PCL. Instead Autodesk Maya will be used and by using the Normals→Conform. Conform fixes almost all the normals as seen in Figure 6.9b and Figure 6.9d but a few normals inside the holes are still reversed.

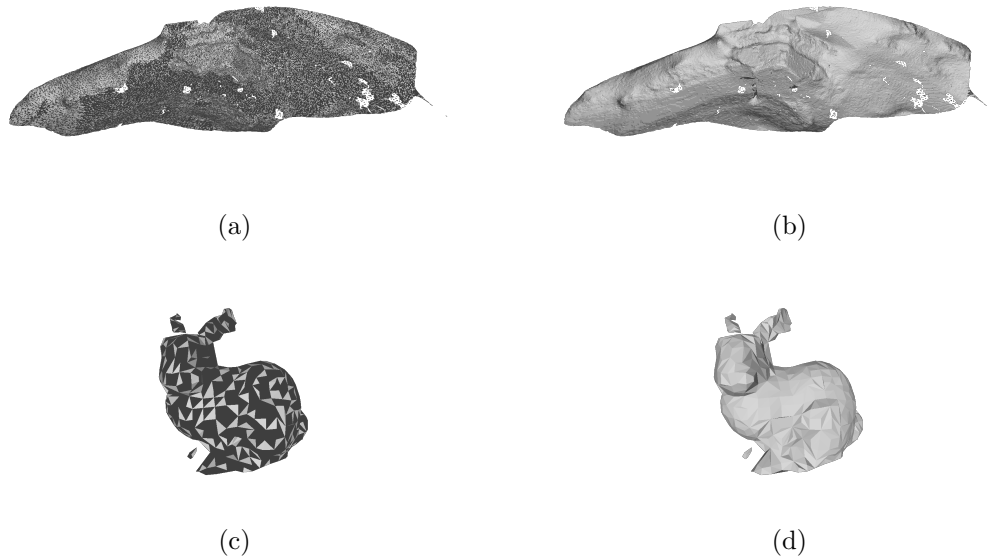


Figure 6.9: Example of the caponier (a) and bunny (c) with opposite normals and caponier (b) and bunny (d) with normals pointing in the same direction.

6.2.5 ReCap

After trying to implement different types of 3D reconstruction techniques, the software ReCap 360 from Autodesk was tested. This software was also mentioned briefly in the Investigation (see Section 3.2 Autodesk ReCap). To sum up, the software takes a range of images as input and outputs an .obj mesh. This is the best ready to use implementation of 3D reconstruction available today that does not require advanced knowledge.

An example of a mesh created using 16 images can be seen in Figure 6.10. The image shows a well constructed mesh without any small holes or deformations, although a big hole is present in the background of the model as none of the images had data on that part.

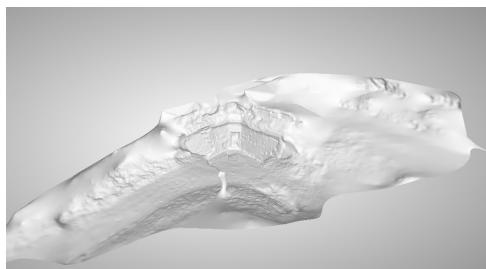


Figure 6.10: Mesh created in ReCap 360.

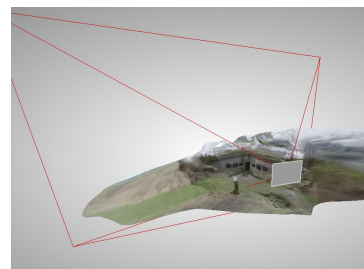


Figure 6.11: Example of where an image is taken from.

ReCap uses the same technique as VisualSFM (see Section 3.3 VisualSFM) and Figure 6.11 shows where one of the images are taken from, based on the mesh.

The problem with using ReCap is that it is very time consuming. Everything is

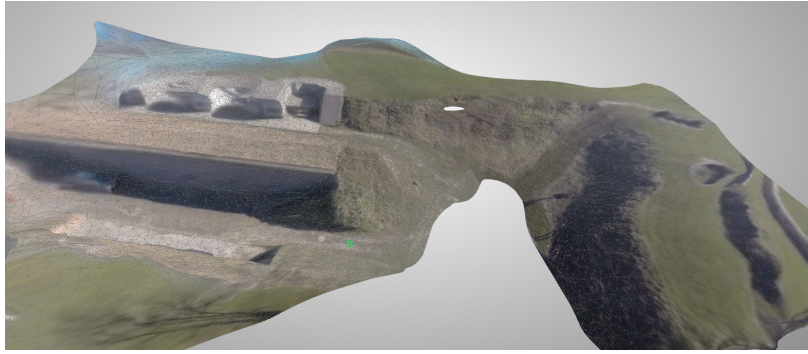


Figure 6.12: Mesh created in ReCap using 250 images.

running on Autodesk's servers but project like the one shown in Figure 6.10 and Figure 6.11 still takes about 30 minutes to run and a 250 image project as the one shown in Figure 6.12 takes about 4 hours.

Even though the outcome could be better than using the other proposed methods, the time it takes is a major con when comparing the different methods.

6.3 Comparison

When working with 3D reconstruction a couple of different observations have been made. It is to be stated that the judgements on the accuracy of the reconstructed models are strictly personal when being compared to their detail level, mesh holes and original object similarity. When looking at Table 6.5 a comparison of the reconstructed meshes can be seen, the following section will describe accuracy levels of the outcome. An overall accuracy level can be seen in Table 6.4.

Since the meshes have different sizes some of the values mentioned in the implementation are different for each reconstruction. For GT the values for all three meshes were:

- $R = 0.025$
- $mu = 2.5$
- $D = 100$

For GP the `res` value for the house and bunny were $res = 0.025$ but for the caponier it was $res = 0.0025$. This is because PCL gave an error when using a too low value for the bunny and house and automatically increased the value. For Poisson the `D` value for all meshes were $D = 10$.

A low accuracy grade is given to the GT of the house due to a high amount of holes that are scattered throughout the mesh therefore not giving a proper usable representation. Even though the opposite happens with the GP house representation it still gives a low accuracy. The GP creates faces around each of the point cloud data and therefore neutralizes the faces resulting in lack of details. Looking at the third reconstruction method of the house, the Poisson, it can be assumed that this is the most accurate

representation of the house with the medium accuracy grade. This grade is based on the smooth surface representation however due to the amount of distorted details a high grade cannot be given. It is safe to assume that one of the reasons it was not possible to achieve an accurate mesh representation for either of the reconstructions of the house was due to the point cloud data, which does not contain enough information on the roof area and the side of the house. Since the house is a large scale object there is a high amount of points in the point cloud data, however these points are highly scattered in certain areas clearly indicating that the amount of data points does not always lead to an accurate mesh.

For the GT reconstruction of the caponier it was required to reverse the normals which is described and illustrated in Section 6.2.4 Normals and therefore it is assumed that a grade fit for this reconstruction is high accuracy. The same occurs with the GP of the caponier however it seems that this representation shows slightly less amount of details than the GT. Lastly, using the Poisson reconstruction badly represents the mesh due to the blobs. Based on the cloud representation it can be seen that it accurately represents the caponier even though the amount of point cloud is much lower than the houses data points. The last reconstructed object, the bunny seems to be best represented by the GT if the normals are properly inverted, which can be seen in Section 6.2.4 Normals Figure 6.9d. The GP and Poisson also appear to represent the bunny correctly however they seem to show a higher amount of flaws in the mesh.

Based on analytical judgement, theories were made which are represented in the Table 6.2 and Table 6.3 and described in the following section. The analytical judgement will be made based on time, computational complexity and memory it requires to compute the reconstruction. Looking at Table 6.2 it can be noticed that the amount of point cloud points is directly proportional with the time it takes to compute, however in the case of the grid projection reconstruction of the caponier is different due to the values presented in the beginning of this section.

Image type	Points	Images	GT	GP	Poisson
House	115794	39	10.0275s	65.3413s	41.6932s
Caponier	35291	34	2.70869s	97.3717s	34.9433s
Bunny	397	Unknown	0.017448s	0.767622s	2.18474s

Table 6.2: Comparison table with time.

When comparing the three reconstruction methods performed on the house point cloud it can still be stated that the GP is the most time consuming reconstruction method for large scale objects. The least time consuming reconstruction technique for big scale objects is, GT while the GP takes the most amount of time to compute the mesh. The same occurs with the reconstruction of the medium scaled objects, the caponier, being able to compute the GT reconstruction in less than 3 seconds, however it is to be noted that the GT reconstruction of the caponier is considered to most accurate representation after the normals are fixed. Based on the timestamps of the

three different scale sized objects it can be concluded, in this case, that all three objects are fastest represented by GT and takes the least amount of memory for the caponier and bunny when performing this.

Algorithm	GT	GP	Poisson
Complexity	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$	$\mathcal{O}(n \log n)$
Memory (house)	80.84MB	46.09MB	263.12MB
Memory (caponier)	27.41MB	96.78MB	205.85MB
Memory (bunny)	1Byte	9.437MB	104.14MB

Table 6.3: Comparison table based on the complexity from Big-O (\mathcal{O}) [Levcopoulos and Lingas, 1992], [Medioni, Lee, and Tang, p 72-73], [Kazhdan and Hoppe, 2013] and Memory use.

However there is not a big difference in the size of the area where the pictures were taken, though there is a difference between the two objects represented by the distance of the camera to the object, which is why it is still possible classify them as a large and medium area. Overall, for the medium and large scaled objects, Grid projection takes

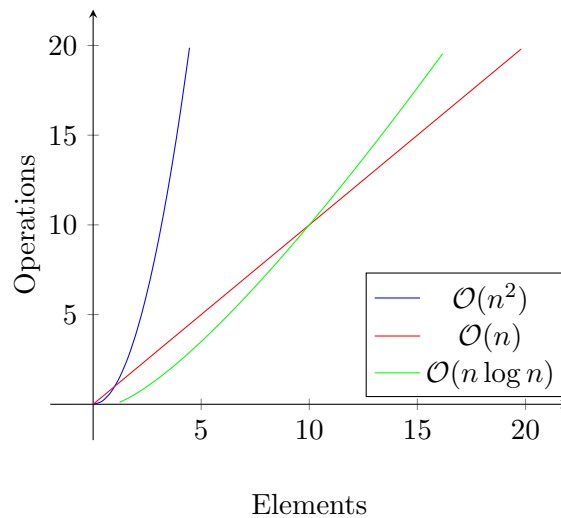


Figure 6.13: Visualization of Big-O notations used for GT, GP and Poisson.

the highest amount of time to reconstruct the objects, while for the bunny the Poisson reconstruction takes more than 2 seconds compared to the other two reconstruction techniques. When looking at the Table 6.3 it can also be noticed that Poisson is the most memory consuming reconstruction for the three scales of the objects.

Looking at Table 6.3 it is possible so see a comparison between the memory use of each reconstruction and an overview of the Big-O notation for each algorithm. The table shows that the the complexity for Greedy Triangulation is quadratic [Levcopoulos and Lingas, 1992], for the Grid Projection it is linear [Medioni, Lee, and Tang, p 72-73] and for the Poisson it is linearithmic [Kazhdan and Hoppe, 2013]. A visualization of the Big-O notation can be seen in Figure 6.13. The grid projection should in theory be fastest at computing the mesh, after a certain amount of elements, where it would

intersect with the linearithmic complexity of Poisson. Greedy Triangulation should have the slowest computing time since the complexity is quadratic. This is however what the theory states but when implementing the algorithms that's not the case. When looking at Table 6.2 it is possible to see that the Greedy Triangulation was fastest overall in all three instances with Poisson second and Grid Projection last.

Accuracy Level			
Object size	GT	GP	Poisson
Large scale	Low	Low	Medium
Medium scale	High	High	Low
Small scale	High	Medium	Medium

Table 6.4: Accuracy level based on mesh results

Also in Table 6.3 is the memory use for each of the reconstructions. The memory should be connected to the complexity since the complexity is about the elements (points) vs the number of operations it takes to reconstruct. For the house the the Grid Projection uses the least memory, which matches with the complexity, but then Greedy Triangulation is second and Poisson third, which does not match with the complexity. For both the caponier and the bunny the memory use and complexity does not match, since Greedy Triangulation takes the least amount of memory with Grid Projection second and Poisson third.




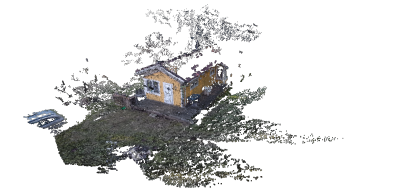
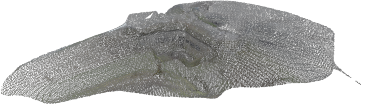


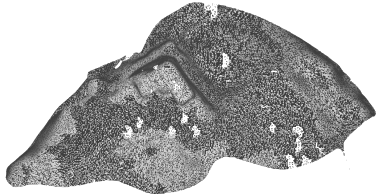

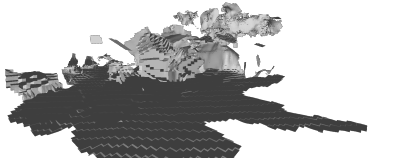
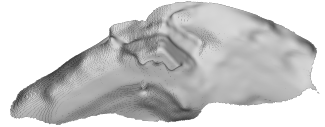

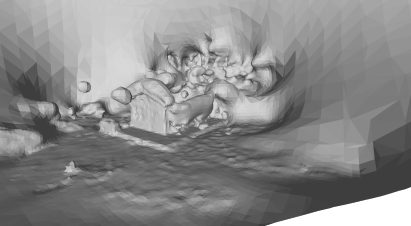


Image			
Point Cloud			
Greedy Triangulation			
Grid Projection			
Poisson			

Table 6.5: Comparison table with mesh examples.

Chapter 7

Use Case

7.1 Introduction

This chapter will include a use-case where an application for a 3D reconstruction aimed at a specific client will be implemented. The reason for having a use-case is to represent a way of how to use 3D reconstruction in real world situations. This application will be implemented and tested, however, since this is just a use-case implementation, only a small test will be conducted. The test will not be conducted with the goal of gaining any results to prove a specific theory, but instead to get an insight of whether or not this application of the 3D reconstruction can become a usable tool in the real world. The chapter will start of by introducing the client and the collaboration, followed by a design chapter in which choices and requirements from the client are discussed and fulfilled. After this, the implementation will be introduced, ending with the test and the results.

7.2 Collaboration

Mosedede Fort is an old fort built in 1913 just outside the city Greve. It was used to protect the coast during World War 1. The fort was built to prevent people getting to shore at Køge Bay and it was a part of a bigger defense up through the coast line also known as Tunestillingen. [Buus et al., 2010, p 43]. Today Mosede Fort is used as a museum and the outdoor area as a public place where people can walk around freely. The museum was selected for a field study based on feedback from fellow students who had also created a collaboration project with this museum. The field study showed that the museum focuses a lot on interactive elements at the museum, however the outdoor area was neglected and not much information regarding these buildings was given. After talking with the contact person, it was decided to have a collaboration, where some interactivity is implemented to the outdoor area. It was up to us as developers to decide what area we could implement our project with, since the museum did not have any specific ideas they wanted us to fulfill, and since the museum was interested in all types of technology, it was decided to make an app, showcasing a specific area and its story. For the area the caponier was chosen. The second time visiting the museum a more

detailed story was given about the caponier, learning about more objects that needed to be added. It was agreed that the museum would find or create the story they wanted to tell through the application.

7.3 Design

The design of all the elements later used for the animation was based on the clients wishes in this case the museum Mosede Fort and the 3D reconstructed map of the caponier. The reason the caponier was chosen as the location was due to the museum wanting some activities with their outdoor areas. The fact that it is a smaller area made it more manageable to work with regarding creating an animation in that specific area of the caponier. The story behind the caponier was also a part of why this specific area was chosen.

7.3.1 The Manuscript

The story used for this use-case should be based on guidelines given by the museum. The guidelines is used to make a first draft of a manuscript that is later to be corrected by the museum so a final one can be made, to see this manuscript look at [Appendix A](#). In the manuscript three soldiers appears throughout the storyline. The first one is an officer who gives the audience an introduction to the caponier since the manuscript is written in a way where the audience is included as a character in the story. In this case a new soldier reporting for duty at the caponier. The story continues on to the two additional soldiers who is arguing about the madsen gun which is one of the main topics the museum wanted to include in the story. The idea is to present actual historical information making people learn while watching the animation.

7.3.2 The Storyboard

Based on the manuscript, a storyboard is made and can be seen in [Figure 7.1](#). The storyboard is shown from a top view in order to get the camera angle and position included. Each image illustrates a scene and thereby also a new angle for the camera. In the actual animation the camera will be the view of the audience. The idea is to make the animation and story represent the caponier and its usage during World War 1. In [Figure 7.1a](#), [Figure 7.1b](#) and [Figure 7.1c](#) you see the camera zoom in on the first soldier who is an officer ready to welcome the audience and explain the usage of the caponier and his role. In [Figure 7.1c](#) the camera rotates showing more of the caponier. The last scene is [Figure 7.1d](#) showing a conversation between two soldiers and the audience.

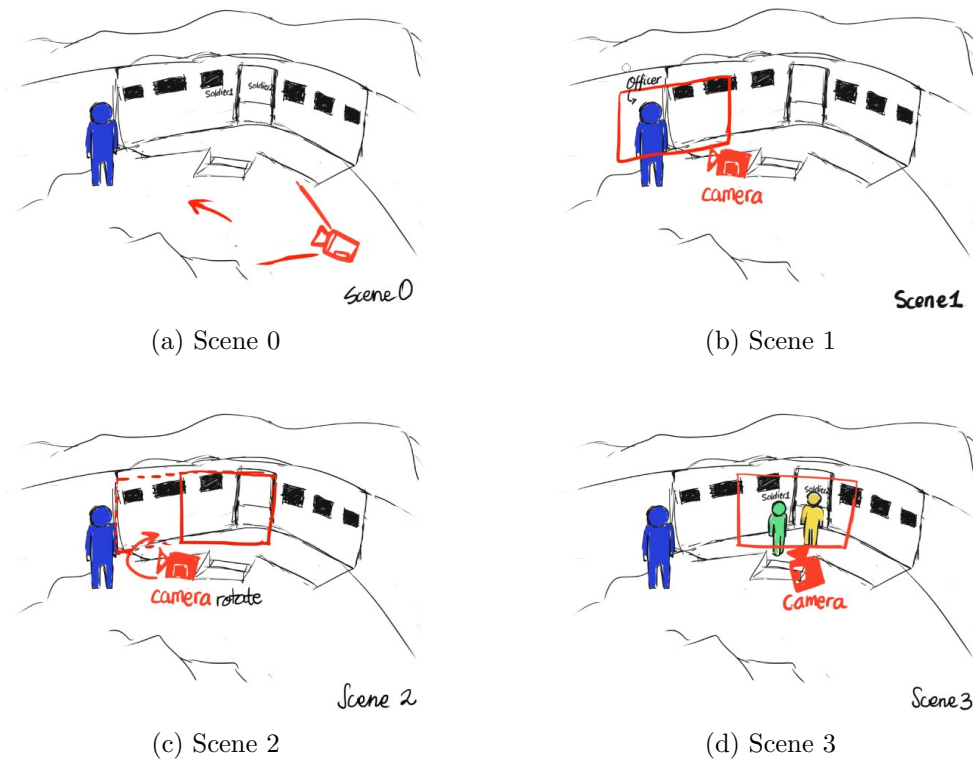


Figure 7.1: The Storyboard created for the animation seen from a top view.

7.3.3 The Models

Looking at the manuscript, the storyboard, and information learned at the field study, it is possible to determine what kind of objects should be modeled in order to make the animation. Based on the manuscript three soldiers are needed, two light-projectors for the area, uniform, accessories, and the Madsen gun. Images taken on the field study were used as references when making the models and adding colors to the material, some of the reference images can be seen in Figure 7.2. All the models are made in the 3D program Autodesk Maya 2014.



Figure 7.2: The reference images used to model the different objects and the uniform

7.4 Implementation

The implementation section is divided into two parts, the creation of the animation and the creation of the application. The part regarding the animation will contain information from how the models were created in Autodesk Maya, the rigging, and finally the actual animation. For the implementation of the application in Unity, the work progress and the code will be explained, while giving information on different features of the application and how the animation was imported.

7.4.1 The Props and Accessories

For the accessories a belt and bag, a hat, buttons, and shoes were modelled using different polygons and techniques, such as extrude, smooth, etc. The accessories can be seen in Figure 7.8 on Page 44. For the additional props a light-projector and the Madsen gun were modeled. Again using different polygons individually or attached to each other and a variety of techniques. For most of the objects, the images presented in the design chapter (see Section 7.3.3 The Models) were used as guidelines for creating them, this especially counts regarding the light projectors, the hat, and the gun. Most time was used on adding details on the gun, since this was the main prop and the object that would be in focus due to the wish from the client. The gun and the light-projector can be seen in Figure 7.3.



Figure 7.3: The additional objects created for the animation

7.4.2 From Polygons to Animation

7.4.2.1 The Models

The first parts that were modeled were the head and face. The way this was done was by creating a simple cube and add divisions to have as guide lines. Half of the cube was then deleted and instead a feature called duplicate special was used. This feature makes it possible to mirror every movement done on the other side too, not only making the workload easier but also making sure that each side does not end up looking like it belongs to two different people. Image planes were used as guidelines for the head shape, however since the image planes used were female; changes were made in order to make the head look more masculine, see Figure 7.4.

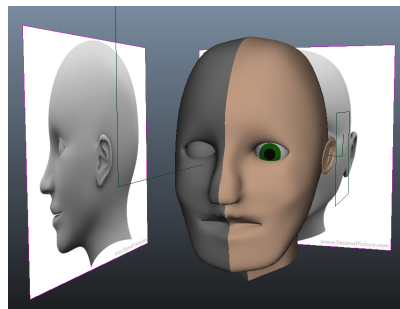


Figure 7.4: The head in progress

After shaping the actual head, details were made on the face, these features include the eyes and eyelids, nose, and mouth. The nose was made directly on the model and the eye socket, however the mouth, eyeballs, and eyelids were made separately and later on attached to the head. For the mouth it was very important that it was modeled in a way that would make it easy to rig and animate, therefore a gap between the lips was made and a black sphere inside the mouth was created. The sphere was made so when opening the mouth you would not see the back of the head.

The next step in the process was the body, creating the body, cylinder polygons were used one for the actual body and one for each leg and arm. Once again, one of the sides was deleted in order to use the duplicate special. After the upper body and arms were made and attached to each other it was time to model the legs, however even though the original idea was to use a cylinder another solution was used. The new solution was to extrude the legs from the torso. The reason a new solution was needed was because of the uniform that was modeled did not go well with the body and especially the legs. Therefore, the body faces were extruded creating a uniform shape instead of a naked body.

The last thing added as an actual part of the body was the ears which was extruded from the head faces. In the end the two body sides were combined, which means that any changes made in one of the sides would not affect the other, this step should only be made if it is certain that any changes from now on should not affect both sides. The following objects were modeled as accessories: a hat, belt, belt-bag, buttons, and shoes,

these were later attached to the soldier. After the modeling was done, color was added to the model in Maya by selecting faces and applying the material Lambert. The colors were chosen based on the decisions made in the design phase.

7.4.2.2 The Rigging

When the model was completed, it was time to create the rigging. Rigging is a skeleton making it possible to change location of different joints while creating movement. The rigging for the soldier was created based on what kind of movements it was expected of him to make, in this case he had to be able to walk a cycle and animate mouth. The most important parts were therefore the rigging of the head and the rigging of arms and legs and not the torso.

In Figure 7.5 each sphere represents a joint and the arrow shape simply shows what it is connected to. For the legs and arms IK handles were created these make constraints and make it easier to work with when animating the movements of said joints. The IK handles are the lines sticking out of the body in Figure 7.5 and Figure 7.6. Before being able to animate the models weight paint had to be added. By looking at Figure 7.6 the weight paint has been added to the leg and is represented by the white part. The weight paint makes it possible to decide how much the specific joint should affect the body, so the more intensely the white is the more it will affect the body when moving a joint. Looking at Figure 7.8, there are red and blue circles in front of the mouths and under the bodies, these are made to be able to control specific parts; the blue ones are used for the mouth movements and the red ones are used when moving the body and the rigs all at once, therefore, not leaving anything behind. After the rigging was done on the body, the body was duplicated twice. Now that three identical men were created, small features were changed with each of them making them a bit more unique and individualized, and ready to be animated.

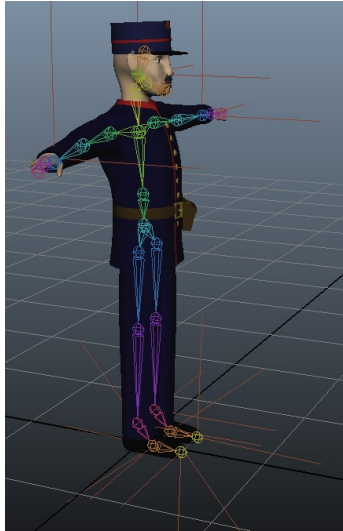


Figure 7.5: The rigging made for the animation

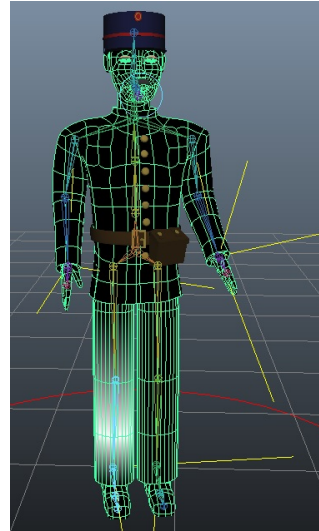


Figure 7.6: Shows the Maya technique called Weight paint

7.4.2.3 The Animation

The animation was based on the story delivered by the client in this case the museum. For the mouth animation the voice recordings were used as a guide, to make the mouth move at the correct time. For the leg and arm animation a walk cycle was made and changed a bit to make it fit the environment of the map a walk cycle can be seen in Figure 7.7.

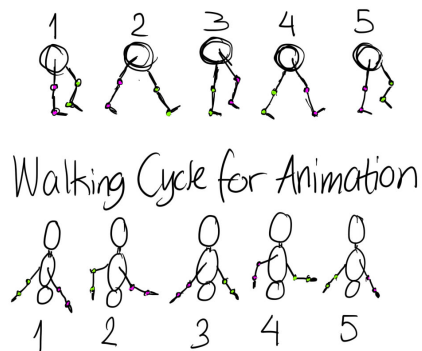


Figure 7.7: Walking Cycle for the animation

Because of the rigging, IK handles and the control cycles, it made the animation a bit more simple, it also helped that the story and the storyboard only contained a little movement. Importing the voice recording into Maya also helped a lot when making the mouth animation since it made it possible to see clearly when something was being said, an example of this can be seen in Figure 7.9 where the green waves represents the sound and the red lines the key frames.

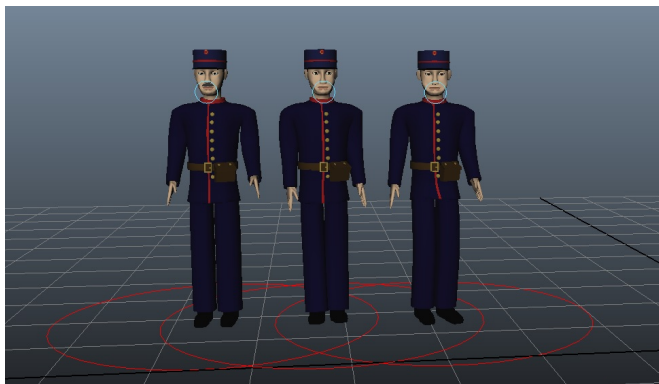


Figure 7.8: Illustrate the controllers created for the animation

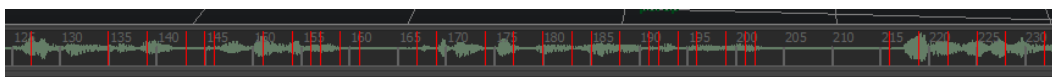


Figure 7.9: Shows the sound file when imported to Maya

7.4.3 ReCap Mesh

After performing the three reconstruction methods with the help of PCL and neither providing a detailed mesh, it was decided that for a proper test it would be ideal to use the reconstruction created by ReCap. Therefore, the model/obj file provided by Autodesk ReCap was used in the creation of the application for the museum.

Even though, the model seemed to have a higher accuracy in the recreation than the other 3 techniques used, there were still various points that needed to be taken in consideration before being added to the application. One problem encountered with the mesh was the number of faces and polygons being too high for it to be usable by Unity, the Game engine in which the application was created. Therefore a few steps were taken, which involved the usage of Autodesk Maya.

The object file from Recap was imported in Maya and with the help of the option Mesh→Reduce, it was possible to lower the amount of polygon faces used within this Mesh. Once this was done, working with the unwanted artifacts had become an easier task than before and had also allowed for Unity to handle the mesh. Figure 7.10 shows the mesh before it was retouched, the retouched version can be seen in Figure 7.11.

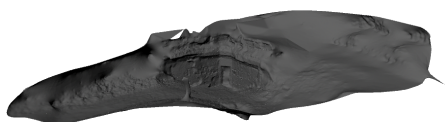


Figure 7.10: Recap Mesh

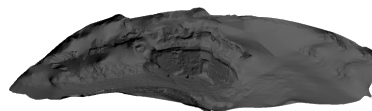


Figure 7.11: Edited Recap Mesh

It was now possible to retexture the whole map in order for it to be added to the application created in Unity. The texturing was done by selecting faces of the map. Since the faces were very small, numerous and shaped as triangles, many issues are presented with the texturing; however, it was the least time consuming approach given the conditions of the mesh. The textures were taken, as images, from the footage gathered at the museum resembling the caponier both in shape and in colors. The windows also use as texture the same images of windows of the caponier which makes the model resemble the real caponier a lot more. The map was smoothed before importing it into Unity as an .FBX file. The textured mesh can be seen in Figure 7.12 on Page 45.

7.4.4 UnityEngine Implementation

Once the map was retouched, it was imported into the game engine used, as mentioned before, Unity version 5.01 was used to create the Android application released on a tablet.

The application created within Unity allows the user to play and stop the animation regarding the historical event and preview up close the machine gun which is mentioned in the animated story. Since the application is aimed at being exported for android tablets, the user can also touch to rotate the camera allowing them to see more of the map. The user can also pinch the display to zoom on a certain area, however this is only allowed for the preview of the machine gun. An overview of what the application's layout is can be seen in Figure 7.12.

In Figure 7.13 it is possible to see the Hierarchy window inside Unity which contains all the objects within the application. This section will go into depth with the roles and functions that each object was set to perform.

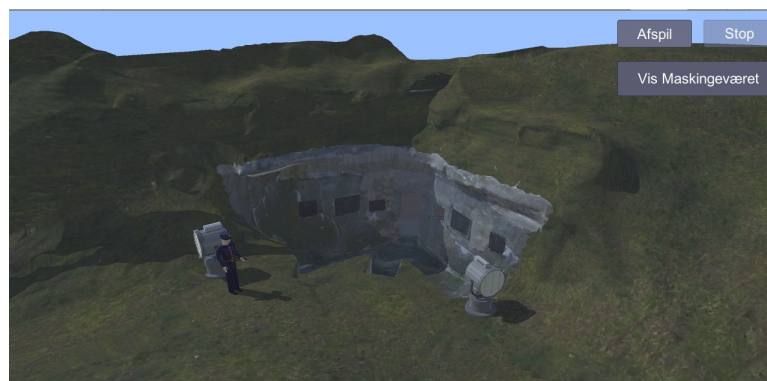


Figure 7.12: Overall Layout of the Application

7.4.4.1 3D Models and Animation

The models edited and created within Autodesk Maya were imported as .fbx files, therefore containing the textures and only requiring relinking the images of the textures to the images placed inside the assets folder in Unity. The models imported and just placed within the scene are the map (`fort`), machine gun (`Gun` and `GunInWindow`). However the `FullAnimation` contains the lights, the officer and the soldiers which were animated within Maya.

The only 3D models that have scripts attached and perform actions are `Gun` and `FullAnimation`. The object `Gun` is the object with which the user can interact with by rotating and zooming in on it. The script attached to it called `RotateObject` allows the user to touch and move finger on the tablet to rotate the object in the desired direction. The same occurs with the script attached to the main camera, which allows the user to rotate camera to the sides, however the rotation the user can perform is limited.

Code 7.1: Touch Screen Object rotation

```

1 if (Input.touchCount == 1 && Input.GetTouch (0).phase == TouchPhase.
    ↳ Moved) {
2     Vector2 touchDeltaPosition = Input.GetTouch (0).deltaPosition;
3     transform.RotateAround (this.transform.position, Vector3.up, -
    ↳ touchDeltaPosition.x * speed);
4     transform.RotateAround (this.transform.position, Vector3.down,
    ↳ touchDeltaPosition.y * speed);
5 }

```

The Code 7.1 contains an if-statement which checks whether the user is touching and moving their finger on the display and performs action according to their movement which is based on the variable `touchDeltaPosition`. This code is attached to the `Gun` object.

`FullAnimation` object contains a single script as well, which forces the app to stay awake and creates the two functions used by the stop and play buttons.

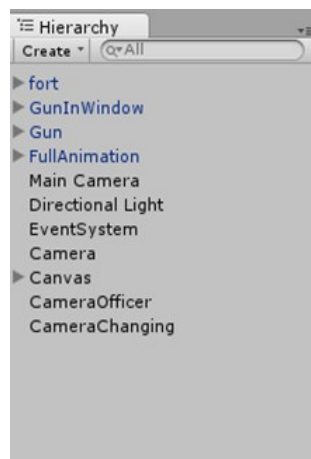


Figure 7.13: Hierarchy tab within the application scene

Code 7.2: playAnimation function for the Play button

```

1 public void playAnimation() {
2     Animation PlayIt = GetComponent<Animation>();
3     PlayIt.Play("FullAnim");
4     cam.GetComponent<Animation>().Play("cameraOfficerAnimation");
5 }

```

Code 7.1 shows the `playAnimation` function, which is very similar to the stop animation. It is in charge of playing both the Characters animation but also the Camera animation.

7.4.4.2 Cameras and Canvas

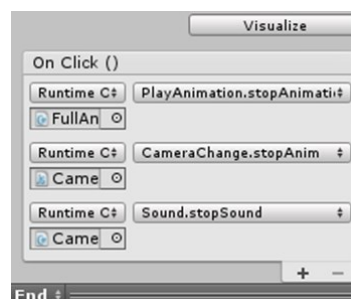
Unity allows the users to easily add buttons to an application by creating a Canvas. As mentioned before, there are three functionalities that the user will be able to perform, play an animation, stop an animation and view a close up of a machine gun; therefore three buttons are created within the canvas referred to as Play, End and Wep. The Canvas allows functionality that reacts to clicking, therefore when this is created, an `EventSystem` is created with it.

The Canvas functionality within Unity allows easy connection between functions and buttons, with already implemented functionality `OnClick()`. Figure 7.14 is an example of the `OnClick` functionality of the End button, which stops the animations of the camera, sound and the animated story.

The majority of the functionalities of the button are closely related to the cameras within the scene, in the Hierarchy tab three cameras can be seen; Main Camera, Camera and CameraOfficer. These cameras switch between an overall view of the map, the close up of the officer and the machine gun.

In order to be able to assign a function to a button, the scripts which contain the functions need to be attached to the objects within the scene. In this case, an empty object named `CameraChanging` contains the script with the camera related button functions.

What is also necessary within this application is for the cameras to change according to the button requirements, the Code 7.3 shows an example of how the camera is changed when Wep button is pressed and the weapon is showed.

Figure 7.14: `OnClick()` function of the End button

Code 7.3: Change camera function

```
1 function showWeapon () {  
2     mainCam.enabled = !mainCam.enabled;  
3     cam2.enabled = !cam2.enabled;  
4 }
```

Another functionality which improves the user's experience when using the application is the interactability of the buttons. When previewing the gun, users cannot play the animation and must first return to the main camera, therefore the buttons are not clickable when this happens. Code 7.4 shows an example of how this is coded. The `build` in function in Unity `interactable` allows for easy implementation of this functionality.

Code 7.4: Make Play button unclickable

```
1 ...  
2 else if(cameraOfficer.enabled){  
3     playButt.interactable = false;  
4     stopButt.interactable = true;  
5 }
```

Since it was not possible to import camera animations from Autodesk Maya to Unity, the camera animation within the animated story was created with Unity. Unity provides the developers with a feature *Animation* which allows the usage of keyframes and the properties of the object which are to be animated. An example of how the camera animation in Unity looks can be seen in figure 7.15.

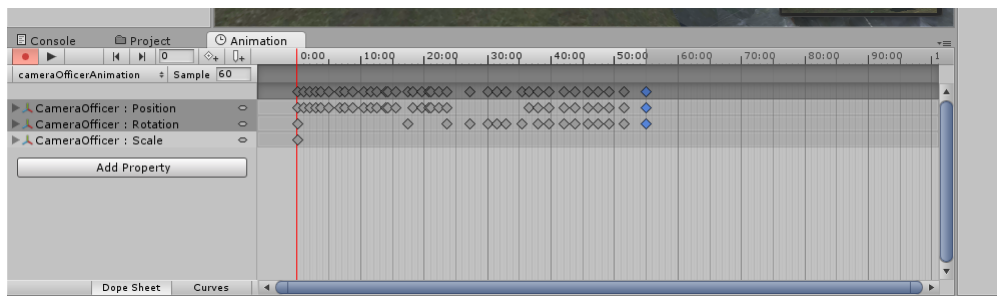


Figure 7.15: Camera animation timeline in Unity.

After the creation of the animation and its implementation in the application, it was possible to add the last part required before being complete, which is the sound. The sound is triggered by a function, whose script is added to the cameraOfficer and plays a few seconds after the camera animation has begun.

7.4.5 Sound

The sound was recorded by three different people, one for each character in the manuscript. The sound was recorded in Adobe Audition CC using a Blue Yeti microphone. After the

sound was recorded a few sound effects were added from freesound.org, using the Creative Commons license. The voices were normalized to a fitting volume level and mixed with the effects and background sounds before exported as an .mp3 file and imported into Unity.

7.5 Test

Two different tests were conducted on two different target groups and a client, overall testing on 10 participants. One taking place at Aalborg University, where five people tested out the application and were asked questions about the usage and their opinion on the 3D reconstruction. The second test taking place at Mosede Fort where five other people tested the application while standing in front of the caponier. This gave the test participants a chance to look at the real structure while using the application with the reconstructed structure.

7.5.1 Interview results

There were three different interviews conducted each with its own target group. The first is on computer science students, second group is audience at the museum, and the last one is the client. The reason why each group receives different questions is because of their skills and what is considered the most important information to be gained from the said group. For example, the students will know a lot about the technical aspects and can therefore give feedback based on that. The client will know what fits the museum and what kind of changes need to be made. Lastly, the question given to the audience is to see whether people understand the information within the application, if they know how to use it and if they learn something. All of the interviews were conducted semi-structured, therefore, each interview has a specific set of guidelines. In the following part, each interview will be shortly introduced together with the results gained from the interview.

7.5.1.1 The Computer Science Students

Focus points:

- Their feedback on the application and its usage
- Their feedback on the 3D reconstruction

Therefore the following questions were made as guidelines: *Do you think having the caponier 3D constructed, is a better way of approaching it compared to having, for example, a model created in 3ds Max or Maya?*

Two said that they would prefer a model, since the 3D reconstruction was not smooth enough where two others stated that they thought that it made it look more realistic compared to a freehand model, one of them did however state that is was not important whether or not it was 3D reconstructed or simple modeled.

Do you think this is a good way to showcase an outdoor area at a museum and would it be worth it to extend it further, having more buildings where the app could give information?

Two of them sounded very certain in their case since it would be considered more interesting to have this interactive element. The third participant agreed, but that was based on thinking that without it, the area would seem a bit left behind. Regarding the extension it would depend on how much time the audience would want to spent at a museum. The fourth participant stated that there should either be more interactivity in the application or there should just be signs.

7.5.1.2 The Audience

Focus points:

- Did they know how to use it
- Did they get anything out of it
- Would they want to use this if available at a museum

The audience was asked about their understanding of the story and the majority of them seemed to remember most of the important information the application presented. When the group was asked whether they know what the caponier is used for they were not completely sure, but had an idea about it, the employee knew before the animation what the caponier was used for during World War 1.

When presented with the question; *Is this a good way to show an outdoor area at a museum?*, both the group of 3 and the employee here responded yes, one of them did however give the comment that the light might be an issue with the tablet screen outside.

General comments

The groups believes there should still be a focus on the actual building and not the application. One of them prefer to hear the information and therefore found this way of presenting information suitable. Even though previously stating that the tablet might take over the experience for kids, it is also believed that this kind of technology could attract kids to go to museums. Overall, the audience agrees that it would be a positive thing for the museum to have such an application.

7.5.1.3 The Client

Focus points:

- Did it represent the caponier correctly?
- What does he like and what does he want to change?
- Would he implement the system at the museum?

The client believes that the caponier was correctly represented in the application.

At the question *What kind of improvements would you like the application to have to work better with the museum?*, the client wishes for it to be possible to interact with the soldiers in the animation and be able to walk around in the 3D area. Another point would be adding old and historical images to the application so it is also a form of a photo album. The caponier should not look like it does today, but instead how it looked back then. Since they talk a lot about the gun in the animation he would have liked for the gun to be showcased more in the animation.

General comments

The advantage of 3D should be used more; show more angles especially of the caponier. He really likes the feature which allows a closer look at the gun. The client believes that this could be a useful application at the museum.

7.6 Sub-Conclusion

Overall, the Use Case chapter was aimed at fulfilling the clients needs and also applying the reconstructed objects focused on throughout the project in a useful scenario. Together with the museum, it was possible to come up with an interactive application for the reconstruction involving animations. During the field study at the museum, pictures of relevant objects to the application were taken and with the help of the pictures it was possible to closely recreate the objects from World War I. Some problems were also encountered, such as the texture of the map, which might have caused some bias regarding the realistic representation of the caponiere. Since the application revolves around a big area, there were many points that could have been taken in consideration when creating the application. A few of the improvements were mentioned by the client in the interview, however this could be regarded to as future aspect points.

This page was intentionally left blank.

Chapter 8

Discussion

The investigation has led to the possibility to narrow down the IPS to the FPS. Throughout the investigation we researched into ways that makes it possible to use video footage to create 3D reconstruction of objects and buildings encountered throughout the world. It was also discovered that drones can be used to take video footage of large scale outdoor environments, buildings, nature sights, etc. This information had led to the FPS which revolves around implementing different reconstruction techniques for large scale outdoor environments while using the help of a drone. It was also decided that our findings regarding the combination of a drone and 3D reconstruction should be implemented in a specific use case, which in this case was an application for a museum and thereby allowing us to experience how a collaboration with a client is. Firstly it can be argued whether the research that was presented was sufficient to be able to base the information within the comparison section on. The research regarding PCL has been of great use when working with the three reconstruction techniques, however, it was not possible to find more information on Grid Projection. The theory described does not contain as much information as needed and therefore it can be stated that it was not possible to make a valid assumption on the accuracy levels of the reconstructions.

In general, the FPS is fulfilled in regards to applying different reconstruction techniques and comparing these to check which would be the most optimal reconstruction method to use for different sized objects. In this case, a comparison between three sized objects is made, large, medium and small. The size of the objects was classified both regarding the size of the actual object and the amount of data points the point cloud for each specific object use. One thing that can be pointed out regarding the point cloud gathered, was the fact that the point cloud for the house had not been very proper. It could have been that the images might not have been properly taken from all angles of the house using the drone and therefore the pcl could not get enough information on the roof and the side of the house. This could have been solved by circling the drone multiple times around the house at different angles to gather more data to be used for the point cloud creation.

By comparing the assessment table described in [6.1 Image Acquisition](#) with the results presented in the [6.3 Comparison](#) section, it can be evaluated whether the assess-

ments made based on the theory were correct. The personal judgements made based on the results led only to two correct assessments, which were the GT reconstruction of the large and small scaled objects it does however require that the normals are afterwards reversed to face in the right direction. It should be mentioned that the reason to why these assumptions cannot be trusted is due to the high influence human error has on these assessments, which are made based on personal observations. Another error that could have biased the results and the assumptions made based on these was due to the fact that the pictures taken could not have been taken exactly from the same angles when rotating around all of the three objects. It could be solved, in a future scenario case where these reconstructions could be tested once more, by programming the drone to fly at the same heights and guide it to circle around the buildings while maintaining that height for all the objects that would be reconstructed and compared. As mentioned before, the FPS also presents the challenge of finding and implementing an application for a 3D reconstruction. Mosede Fort is known to be open to interactive applications and the use of new technology, therefore collaboration with this museum was decided.

The report presents the collaboration with the museum through the chapter 7 Use Case which is a thorough description of the important parts of the application and how this relates to the museums needs. As the use case presents, one way to use 3D reconstructions is by creating an application usable by a museum and implementing an animation to tell a story with it. The main focus of this application was to implement the best reconstructed caponier of the three implemented reconstruction methods while using the PCL, however due to the low quality and the improper mesh representations that was achieved, it was not possible to use them for creating an application for the museum. Due to said reason it was decided to use the best represented mesh obtained from ReCap. Using ReCap also provided textures with the mesh, however it was not possible to use the textures given by ReCap within Unity. The problem was solved by manually colouring each face of the mesh, causing it to look less detailed and showing flaws, a solution to this could have been finding a different approach to create the application, a program perhaps as Unreal Engine might allow importing the mesh together with the texture provided by Recap.

When the application was tested it had all the necessary functions implemented, however we did not manage to implement all the discussed features before the test, such as a proper animation of the camera, allowing user to interact with the gun by touching the object in the scene this was due to time limit and priority of the use case. During the test, the majority of the participants and the client had valuable inputs for future aspects regarding the application. Some of these points were for example allowing the user to scroll through an album of pictures of World War I objects, people and places. Even though the client gave various of ideas on how the application could be improved, he believes that this application would be usable in the museum and that we fulfilled our expectations as developers for the client by delivering a working application which implements a reconstructed outdoor environment, the caponier. Having a base for this

application working in the future it should be fairly easy to continue developing it and expanding it to other areas at the museum as well as completely other locations.

Overall, there are various applicable implementations for the 3D reconstruction. As this report also presents with the use case, the 3D reconstruction can be used to showcase buildings and outdoor environments in an application, allowing users to zoom in on certain areas and rotate the meshes. Another implementation would be for animation purposes which could allow a faster approach at animating a story.

This page was intentionally left blank.

Chapter 9

Conclusion

The research in the investigation offered the needed information in order to narrow down and find a Final Problem Statement:

What would the ideal reconstruction technique for big scale outdoor environments be when using a drone to gather point cloud data and what possible application can a reconstruction of this be used for?

This gave way to the focus point being based on using the drone and 3D reconstructing an outdoor area. The second approach of the FPS gave us the opportunity to collaborate with Mosede Fort and create a useful application using the knowledge gathered in Method and Implementation.

During the Methods chapter, theory regarding different reconstruction techniques was gathered. Some of these techniques were Greedy Triangulation, Grid Projection and Poisson reconstruction which are also implemented in the point cloud library used to perform the reconstructions. By gathering information regarding the three methods it was possible to assume their reconstruction accuracy based on the size of the objects to reconstruct and compare them to each other. The implementation was based on the theory described in Methods, going in depth with the code that was written to reconstruct the three objects; the house, the caponier and the Stanford bunny. We reconstructed the three objects using the three reconstruction techniques and then compared them to one another based on their accuracy level of the reconstruction, their performance time and computational complexity. This comparison was made in order to see which of the techniques would be most optimal to perform for three different object sizes, in this case a small, medium, and large scale objects.

In the Use Case chapter an application using a 3D reconstruction was implemented based on a field study to Mosede Fort where references images were gathered and a location was decided upon. By modelling the necessary props for an animation and a story to be implemented in an application it was possible to fulfill the clients needs of presenting the caponier. The Use Case chapter has proven that 3D reconstruction can be used for applications such as presenting a historical building or environment.

Based on the three reconstruction methods accuracy level, time, computational complexity and memory it takes up, it can be assessed that the Greedy Triangulation presents

the most positive features for both the small scaled objects and the medium scaled objects, however in the case of the large scale object Grid Projection seems to be most optimal. Therefore, based on the reconstruction method being applied using PCL it can be assessed that Greedy Triangulation would provide the most optimal mesh when working with small and medium sized objects. Large objects reconstructed using GP would prove to be less accurate and more time consuming however with a lower computational complexity and memory usage, to get an accurate and fast reconstruction GT would be most optimal.

Appendix A

The Manuscript

OFFICEREN

Velkommen soldat, vi har ventet dig.

Betonbygningen her er en kaponiere. Her er du i sikkerhed, hvis fortet bliver angrebet. Fra kaponieren skal vi beskytte de tørre voldgrave.

Din rolle er at stå bag Madsen-geværet, som er placeret i det midterste vindue. Din opgave er at skyde, hvis fjenden trænger ind på fortet.

Du er på vagt sammen med 11 andre mand. I er en livsvigtig del af fortets forsvar. Jeg stoler på at du kan klare opgaven. Dine kammeraters liv og landets fremtid afhænger af dig.

OFFICEREN

Nu må have mig undskyldt, jeg er ventet andre steder, men jeg er sikker på at de andre vil give dig en varm modtagelse.

SOLDAT 1

Goddag, du måvære ham den nye. Du skal være en del af vores gruppe.

SOLDAT 2

Jeg høre at du er den heldige vinder af Madsen-gevær positionen. Jeg ville ønske det var min position. Det et helt fantastisk gevær. Der findes ikke noget som det noget andet sted i verden og det er ovenikøbet opfundet af en dansker.

SOLDAT 1

Du skal ikke lytte til ham, han er i familie med producenten af geværet, generalmajor W.H.O. Madsen.

SOLDAT 2

Det har intet med det at gøre. Madsen-geværet er et rekylgevær. Det er en let form for maskingevær. Det særlige er, at geværet er så let, at en person kan betjene det alene. Andre maskingeværer kræver 3-4 mand.

De krigsførende lande har også opdaget, hvor genial en opfindelse det er. Jeg har hørt at flere lande er interesseret i at købe det... Og prøv at tænke på alle de penge man kunne tjene.

SOLDAT 1

Du ved udmærket godt at man ikke må sælge våben som neutralt land. Hvis vi nu sælger det til England så bliver Tyskerne sure og omvendt. Nej som neutralt land er det bedst at holde sig neutral. Vi skulle nødig risikere at vores handel går i stå fordi vi handler med ulovlige varer.

SOLDAT 2

Men...

ALARM!!!

OFFICEREN

ALLE MND TIL DERES POSITIONER!

Bibliography

- M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):3–15, 2003. ISSN 10772626. doi: 10.1109/TVCG.2003.1175093.
- Autodesk. Autodesk 123d catch. <http://www.123dapp.com/catch>, 2014. [Accessed 2015-05-24].
- Autodesk. Recap reality capture. <https://recap.autodesk.com/>, 2015. [Accessed 2015-03-08].
- M. Berger, P. Alliez, A. Tagliasacchi, L. M. Seversky, C. T. Silva, J. a. Levine, and A. Sharf. State of the Art in Surface Reconstruction from Point Clouds. *Proceedings of the Eurographics 2014, Eurographics STARs*, pages 161–185, 2014. doi: 10.2312/egst.20141040. URL <http://lgg.epfl.ch/reconstar>.
- H. Buus et al. *Frste verdenskrig ved Tunestillingen Forsvarsvilje og hverdagsliv*. Greve Museum, 2010. ISBN 978-87-89367-32-4.
- F. Cazals and J. Giesen. Delaunay triangulation based surface reconstruction: Ideas and algorithms. In *EFFECTIVE COMPUTATIONAL GEOMETRY FOR CURVES AND SURFACES*, pages 231–273. Springer, 2006.
- L. Chen, S. Betschart, and A. Blaylock. Projeto redentor: Modeling rio de janeiro’s christ the redeemer in high-resolution 3d. <https://pix4d.com/mapping-christ/>, 2015. [Accessed 2015-05-24].
- P. Corto. Dji phantom. http://4.bp.blogspot.com/-jI6_HmEUkr8/UkqnsW-Ic9I/AAAAAAAAABR8/nPq1DKWr71k/s1600/maxresdefault.jpg, 2013. [Accessed 2015-05-27].
- M. T. Dickerson, R. L. S. Drysdale, S. A. McElfresh, and E. Welzl. Fast greedy triangulation algorithms. In *PROC. 10TH ANN. SYMP. COMPUTATIONAL GEOMETRY*, pages 211–220, 1994.
- J. Digne. Géométrie inverse : du nuage de points brut à la surface 3D Théorie et Algorithmes Inverse Geometry : from the Raw Point Cloud to the 3D Surface Theory and Algorithms. pages 1–201, 2010.

- Drone 55, Inc. Drone aerial cinematography and photography. <http://www.drone55.com/>, 2015. [Accessed 2015-05-24].
- R. Fabio. From point cloud to surface: the modeling and visualization problem. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34:W10, 2003.
- D. Girardeau-Montaut. Cloudcompare. <http://www.danielgm.net/cc/>, 2013. [Accessed 2015-03-19].
- S. A. Goldman. A space efficient greedy triangulation algorithm. *Information Processing Letters*, 31(4):191 – 196, 1989. ISSN 0020-0190. doi: [http://dx.doi.org/10.1016/0020-0190\(89\)90122-1](http://dx.doi.org/10.1016/0020-0190(89)90122-1). URL <http://www.sciencedirect.com/science/article/pii/0020019089901221>.
- Google. Atap project tango. <https://www.google.com/atap/projecttango/?ref=producthunt#project>. [Accessed 2015-05-20].
- M. Gurman. Apple acquired mind-blowing 3d mapping company c3 technologies, looking to take ios maps to the next level. <http://9to5mac.com/2011/10/29/apple-acquired-mind-blowing-3d-mapping-company-c3-technologies-looking-to-take-ios-maps-to-the-next-level/>, 2011. [Accessed 2015-03-22].
- J. Hyvärinen. Surface Reconstruction of Point Clouds Captures with Microsoft Kinect. 2012.
- S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. ACM Symposium on User Interface Software and Technology, October 2011. URL <http://research.microsoft.com/apps/pubs/default.aspx?id=155416>.
- T. Jebara, A. Azarbayejani, and A. Pentland. 3d structure from 2d motion. *Signal Processing Magazine, IEEE*, 16(3):66–84, May 1999. ISSN 1053-5888. doi: 10.1109/79.768574.
- M. Kazhdan and H. Hoppe. Screened poisson surface reconstruction. *ACM Trans. Graph.*, 32(3):29:1–29:13, July 2013. ISSN 0730-0301. doi: 10.1145/2487228.2487237. URL <http://doi.acm.org/10.1145/2487228.2487237>.
- M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction, 2006. URL <http://dx.doi.org/10.2312/SGP/SGP06/061-070>.
- C. Levkopoulos and A. Lingas. Fast algorithms for greedy triangulation. *BIT Numerical Mathematics*, 32(2):280–296, 1992. ISSN 0006-3835. doi: 10.1007/BF01994882. URL <http://dx.doi.org/10.1007/BF01994882>.

- R. Li, L. Liu, L. Phan, S. Abeysinghe, C. Grimm, and T. Ju. Polygonizing extremal surfaces with manifold guarantees. In *Proceedings of the 14th ACM Symposium on Solid and Physical Modeling, SPM '10*, pages 189–194, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-984-8. doi: 10.1145/1839778.1839808. URL <http://doi.acm.org/10.1145/1839778.1839808>.
- L. Ma, T. Whelan, E. Bondarev, P. H. N. De With, and J. McDonald. Planar simplification and texturing of dense point cloud maps. *2013 European Conference on Mobile Robots, ECMR 2013 - Conference Proceedings*, pages 164–171, 2013. doi: 10.1109/ECMR.2013.6698837.
- G. Medioni, M.-S. Lee, and C.-K. Tang. *A Computational Framework for Segmentation and Grouping*. [ISBN: 978-0-444-50353-4].
- F. Mémoli and G. Sapiro. A theoretical and computational framework for isometry invariant recognition of point cloud data. *Foundations of Computational Mathematics*, 5:313–347, 2005. ISSN 16153375. doi: 10.1007/s10208-004-0145-y.
- Meshlab. Meshlab. <http://meshlab.sourceforge.net>, 2014. [Accessed 2015-03-19].
- R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE ISMAR*. IEEE, October 2011. URL <http://research.microsoft.com/apps/pubs/default.aspx?id=155378>.
- OctoFilms. Octofilms aerial cinematography. <http://octofilms.com/>, 2014. [Accessed 2015-05-24].
- S. Peterson. Computing constrained delaunay triangulations. http://www.geom.uiuc.edu/~samuelp/del_project.html. [Accessed 2015-05-20].
- PointClouds.org. Concatenate the fields of two point clouds. http://pointclouds.org/documentation/tutorials/concatenate_fields.php. [Accessed 2015-05-24].
- PointClouds.org. How to use a kdtree to search. http://pointclouds.org/documentation/tutorials/kdtree_search.php#kdtree-search, 2014a. [Accessed 2015-05-15].
- PointClouds.org. Pcl - point cloud library. <http://pointclouds.org>, 2014b. [Accessed 2015-05-15].
- R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1–4, 2011. ISSN 10504729. doi: 10.1109/ICRA.2011.5980567.

- A. Saxena, M. Sun, and A. Ng. Learning 3-d scene structure from a single still image. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, Oct 2007a. doi: 10.1109/ICCV.2007.4408828.
- A. Saxena, M. Sun, and A. Y. Ng. 3-d reconstruction from sparse views using monocular vision, 2007b. [Accessed 2015-05-20].
- Stanford University Computer Graphics Laboratory. The stanford 3d scanning repository. <https://graphics.stanford.edu/data/3Dscanrep/>, 1994. [Accessed 2015-05-15].
- Visualization Toolkit. vtksmoothpolydatafilter class reference. <http://www.vtk.org/doc/nightly/html/classvtkSmoothPolyDataFilter.html#details>, 2015. [Accessed 2015-04-29].
- C. Wu. Visualsfm : A visual structure from motion system. <http://ccwu.me/vsfm/>, 2011. [Accessed 2015-03-19].
- Zuse Institute Berlin. 3d reconstruction of anatomical structures from 2d x-ray images. <http://www.zib.de/projects/3d-reconstruction-anatomical-structures-2d-x-ray-images>, 2006-2015. [Accessed 2015-05-20].