University of Cape Town

Masters Dissertation

# Creating 3D Models of Cultural Heritage Sites with Terrestrial Laser Scanning and 3D Imaging

Christoph Held

2012



*Supervisors*

Prof. Dr. Heinz Rüther
University of Cape Town

Prof. Dr. Julian Smit
University of Cape Town

# Abstract

The advent of terrestrial laser-scanners made the digital preservation of cultural heritage sites an affordable technique to produce accurate and detailed 3D-computer-model representations for any kind of 3D-objects, such as buildings, infrastructure, and even entire landscapes. However, one of the key issues with this technique is the large amount of recorded points; a problem which was even more intensified by the recent advances in laser-scanning technology, which increased the data acquisition rate from 25 thousand to 1 million points per second.

The following research presents a workflow for the processing of large-volume laser-scanning data, with a special focus on the needs of the Zamani initiative. The research project, based at the University of Cape Town, spatially documents African Cultural Heritage sites and Landscapes and produces meshed 3D models, of various, historically important objects, such as fortresses, mosques, churches, castles, palaces, rock art shelters, statues, stelae and even landscapes.

To produce a 3D model, the laser scanning data needs to be processed in various steps: cleaning, registration, meshing, hole filling, simplification and texturing. For the creation of a generic workflow, flexible enough to adapt to a large variety of input datasets, the available literature on the key processing steps is reviewed intensively and the most standard approaches for the individual tasks, as well as recent developments proposed by the research community, are presented to the reader.

The processing with available tools, including the key parameters, is analysed and discussed, and the most effective workflow is presented. Also file converters are produced to achieve compatibility between various software tools and formats and for automatic dataset subdivision. In addition, further ideas for the optimization of time-intensive manual tasks are explored, such as the incorporation of SFM-tools into various steps of the pipeline. The presented workflow is flexible, adaptive and independent of the data volume and was verified on a dataset of the Zamani Project, consisting of 720 million points.

# Acknowledgements

This research was hosted by the Zamani Project. The research unit is based at the University of Cape Town and is made possible by the generous support of Andrew W. Mellon Foundation, New York. The author would like to express his sincere gratitude to the principal investigator of the project and supervisor of this research, Prof. Dr. Heinz Rüther, the second supervisor Prof. Dr. Julian Smit, as well as to the co-workers at the Zamani project, Roshan Bhurtha, Ralph Schröder and Stephen Wessels. The author would like to also thank the researchers of the Visual Computing Laboratory (VCG) at the ISTI-CNR, Italy for producing and sharing a large number of the software employed in here. Further gratitude goes to the Stanford University and Alexander Agathos for their work on the software Scanalyze, as well as Noah Snavely for his insights on the software Bundler.

# Plagiarism Declaration

I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own.

Signature removed

Cape Town, May 26, 2012

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

> [...]Cultural heritage is the things, places and practices that define who we are as individuals, as communities, as nations or civilisations and as a species. It is that which we want to keep, share and pass on. *Donald Horne Institute for Cultural Heritage, University of Canberra*[1]

According to this definition, there are special places, buildings, squares or landscapes that are an integral part of our cultural heritage. During the course of history, many of these places were destroyed or simply collapsed. Hence, the documentation of cultural heritage sites, which still exist today, is essential to the definition of future generations' identity. It is thus the aim of this research to explore ways to create computer generated, highly detailed three dimensional(3D) models to document these sites as accurately, detailed and efficiently as possible with today's technologies.

Laser scanning is one of several possible techniques for acquiring precise, centimetre accurate virtual representations of real-world objects. A laser scanner accurately records the position of a surface point in 3D space. One can categorize laser scanning in many different ways. One common criterion is the scanning distance: close (<10m), medium (10m<x<300m) and long range (>300m). While close range scanners are often employed for statues and smaller artefacts and can provide sub-millimetre accuracy, medium and long range scanners are used to scan larger structures and landscapes. One can also distinguish between terrestrial and airborne systems, between the type of range measurement, such as phase/pulse measurements and optical triangulation, and between standard and mobile scanning. This research focuses on terrestrial laser-scanning(TLS).

A laser-scanner records millions of surface points within minutes. The growing amount of point data with each new generation of these instruments is able to capture,

[1] `http://www.canberra.edu.au/centres/donald-horne/cultural-heritage/what-is`

requires the constant improvement of the processing pipeline used to convert these points into 3D models. While not even ten years ago, 20 million points were considered a vast amount of data for a completed project, today's machines can acquire 600 million points and more in less than 30 minutes. However, developments in software have not advanced at the same rate, most likely because the community using these techniques is still comparably small.

Architects and archaeologists for example, still prefer using simple drawings (computer assisted or hand-drawn) showing only little detail. Objects are usually described through geometric primitives, for example, a wall is often only represented by four corner points and four connecting lines. However, this is not because more detail would not be of benefit, but rather because the creation of an accurate, highly detailed computer model of a real-world object still requires considerably more time and is thus very expensive to produce.

The production of a 3D computer model from laser-scanning involves several, non-trivial tasks and it is the intention of this research to explore the various solutions used today for each step of the production pipeline and combine the most effective ones to create an efficient workflow from the acquisition of the data to the final model. This processing pipeline should be universal and applicable to different objects in the cultural heritage field, independent of their nature.



## 1.1 Background

The author of this research is a team member of the Zamani research group, based at the University of Cape Town. The Zamani project, also known as the "African Cultural Heritage Sites and Landscapes Project" is a non-profit initiative, founded and lead by Prof Heinz Rüther and funded by the Andrew W. Mellon foundation. The objective of the project is to document cultural heritage sites and landscapes in Africa with state-of-the-art techniques in order to create a record for future generations (Rüther, 2002).

Many heritage sites in Africa are badly neglected and in danger of deterioration. The Zamani project not only aims at the creation of heritage information for future generations, but also hopes to support the work of researchers, conservationists, archaeologists and architects (Rüther et al., 2009). To achieve this goal, the project uses precise and well established conventional survey techniques, as well as laser-scanning, photogrammetry, and remote sensing. It produces 3D-models, Geographic Information Systems (GIS), 360° full dome panoramas, sections and plans, and videos.

The data is integrated into a scientific database produced by its partner, Aluka[2], New York, which was recently merged into JSTOR[3], New York. Aluka collects historic documents and combines them with the visual data, generated by the Zamani team. The data is accessible at no cost to African institutions, such as the antiquities departments of the respective countries, universities and other educational or heritage related institutions. To date, over 40 sites with more than 80 individual objects in eleven African countries were documented, including churches, mosques, temples, fortresses, castles, palaces, houses, rock-art shelters, stelae and entire cultural landscapes.

This research focuses onto the specific needs and requirements of the project. One of the key problems experienced within the project is the large data volume, which increased from 20 million points to 7.4 billion points per site, since the beginning of the project in 2004. Also special attention is paid to the flexibility of the established workflow, allowing for objects of various natures.

## 1.2 Existing 3D Modelling Workflows

Creating 3D models from point cloud data is not a new topic and has been researched for the past 15-20 years. While all sub-steps of the pipeline are still active fields of research, a natural standard sequence of processes (fig.1.1) evolved: scan alignment or registration, data cleaning, surface reconstruction, hole filling or mesh repair, simplification, and colouring, also referred to as texturing. The order in which some of these steps are applied sometimes change slightly for different projects, depending on the deliverables. The "cleaning" of the data was often neglected or not mentioned in publications, most likely because many research projects worked under (near) laboratory conditions.

One of the most seminal publications on the topic was the "Digital Michelangelo Project" by Levoy et al. (2000). The authors proposed and demonstrated ways to process large datasets and produced some open-source tools for scan alignment, scan merging and hole filling. Another well know example is the publication of Debevec et al. (2004), who recorded the Parthenon in Athens, Greece and set high standards for documentation projects regarding realistic rendering of a 3D model.

Data processing of cultural heritage sites differs largely from workflows in industrial disciplines, such as the construction sector, industrial plant surveys, or the mining industry, where the point cloud is approximated by best-fitting geometric primitives, such as planes, spheres and cylinders, reducing the amount of detail to a minimum.

[2] http://www.aluka.org
[3] http://www.jstor.org

In the context of documenting cultural heritage however, detail in the final model is regarded highly important and thus other techniques need be employed, which make use of every scanned surface point to preserve as much detail as possible. For this reason workflows and techniques, which drastically simplify the data are neglected within this research.

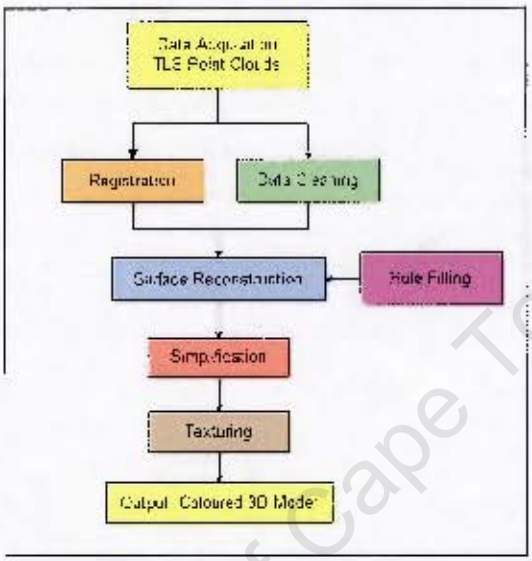


Figure 1.1: The laser-scanning pipeline



## 1.3 Research Aims

### 1.3.1 Primary Objective

The primary objective of this research is to find a generic, efficient workflow for creating a coloured, high-resolution 3D-surface-model using TLS point clouds, with a special focus on the requirements of the Zamani project. The Zamani project documents structures of varying sizes and complexity, resulting in a wide variety of problems, which is why the workflow, established during this research, will need to be designed in a flexible format, making it independent of the objects nature, whether it is a small statue or large structures such as a mosque, church, fortress, ruin, or even an entire landscape. This also requires finding alternatives, in case the proposed methods fail.

One of the major challenges is the amount of points to be processed, which will stress most algorithms. In the past, the technical progress of laser scanning instruments caused an increase in the amount of data being recorded, by one or even two orders of magnitude. About ten years ago, 40 million points were regarded as a large

dataset, while at present, four to seven billion points per site are not unusual any more. The large data volume is specifically problematic in environments with standard hard- and software, which is not designed for advanced processing with high memory consumptions.

### 1.3.2 Research Questions

From this objective, the following research questions can be derived:

- How to create a 3D-model independent of the input data volume?
- How to design a workflow which is flexible enough to be applied to different objects of various natures?
- How to optimize or even automate the workflow, so that manual intervention is minimized?

### 1.3.3 Outline of the Work

To answer these questions, this research begins with a literature review of seminal approaches for the main steps of the processing pipeline: registration, surface reconstruction, simplification, hole filling and texturing.

Afterwards, each step of the pipeline is examined to establish the most efficient solution with available tools. The workflow is designed with a view to preferably use freely available software wherever possible. Large support, in the form of free software for the documentation of cultural heritage sites, can be found within the research community. As standard commercial software used in the industrial environment is usually very expensive, it seems to be of great benefit to the cultural heritage community, which is often a not-for-profit business, to employ free software. However, as free software is just a preference within this research, commercial software packages will also be considered, if a license is available to the Zamani project. If necessary, tools (for example to convert data between the different formats used by the various software packages) are produced or adapted, if the source code is available.

After presenting the recommendations for each processing step, the established, entire workflow is demonstrated on a dataset of one of the Zamani heritage sites.

## 1.4 Definitions

Laser-scanning is at present primarily a surveying technique, but also an extensively studied topic within the computer science community. Thus, in the following, a few

definitions are given of commonly used terms for the processing of laser-scans.

### 1.4.1 Laser Scanning

#### 1.4.1.1 Scan vs. Range Image

A laser scan is the sampling of a 3D surface. A laser scanner performs all measurements at regular vertical and horizontal angle intervals. A scan, as the output of a measurement task, can thus also be understood as a grid or a matrix.

#### 1.4.1.2 Sub-Scan

A sub-scan is an additional scan without moving the scanner. Hence, all points are within the same coordinate system as the main scan. A sub-scan is used to scan a part of a surface or a target in a different resolution than the rest of the structure to save time or to re-scan a part which has previously been occluded by moving objects, such as cars or people.

#### 1.4.1.3 Range Classification

In Laser-scanning, one distinguishes between three range-classes. However, there is no formal or standard definition and usages of the terms can vary from manufacturer to manufacturer. Within this research, the terms are used as the following:

- $< 10m$ - close range
- $10m < X < 300m$ - medium range
- $> 300m$ - long range

### 1.4.2 Computer Geometry

- **3D Model** - In computer vision applications, a 3D model usually refers to a continuous virtual representation of an object and can be represented by different geometry, mostly by NURB-splines or polygons. A set of connected polygons form a **mesh** and thus, a **3D model** is also often referred to as a **meshed model**. This research is focusing on meshes, formed by triangles, also referred to as faces, as they are simple and efficient to handle and are still the standard for surface representation in computer graphics applications.
- **Quads** are polygons, consisting of four vertices and are often used as an alternative to triangles. They become more and more popular, but up to now, triangles do have wider support in software.

- **Vertex** refers to a corner point of a polygon/triangle. In a mesh, vertices are connected with **edges**.
- A **Point Cloud** is a set of vertices in a 3D coordinate system.
- **Point Set** is similar to a point cloud, but it most commonly refers to a collection of points used as a data input for surface reconstruction and is usually enriched with additional information, such as point normals.
- **Normals** - a normal is a line perpendicular to a plane and usually refers in here to a normal of a triangle, but can also be used in relation with points. A point- or vertex normal is defined indirectly by averaging the surface normals of all triangles incident on the vertex.
- An **indexed mesh** is a mesh, where the triangles are defined by three indices, referring to a list of vertices.
- **Triangle soup** refers to a collection of triangles without any connectivity information, such as shared vertices of neighbouring triangles. It is the most basic triangle representation, but at the expense of storage efficiency as vertex duplicates do exist.
- **2.5D Model** - as opposed to 3D models, a 2.5D model refers to a surface model which, for each x- and y-value only allows one z-value. This representation is widely used for digital terrain or elevation models (DTM/DEM) and allows for simpler surface reconstruction methods. This research focuses on the creation of true 3D models.
- **Manifoldness** - "[...]an important topological quality of a surface is whether or not it is *2-manifold* (short for *two dimensional manifold*), which is the case if, for each point, the surface is locally homeomorphic to a disk (or a half-disk at boundaries)" (Botsch et al., 2010, p.11). According to Botsch et al. (2010, pp.11-12), manifold surfaces cannot contain intersecting triangles, edges with more than two triangles connected to it, or single vertices connecting two otherwise independent surface patches (non-manifold vertex). Manifoldness is often a requirement for surface processing algorithms.

### 1.4.3 Specifications for the Output Model

The found workflow needs to be able to produce a 3D model from point data of a building with the following specifications.

The model is desired to be

- in 2-3 cm resolution, or at least should be visually similar to a model of this resolution. It should be possible to produce models in even higher resolution, at least of sub-areas.
- available in several other, lower resolutions to accommodate computer system with slow performance
- free of artefacts
- manifold and have a consistent orientation of it's surface normals
- in colour, based on photographs
- reliable, which means that no data shall be added which is not based on measurements, unless if specifically requested

Even though space is a limiting factor, the Zamani Project intends to keep the 3D models in an uncompressed file format for faster access. In some cases, such as Point clouds, the files should even kept in ASCII format to make sure the files can be read in future. Thus compression was not considered for this workflow.

# Chapter 2

# Literature Review and Theoretical Background



In the following chapter, the most seminal and also some innovative concepts for each part of the pipeline are presented in a comprehensive literature review. This, however, cannot be a complete survey of all available techniques, as each of the sub-tasks can be a research on its own. The following chapter shall thus serve as an introduction into the various methods being discussed later, and, in addition, provide the reader with an impression of where the current developments in research are heading towards.

## 2.1 Data Sources

At the beginning of each data processing workflow is the acquisition of the data and thus, a brief introduction is given into the various laser scanning techniques, as well as image-based techniques.

### 2.1.1 Laser Scanning Techniques

#### 2.1.1.1 Time of Flight

An often used technical principle to determine the distance from a survey instrument to the surface of interest is the "time-of-flight" or "pulse" measurement. A laser pulse is emitted and the time it takes to be reflected by the surface and returned to the scanner is measured. As the speed of light is known[1], the distance can be calculated (fig. 2.1a). Current, medium-range pulse scanners can detect points over a range of 300

[1]The speed of light depends on atmospheric conditions, which can, and often are, taken into account by laser scanner manufacturers

m and achieve a scanning rate of 50,000 points per second (e.g. Leica HDS ScanStation C10). There are also systems which are optimized for much longer ranges, up to several kilometres, but at the expense of accuracy, as the footprint of the laser beam becomes much larger. In general, specifications of the manufacturers should be treated with care, as no standard is used in the industry. Especially accuracy of distance is difficult to determine and largely depends on the reflectivity of the surface. A scanner can maybe measure a single point over 150 m, but that does not mean that it will be able to receive and correctly interpret the signal of all other points with different reflectivity at the same distance. Boehler et al. (2003) proposed a test system for the comparison of laser-scanners, but still no comparable standard is used in the industry.

#### 2.1.1.2 Phase Measurement

In contrast to pulse scanners, "phase-shift" instruments detect the difference in the phase of the signal at the time of transmission and return. Because of the wave ambiguity, the maximum range is limited to half of the wave length (fig. 2.1b). To extend the range, several signals of different wavelengths are modulated onto each other (Thiel and Wehr, 2004). Today's phase-based instruments can scan up to 1,000,000 points/s over a range of up to 180 m (e.g. Z&F Imager 7000). As above, the specifications should be treated with reservations.

#### 2.1.1.3 Triangulation or Structured Light Scanners

Triangulation or structured light scanner work with optical measurements and are thus also related to photogrammetry. A laser or a projector in combination with one or two cameras are positioned and oriented in a fixed and known location. The camera, which is calibrated according to photogrammetric principles (see below), determines the position of a fixed 2D pattern in its field of view, or the position of a dot or a line, which is swept in a constant speed across the surface. Via triangulation principles, 3D point coordinates can be determined. (fig. 2.1c). These types of scanners are very efficient for close range ($< 10m$) purposes, but the accuracy drops with the distance (Boehler and Marbs, 2002). The baseline between camera and laser cannot be freely adjusted and the resolution of the camera sensor is fixed, which sets limits on accuracy and range.

### 2.1.2 Photogrammetry

Photogrammetry, the science of retrieving metric information about an object from images, is another data acquisition method. As opposed to laser-scanning, Photogram-

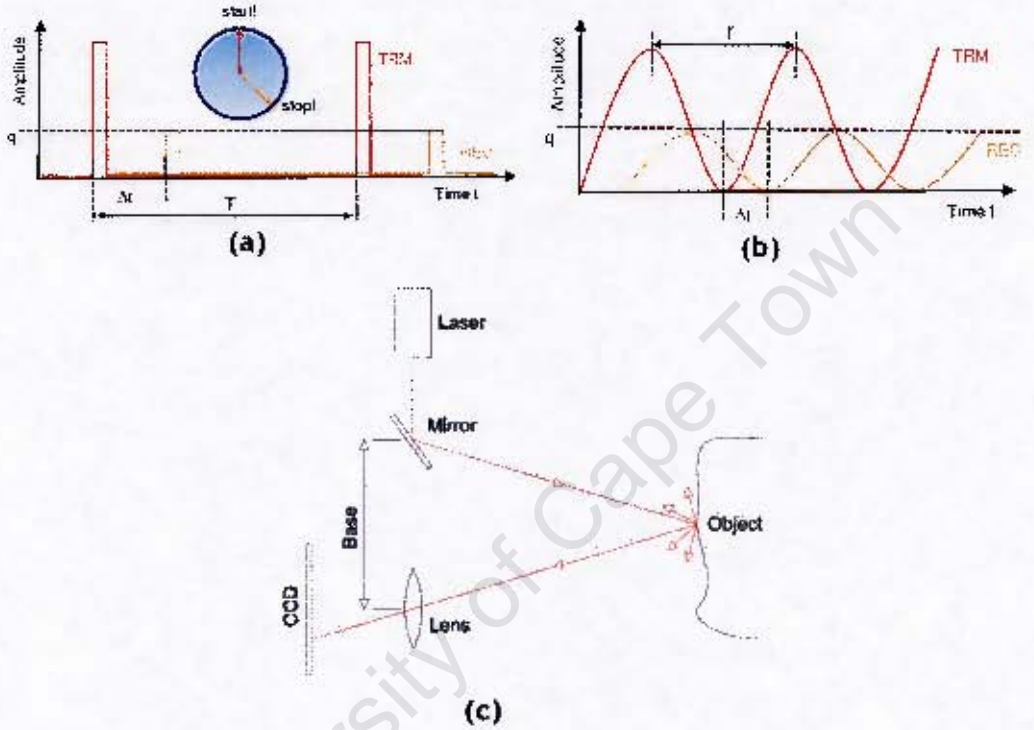


Figure 2.1: (a) "Time-of-flight" or "pulse" distance measuring method. A pulse signal TRM is sent out and the time difference to the return signal REC is being measured. The theoretical maximum distance of this method is determined by the time interval T between two successive pulses. (b) "Phase-Shift" distance measuring method, measuring the phase difference of a (constant) outgoing wave signal TRM of wavelength T and the return signal REC. Images from (Zoller+Fröhlich-GmbH, 2011). (c) Optical triangulation system, single camera solution. A laser point or line is deflected by a mirror and swept across the object and recorded by a camera. Image from (Boehler and Marbs, 2002)

metry is a passive sensor method. The camera just receives light and does not emit any, apart from a flash. For high accuracy, the cameras have to be calibrated, meaning that the cameras' internal orientation parameters have to be known for all images. These include the focal length, or to be more precise, principal distance[2], the principal point and the lens distortion parameters. The location and orientation of the camera in space are defined as the external parameters. For a field campaign, the cameras can be either pre- or post calibrated on a set of targets, or directly on the final photographs via feature points on the object, although the latter option usually results in less accurate calibrations.

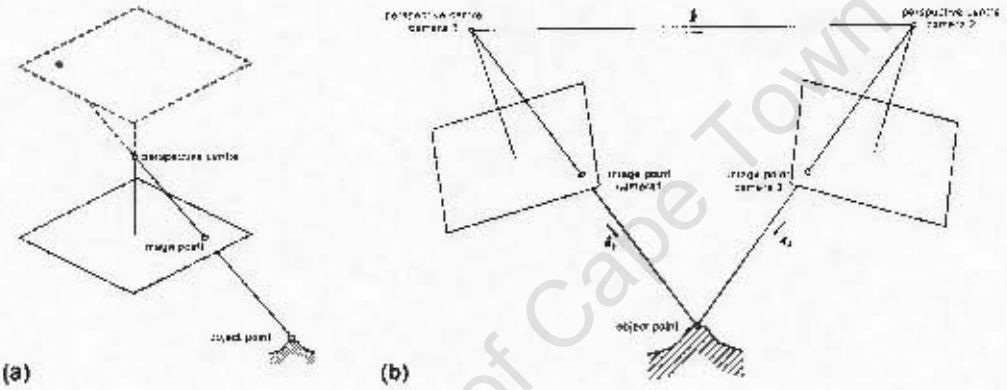


Figure 2.2: Co-linearity equation (a), co-planarity equation (b). Images adapted from (Mikhail et al., 2001)

Two basic principles of Photogrammetry to determine 3D point coordinates are the co-planarity and the co-linearity condition. The latter states, that a line from the perspective centre of an oriented camera through a feature point on the image plane will pass the corresponding point on the object (fig. 2.2a). "The co-planarity condition [...] implies that the two perspective centres, any object point and the corresponding image points on the two photographs of the stereo-pair, must all lie in a common plane." (Ghosh, 2005, pp.110-111) (fig. 2.2b)

Once the cameras' internal and external orientations are known, a dense 3D point cloud can be recovered via Stereo or Multi-View-Stereo algorithms. Theoretically, point data acquired with photogrammetric methods could be processed with the same workflow described herein, but the primary focus will be on data acquired by laser-scanning, with its specific problems and challenges. However, Photogrammetry is relevant here, as problems can often be overcome by the combination with a different

[2]In the following, these two terms are used interchangeably

technique.

As opposed to laser scanning, where points are recorded indiscriminately and a point is rarely hit twice from different positions, Photogrammetry is able to measure specific interest points, such as edges or features. But also automatic point cloud extraction from photographs is quite advanced, but as the images need to be processed first before results can be seen, a successful measurement of all points is not as guaranteed as during laser-scanning. At least, laser scanning enables checking the data easily while on site.

#### 2.1.2.1 Structure from Motion

Although not a new concept, the recent advance of computer performance and the advent of cloud computing pushed the research in the field of Structure from Motion (SfM)[3]. Projects such as Bundler (Snavely et al., 2006), VisualSfM[4], SfM-Toolkit[5], Agisoft Photoscan[6], Photosynth[7], My3DScanner[8] and Autodesk 123D Catch[9] made it a very popular concept to automatically derive a sparse 3D point cloud of an object from an image sequence with a large base to object distance ratio. A small ratio produces a large angle at the point of interest between rays from the cameras at each end of the baseline (fig. 2.3a). The larger these angles become the more accurate the results of the point determination will be and also the recovering of the camera parameters. However, with increasing angles, the similarity of features between the images decreases and thus also the chances for automatic matching are lowered. Thus, SfM-tools require a large base to object distance ratio, producing small angles, to be able to exploit a large correlation between successive images for automatic feature matching (fig. 2.3b).

Snavely et al. (2006) demonstrate, with their Photo-tourism approach, a workflow for the easy navigation through a large set of images, even in an unordered sequence (fig. 2.4). To estimate internal and external camera orientation parameters, all images are analysed with the "scale-invariant feature transform" (SIFT) operator (Lowe, 2004), a standard feature detection method. These are compared and matched with features from all other images, followed by an estimation of the camera's external and internal orientation via the direct linear transform (DLT) and a subsequent bundle

[3]Also often referred to as Structure and Motion (SaM)
[4]http://www.cs.washington.edu/homes/ccwu/vsfm/
[5]http://www.visual-experiments.com/demos/sfmtoolkit/
[6]http://www.agisoft.ru
[7]http://www.photosynth.net
[8]http://www.my3dscanner.com/
[9]http://www.123dapp.com/catch

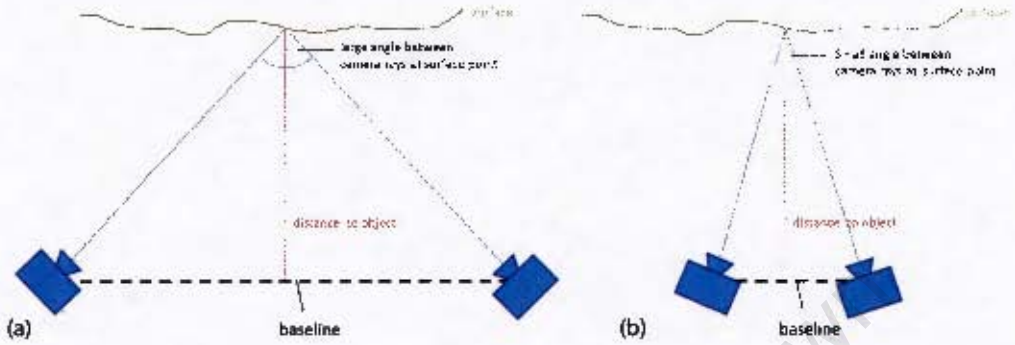


Figure 2.3: Base to Object Distance Ratio. (a) a small ratio produces large angles of camera rays at the surface. (b) example of a large ratio




Figure 2.4: Structure From Motion: the Photo Tourism Project. A large collection of images (a) is automatically matched. 3D surface points and camera positions are recovered (b) to allow a visual and intuitive navigation in 3D through the collection(c). Images from (Snavely et al., 2006)

adjustment. To avoid converging to local minima, images are added one by one to the process. The source code of the optimization software was released to the public under the name Bundler, together with a collection of other tools and scripts. The success of Bundler is a result of its undemanding design, as it works on un-calibrated images. Only a rough initial focal length is required, which can be extracted from the EXIF headers, present in most images originating from a digital camera. SfM is also used for internet-based collections of photographs: (Agarwal et al., 2011) report the estimation of 250.000 camera positions on a multi-core system.

Main Problems with Bundler are speed, as all features from all images are compared against each other, and its sensitivity to the selection of the initial views, to which all others are registered. By standard the views with the most matches are selected, which does not mean that all views can be well registered to it. Another algorithm called Samantha (Farenzena et al., 2009; Gherardi et al., 2010) suggested the sorting of the images into a hierarchical tree, a dendrogram, spatially sorting the closest views, which reduces the computational complexity by an order of magnitude.

Bundler produces only a very sparse point cloud, consisting of the matched features. Bundler's main purpose is the estimation of camera parameters, which can be used in a following (multi-view) stereo process to reconstruct dense point clouds or surface representations of the photographed object. Open source tools such as Patch-based Multi-view Stereo Software (PMVS)(Furukawa and Ponce, 2007) in connection with the Clustering Views for Multi-view Stereo (CMVS) algorithm (Furukawa et al., 2010) allow the reconstruction of very dense point clouds from large image sets. More algorithms are compared by Seitz et al. (2006) and updated results are published online under `http://vision.middlebury.edu/mview/eval/`.

Even though the results produced by Bundler etc., do look very impressive, the accuracy is not sufficient for photogrammetric surveys as reported by Barazzetti et al. (2010). To achieve more accurate results, the authors suggest employing only calibrated cameras. They also extend the process of feature matching by a least-squares matching (LSM) process (Gruen, 1985) and suggest to run another feature detection and LSM process, once a rough bundle adjustment was completed. For the additional feature detection step, they suggest to use the FAST-operator (an edge detection operator), as it finds the same features in more images. More observations of the same point will increase the accuracy, as outliers can be filtered better. A final bundle adjustment improves the orientation parameters and similar accuracy to a manual observation is achieved.

## 2.2 Scan Alignment

As terrestrial laser-scanners are based on emitting light and receiving its reflection, they can only record points which are in a direct line-of-sight to the instrument. Hence, to fully capture the object and obtain a complete 3D digital reproduction, it is necessary to scan the object from different perspectives. This results in multiple, independent scans, each in their own coordinate space. Uniting them in one common coordinate system is generally referred to as 'registration' or 'alignment' and can be achieved by either using targets or by matching surface features, provided that the scans are overlapping in some parts.

### 2.2.1 Targets

Using targets is most likely the fastest method to align several scans, at least regarding the time spent during post-processing. Some customized targets, with special reflectance properties or shapes, can be identified automatically in a point cloud, by interpreting the intensity of the laser beam reflection and/or by fitting primitives, such as spheres, to potential candidates. Different patterns were developed, which make it possible to determine the centre of the target with millimetre accuracy, even though the very centre has not been hit directly by the laser beam. As opposed to the large number of points used in surface matching, there are usually only very few targets (<10) per scan and the registration can be done quickly and automatically by simply evaluating all different combinations to find the best-fit. The method is known to be precise. Errors are introduced by inaccurate determination of the target's centre, which can occur, due to under-sampling, a bad angle of incidence of the target with respect to the instrument, or movement of the target with respect to the object.

But even though this method is saving processing time in the office, it might not always be the optimal choice, as using targets in the field can complicate matters. To be able to establish a correct 3D transformation, it is required to observe at least two targets, if the instrument was levelled precisely at every setup or another constraint is available. In general however, it is recommended to also observe some more targets (4-5) to account for shifted targets, imprecise levelling, bad inclination angle towards the targets, low reflectivity or simply human error. To be of any use, these targets also need to be recorded from another scan position.

Further, to minimize registration error, the targets should be placed far apart from each other, which is not always practical, depending on the surrounding environment. This is especially true for scanning the interior of a structure, when scanning small rooms which need to be connected through long narrow passages or through only little

door openings. Very often, only a couple of suitable target positions are accessible in- and outside the building, which limits the space where the instrument can be set up. Instead of choosing the best position for the instrument with respect to good inclination angles from the scanner to the surface and a complete coverage of the object, the operator is forced to choose positions from where at least two targets can be seen. This makes the approach inflexible.

It is also important to note, that high precision targets are usually bulky and heavy, and the shipment is thus expensive. There are light and small targets available which can be attached to a wall, but this is a very questionable approach, as cultural heritage sites, especially in Africa, are very delicate and fragile and could be damaged by the adhesive. Moreover, the targets themselves will obscure part of the surface, if no other secondary, solid objects are available nearby.

### 2.2.2 Surface Matching

#### 2.2.2.1 Pairwise Scan Registration

Provided that sufficient overlap between scans is given, an alternative approach to targets is matching surfaces based on closest points within the overlapping areas. The most popular solution for this problem is called the ICP, Iterative Closest Point, introduced by Besl and McKay (1992); Chen and Medioni (1992). The iterative algorithm is designed to perform a pairwise registration of two overlapping scans. For each point on the first scan, Besl and McKay (1992) determine a partner point on the second scan (point-to-point, fig. 2.5a). A rigid body transformation is then determined to minimize the distance between common points in the least-squares sense. The process is repeated until convergence. Instead of determining the closest points on both scans, Chen and Medioni (1992) suggest to shoot a ray from a point on the first scan along its surface normal and minimize the distance between the starting point and the tangent plane at the intersection of the ray with the second scan (point-to-plane, fig. 2.5b).

Since its introduction, many variants of this algorithm were proposed, too many to be discussed here in detail. The interested reader is referred to a survey by Rusinkiewicz and Levoy (2001), who examine the various approaches and try to find the optimal solution for each of the various stages of the ICP:

- **Selection of Points** - How many samples are used for the process and how they are selected
- **Identifying partners** - The method of establishing the partner point/tangent plane on the second scan

- **Weighting schemes** - The criteria for assigning weights to the couplets, such as distance, colour or instrument noise
- **Rejecting false couplets** - The criteria to filter out false pairs
- **Assigning an error metric** - The measure for the distance error between both scans, such as point-to-point or point-to-plane
- **Minimizing the error metric** - Methods for fast minimization

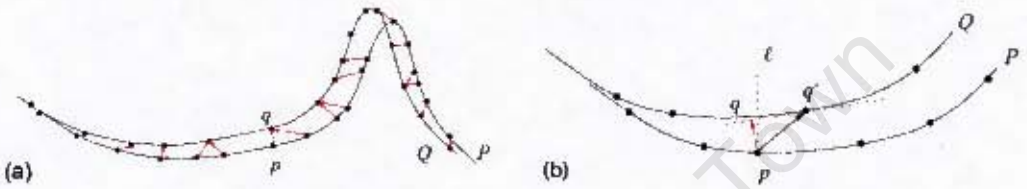


Figure 2.5: (a) Point-to-Point error metric, which minimizes the distance between closest points p and q on two scans P and Q. As only scanned points are considered as potential partners, not points on the edges in between, points of scan Q can be partners to several points on P. (b) Point-to-Plane error metric. A line l at a point p on scan P, parallel to the surface normal, intersects surface Q at a point q. A tangent plane is fitted to the nearest vertex (q') to q. The distance between p and the tangent plane is then minimized. Image from Bernadini and Rushmeier (2002)

The choice of error metric of the ICP variant appears to have the largest influence on speed and the robustness to noise. The most common error metrics are the sum of the squared distances between point pairs (point-to-point (Besl and McKay, 1992)), or between a point and a tangent plane attached to its corresponding point (point-to-plane (Chen and Medioni, 1992)). Both of them have their strengths. Point-to-point appears to be more robust to noise and easier to combine with other attributes like colour and intensity, but point-to-plane usually reaches convergence usually much faster and is less susceptible to converge only to local minima (Nishino and Ikeuchi, 2002). According to Pulli (1999), the point-to-point error metric can cause the scans to converge only slowly, when point pairs, which are often not ideal mates, are very close to each other. According to Chen and Medioni (1992) and confirmed by Pulli (1999), the point-to-plane method can converge much faster, as the metric allows the one point to slide during minimization on the tangent plane of its partner point on the other surface.

In general, it is the ICP algorithm's greatest disadvantage that it is not guaranteed to find the global, as opposed to local minima, meaning the perfect match. Hence, the

initial position of both surfaces is most important and needs to be as good as possible to avoid a convergence to a local minima. Thus, a pre-processing step becomes necessary to approximately pre-align the two surfaces, a process, which, up-to-now, still heavily relies on manual intervention.

### 2.2.3 Pre-alignment

To manually establish an initial scan alignment, registration software offers two common methods: selecting several (>3) corresponding points on both scans by hand or by manually translating and rotating the scans into position by inspection. Secondary data, like scanner position, orientation or the path along which the scanner was moved during the campaign, can provide useful information.

#### 2.2.3.1 Planar Feature Extraction and Matching

While manual, interactive registration is still the standard approach, several automatic solutions were proposed. Allen et al. (2003) try to extract 3D-line features from scans by fitting planes and using their border-lines and intersections with other planar segments for feature matching to locate corresponding areas on different scans (fig. 2.6. Instead of comparing each scan with every other scan, they provide a list of possible pairings, based on the path along which the scanner was set-up.

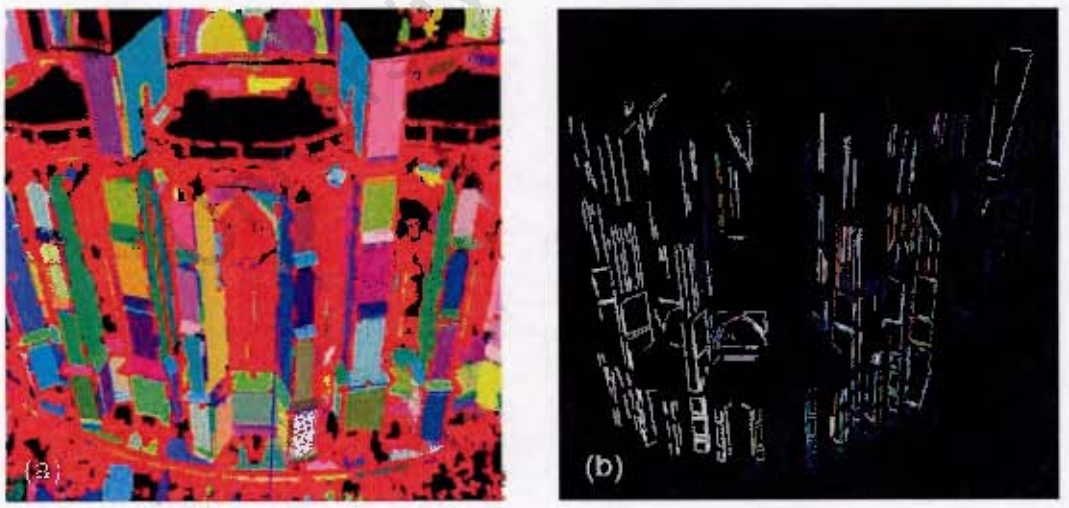


Figure 2.6: Automatic registration according to Allen et al. (2003). (a) Model segmentation and planar feature detection. Non-planar areas are marked in red. (b) Extracted line features. Blue and white colours mark different scans; matched features are coloured in red and green. Images from Allen et al. (2003)

#### 2.2.3.2 Integral Volume Descriptor

A more general approach was presented by Gelfand et al. (2005) who introduce an integral volume descriptor. This descriptor is a quantity describing a point and the surface in its proximity. Here, a sphere of specific size is centred at each point and the intersected volume between the interior of the surface and the sphere is determined to be used as a descriptor (see fig. 2.7). The points with the most extreme descriptor values are used as feature points and compared to other scans. This method proves to be quite robust against noise and outliers.

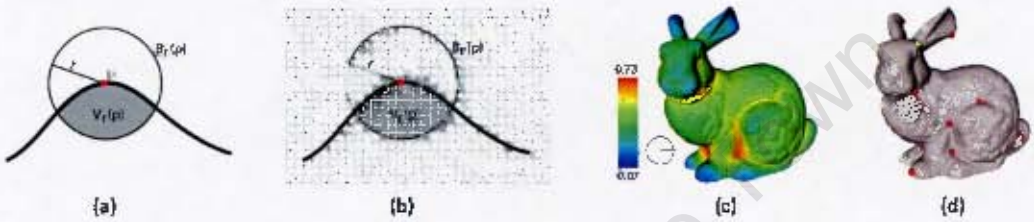


Figure 2.7: Volume Descriptor. (a): A ball B of radius r is centred at each surface point p. The intersected volume $V_r(p)$ of the ball and the interior of the surface is determined to serve as a descriptor. (b) For further processing the volume is discretised (c): Descriptor values for the Stanford Bunny model. (d) Extreme descriptor values are used as feature points for scan matching. Images from Gelfand et al. (2005)

#### 2.2.3.3 Spin-Images

Johnson and Hebert (1997) went a different way with spin-images. For each point of the surface, a vertex normal is established by fitting a 2D plane to the point and its neighbours. Each plane serves as a base for one spin-image, which is created by mapping all other data points onto a 2D image according to, firstly, the distance of a point and its projection along the surface normal onto the base ($\beta$, positive and negative values) and, secondly, the distance from the base point to the projection ($\alpha$, only positive values) (see fig. 2.8). The result can be seen as an image, which is anchored to the vertex normal and rotating around it, creating a map of all other point positions by marking their 'collision' position while rotating around the vertex normal. Hence the name spin-images (fig. 2.9). The approach opens up the world of 2D image processing to 3D scan registration. Images from both scans are compared and a match leads to a transformation aligning the two scans, which can then be refined with ICP. For more efficient performance the images are reduced in resolution.

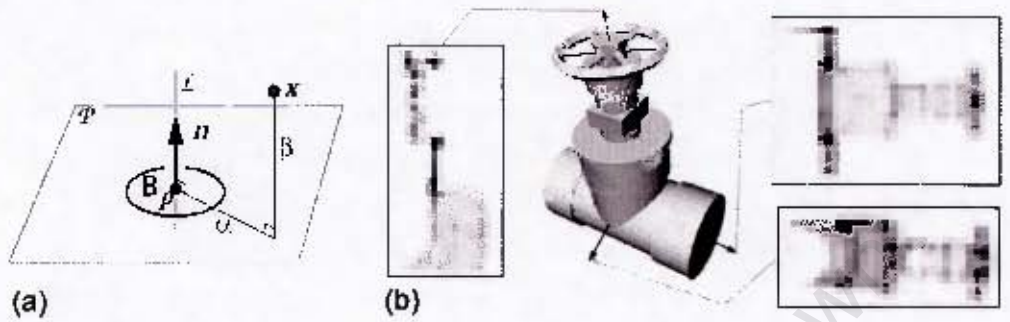


Figure 2.8: Spin Images. (a) a 2D plane P is fitted to each surface point p. A plane with surface normal n serves as a base for a Spin-Image. By parametrisation of all surface points according to the distance to their projection along the surface normal onto the base $\alpha$ and the height over their projection $\beta$, a 2D image can be created, a spin-image (b). Image from Johnson and Hebert (1997)



Figure 2.9: Spin-images can be visualized by a plane 'spinning' around the vertex normal and 'collecting' the other vertices on its way. Image adapted from Johnson (1997)

#### 2.2.3.4 4-Points Congruent Sets

Another promising idea was suggested by Aiger et al. (2008), called "4-Points Congruent Sets" (4PCS). Essentially, three points on one surface are randomly selected. A fourth point is then chosen so that the set of points form a wide, almost coplanar base. The ratios between coplanar 4-points sets are invariant to a rigid body transformation and thus approximately congruent. A wide base guarantees stability, but if it becomes too wide, it can fall outside the overlapping area of the scans. When all sets are extracted, the ratios are compared. If a match is found, the points of the sets can be used as feature points to establish a 3D transformation. The approach seems to be very robust against noise and outliers and limited overlap, but according to the authors the runtime is not proportional to the size of the input dataset, making it less practical for large datasets.

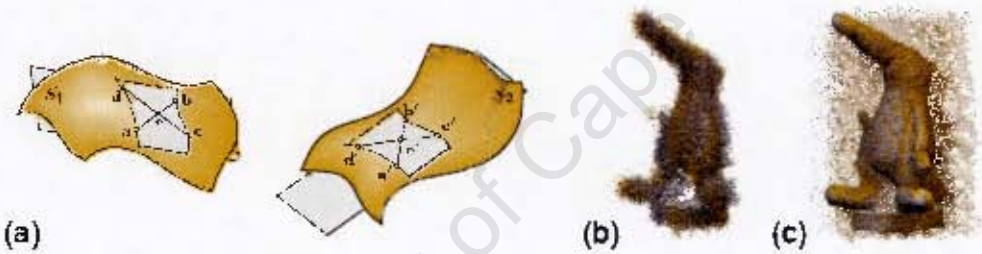


Figure 2.10: 4-Points Congruent Sets (4PCS). (a): Sets of almost coplanar points are established on both scans. The ratios $a-e/a-b$ and $c-e/c-d$ are invariant even to affine transformations, which is an optional feature of the method, not required for laser-scanning. (b) and (c) show matching result with 4PCS of two scans (orange and grey points) corrupted by noise (b) and outliers (c). Image from Aiger et al. (2008)

### 2.2.4 Global Registration

As mentioned above, the ICP aligns the surfaces pairwise, two scans at a time. During each alignment an error is introduced, which is unavoidable, since no two scans are identical, due to the discreteness of a scan and also instrument noise. Even though the error might only be relatively small, it will propagate through the registration sequence and combine with errors in subsequent alignments, reducing the overall registration accuracy. To reduce the accumulated error, one can, after all common pairs are identified and pre-registered, register all overlapping scans simultaneously, thus distributing errors among all scans.

Pulli (1999) proposes to perform the pairwise alignment as good as possible and to store all corresponding points between scan-pairs in a separate file, avoiding the need for loading all scans into memory and another time-intensive search for closest points, very important factors when processing large datasets. In the following step, scans (represented by just the externally saved point matches) are added incrementally from a 'dormant' to an 'active' set, starting with the scan which has the most connections. The added scan is registered to its neighbours in the active set. Should the position of its neighbours vary too much during this registration step, they are also re-registered to their neighbours until the movement falls below a tolerance. This procedure does not improve major misalignments in the pairwise registration, but rather distributes the error.

It is interesting to note, that during global registration, Pulli (1999) and Nishino and Ikeuchi (2002) prefer point-to-point error metric over point-to-plane. While it is of advantage during the initial alignment to have one scan "sliding" faster across the second scan with point-to-plane, during global registration the characteristic of point-to-point of being more robust to noise seems to be more important (see page 18). Also the likelihood of the point-to-point method to find only a local minimum is reduced, because all pairs were well aligned already in the previous stage.

## 2.3 Meshing Algorithms

After the alignment of all laser scans, the point dataset, a discrete data type, can be converted into a polygonal model, a (mostly) continuous representation. Several algorithms are available for this step. The following focuses on the standard approaches, especially on those which have the ability to cope with huge datasets.

Surface reconstruction has been a very active research topic in the past years, and there are numerous ways to classify the proposed methods, for example based on the required input data. Some approaches, work on individual scans, which often need to be triangulated before they can be processed. Other methods can extract the surface directly from a single, unstructured point cloud, but only a few of them do not need additional information such as vertex normals.

One can also classify the reconstruction methods depending on the handling of the input samples. Function fitting approaches try to find a mathematical function fitting best to the input samples, either with a single global- or several local functions. These functions are approximating the surface. Other algorithms like the ones based on Delaunay/Voronoi are directly connecting all, or, a filtered subset of the input points to triangles or polygons (see fig. 2.11).


Figure 2.11: (a) A surface is created by directly connecting the input data samples. (b) A surface is approximated by fitting a mathematical function to the input point set.



| | Handling of Input Samples | | Input Data Structure | |
|---|---|---|---|---|
| | Direct Triangulation | Approximation | Individual Scans | Unstructured Point Cloud |
| Delaunay and Voronoi | x | | | x |
| Power Crust | x | | | x |
| Ball Pivoting Pivoting | x | | | x |
| Zipper | x | | x | |
| Volumetric Integration | | x | x | |
| SR Unorganized Points | | x | | x |
| Moving Squares | | x | | x |
| Poisson | | x | | x |

Table 2.1: Overview over the surface reconstruction techniques described in this chapter

### 2.3.1 Surface Reconstruction by Direct Triangulation of Input Data

#### 2.3.1.1 Delaunay and Voronoi

To explain the Voronoi and Delaunay diagram, the definition will be given for 2D space, as it is easier to visualize. According to (Edelsbrunner, 2000, p.4) the Voronoi diagram decomposes the space, defined by a set of points **S**, into regions of influence for each sample point **p**: "...the *Voronoi region* of $\mathbf{p} \in \mathbf{S}$ [,is defined] as the set of points $\mathbf{x} \in \mathbb{R}^2$ that are at least as close to **p** as to any other point in **S**..."(Edelsbrunner, 2000, p.4)

Or, in other words, all points of a Voronoi region have to be at the same or a smaller distance to the sample point than to sample points of the neighbouring areas. Two neighbouring Voronoi regions are separated by a line (or plane in 3D), perpendicular and midway to the sample point of each region (see fig. 2.12a).


Figure 2.12: a) Voronoi diagram: Two Voronoi cells are separated by a line, perpendicular and midway to the data points of each cell. b) Delaunay triangulation as the dual of the Voronoi diagram. An edge is formed between two data points, if their Voronoi cells meet along a line segment. c) The Voronoi circles do not contain any other data points. The three points on their boundary form the vertices of the Delaunay triangles.

Common points, where three different Voronoi regions meet, can be regarded as the centre of empty circles, also referred to as circumcircles or Voronoi circles (see fig. 2.12c), which have the three adjacent sample points on their boundary. The circles do not contain any other samples. There are special cases where four or more Voronoi regions are meeting at a common point, but theses special cases are very rare and can, for this brief introduction, be neglected.

The Delaunay diagram is the dual to the Voronoi decomposition. "We get a dual diagram if we draw a straight *Delaunay edge* connecting points $\mathbf{p}, \mathbf{q} \in \mathbf{S}$ if and only if their Voronoi regions intersect along a common line segment;"(Edelsbrunner, 2000, p.5) In other words, when two Voronoi regions share a common edge their sample points can be connected. Repeating this step for all vertices will subdivide the space into triangles, and is thus also referred to as the Delaunay triangulation (see fig. 2.12b).

The empty circles are also representing this triangulation, since they will always have three points on their boundaries: the vertices of the Delaunay triangles.

In 3D, the Delaunay triangulation forms tetrahedra and, as can be seen in figure 2.13, the process of extracting a triangulated, manifold surface, where only two triangles share an edge, becomes a challenging task. This is especially true for real world scenarios, where noise is always present, causing points to lie either above or below the correct surface. This difficulty is the central problem, which the following approaches, "The Power Crust" and "The Ball-Pivoting-Algorithm" try to overcome.

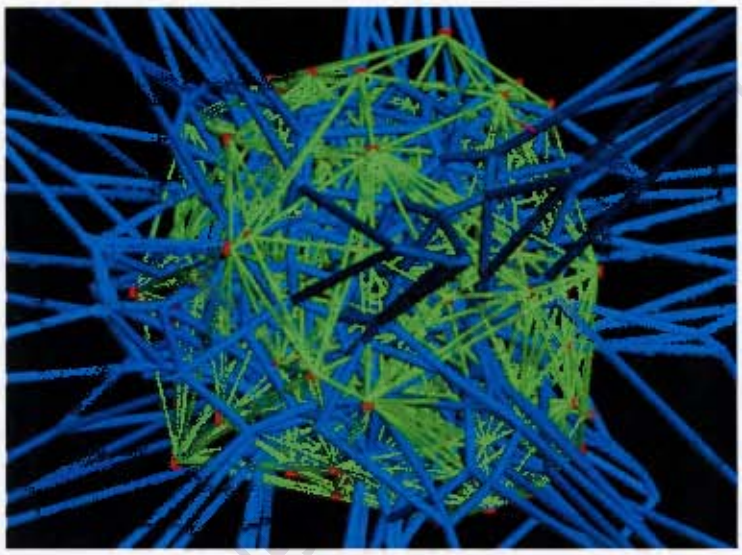

Figure 2.13: Image depicting the Delaunay triangulation in 3D (green lines) forming tetrahedra and the Voronoi cells (blue lines). Image from documentation of Graphite software, http://alice.loria.fr/index.php/software.html



#### 2.3.1.2 The Power Crust

The Power Crust algorithm by Amenta et al. (2001) is based on the Voronoi diagram / Delaunay triangulation. A central element of this approach is the medial axis, which is the set of points which have more than one closest point to the surface S (see fig. 2.14). The medial axis contains the centres of weighted empty balls, which are completely inside the object. Together they form the medial axis transform, which represents the surface. The authors found that the medial axis transform approximates the surface very well, even with noisy data input. Because computing the medial axis is very expensive in computational terms, the authors use the Voronoi diagram and a subset of its vertices, the "poles", to approximate it. Poles are the two corner vertices of a

Voronoi cell, which are located the furthest away from the cell's input data sample. As the medial axis lies in the centre of the model, only the vertices of a long and skinny shaped Voronoi cell will be located close to it. Hence, the authors consider only these vertices as poles. Only empty balls at those poles (polar balls) form the power crust (see fig. 2.14). The approach proves to be robust to noise and can produce sharp edges.

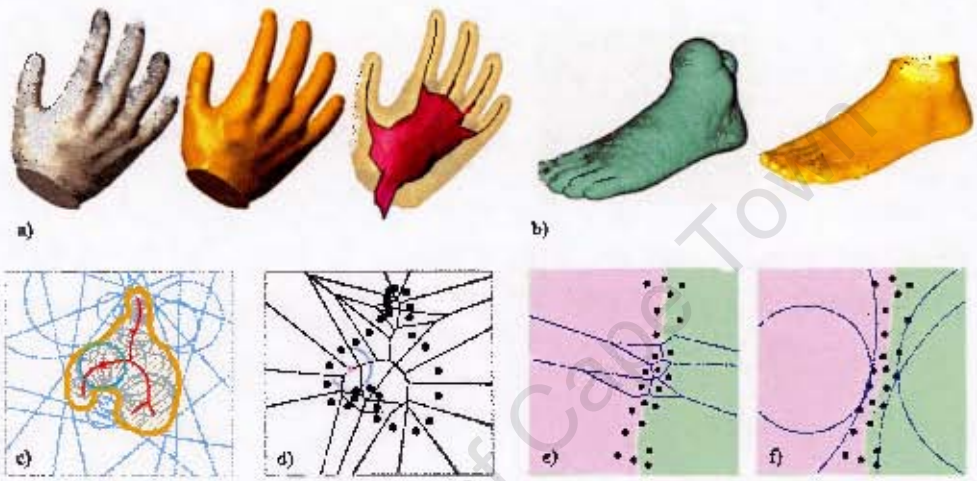


Figure 2.14: a): Input point set, its polygonal representation and a simplified version of its medial axis. b) Inner polar balls and the polygonal result. c) The Power Crust (orange) and the medial axis (red) are defined by a set of empty balls (light blue). d) The Voronoi diagram of c); e) Image depicting the concept of poles: Only long and skinny Voronoi cells are considered for the Power Crust. The poles are the vertices of such Voronoi cells and are located close to the medial axis. f) Polar balls. Images adapted from Amenta et al. (2001)



#### 2.3.1.3 The Ball Pivoting Algorithm

The ball-pivoting-algorithm is a "region-growing" (Bernardini et al., 1999, p.2) approach. A ball of a pre-set size "sits" on the vertices of a seed triangle, so that it the vertices touch the ball's surface. The algorithm of Bernardini et al. (1999) then pivots the ball around the triangle's edges. If it touches another point while pivoting, a new triangle will be formed and attached to the previous one (fig. 2.15). Noisy data samples which are further away from the surface than the sampling distance will not be touched by the ball, as demonstrated in figure 2.15a. In addition, pre-determined

vertex normals are used to prevent the ball from pivoting further, when the surface's boundary was reached (fig. 2.16b).

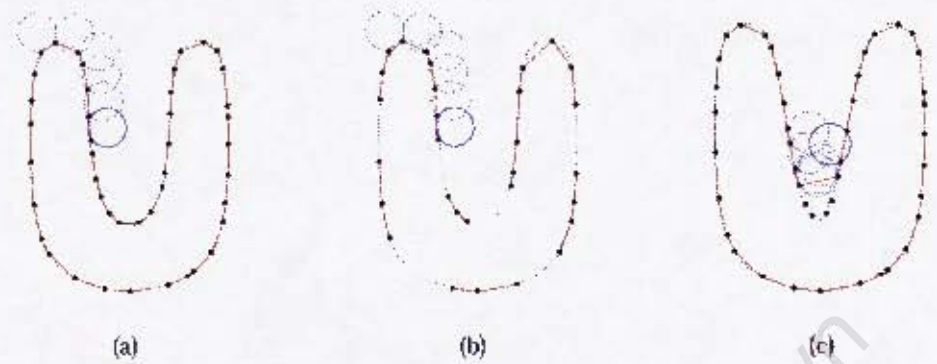


Figure 2.15: Image demonstrating the Ball Pivoting Algorithm: (a) A ball of constant size rest on the vertices of a seed triangle. The algorithm then pivots the ball around the edges of the current triangle (not visible on this image as the edges are parallel to viewing direction). If the ball touches a new vertex while pivoting, new edges (red lines) in between old and new vertices will form a new triangle and the ball will pivot around the new edges to find adjoining triangles. (b) If the point density is lower than the ball radius, holes are created (c). If the ball-radius is larger than the point density, detail will be left-out. Images from (Bernardini et al., 1999)

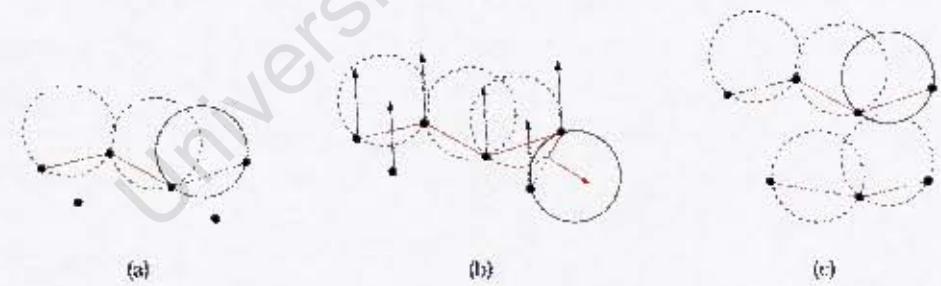


Figure 2.16: Ball Pivoting Algorithm (a). If the ball is large enough, noisy point samples will not be touched. (b) The algorithm requires vertex normals to be determined beforehand. If the orientation of the candidate triangle differs largely from the normal of the data sample which was "touched", no triangle is formed. This prevents the algorithm to cross the boundary of the surface and continue pivoting on the noisy samples of (a). (c) If points are further away than the ball radius, two surfaces will be created. Images from (Bernardini et al., 1999)

It seems non-trivial to finlt the correct ball-radius, because if set too large, detail

will be lost, if set too small, holes will occur. To circumvent this issue, the authors recommend to execute the algorithm repeatedly with an increasing ball-radius, completing the existing surface with each iteration.

#### 2.3.1.4 Zipper

The "Zipper" approach by Turk and Levoy (1994) is different from the previous approaches, as its focus is to connect several individual, already triangulated laser scans, rather than to reconstruct a new surface from a point cloud. It detects the overlapping parts of the input scans and discards the redundant information. The independent surface pieces are connected by a new triangulation between their new borders (see fig. 2.17).

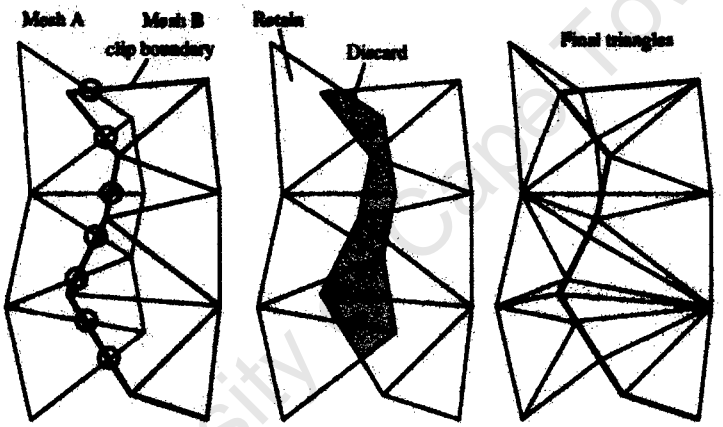


Figure 2.17: Diagram showing the Zipper approach, which (a) detects overlap between two surfaces and (b) discards the redundant information. (c) The gap in between the two surfaces is then closed by triangulation between the boundaries of each surface. Image from (Turk and Levoy, 1994)

### 2.3.2 Surface Reconstruction by Approximation

Approximating algorithms use functions to describe a surface. Mathematically, many surface reconstruction approaches (in here, all of the following) employ implicit-, as opposed to parametric surface representations.

> The basic concept of *implicit* or *volumetric* representations for geometric models is to characterize the whole embedding space of an object by classifying each 3D point to lie either inside, outside, or exactly on the surface S that bounds a solid object. (Botsch et al., 2010, p.13)

#### 2.3.2.1 Implicit Functions

An example of a basic implicit surface representation is F(p) = 0. In this example, all points, where the evaluation of the function returns zero (zero-crossing), are located on the surface. Positive values are usually defined as points outside the object, while negative values indicate points inside the object. The surface defined by the zero-crossing of the function, is a special case of the isosurface. According to Newman and Yi (2006), an isosurface is defined as follows:

> "Given a scalar field F(P), with F a scalar function on $\mathbb{R}^3$, the surface that satisfies F(P) = $\alpha$, where $\alpha$ is a constant, is called the *isosurface* defined by $\alpha$. The value $\alpha$ is called the *isovalue*." (Newman and Yi, 2006, p.854)

#### 2.3.2.2 Marching Cubes

Implicit functions are continuous representations of a surface. To convert them into triangles, they need to be discretised. For efficient processing, a regular or adaptive grid structure is employed to evaluate the implicit function at specific points in space. For this purpose, a cube containing the data is subdivided into smaller sub-cubes, also referred to as voxels. The marching cubes approach (Lorensen and Cline, 1987) evaluates the implicit function at the eight vertices which define the voxel and "marches" from cube to cube.

By evaluating the function on the vertices, the algorithm can detect whether the vertex is above or below the surface. If the state of two vertices, which are connected by an edge, differs, the surface is crossing the edge in between. The algorithm examines all edges for intersection and cross-checks in a predefined look-up table which one of the 256 triangulation possibilities fits the current condition. Because there are eight edges in a cube, there can only be 256 different intersection configuration, which are reduced to 15(fig. 2.18) by considering symmetric and rotational relations.

The dimensions of the voxels determine approximately the resolution of the final mesh, because per voxel, one or more triangles are constructed. The edges of these triangles cannot become longer than the edge of a voxel, only smaller. The regular subdivision makes these approaches very scalable and efficient, as their memory usage depend more on the output resolution, rather than the input dataset and are thus well suited for working on large datasets.

The marching cubes algorithm (Lorensen and Cline, 1987) or one of its derivatives (Newman and Yi, 2006) is the most common method used in implicit/volumetric approaches to extract the isosurface from a volume and to create a triangulated surface.

Most of the following approaches thus only need to focus on assigning the correct implicit function, describing the surface best.

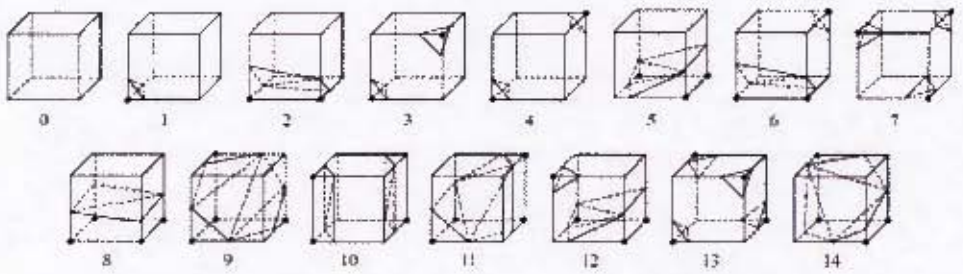


Figure 2.18: Marching Cubes Algorithm: Image showing the 15 unique possible configurations of a surface crossing a cube. Other configurations related through rotation and symmetry are neglected for efficient processing. Images from (Newman and Yi, 2006)



#### 2.3.2.3 Volumetric Integration

To establish the implicit representation on a voxel grid, Curless and Levoy (1996) first triangulate each scan individually by evaluating nearest neighbour information provided by the scanning grid (see section 3.4.2.1). They then assign an implicit function to each vertex of the voxel grid near the surface. To be more specific, the implicit function is a weighted signed distance function, which represents each point x in relation to the surface of the scan along the line-of-sight of the scanner. The functions of all scans are combined and evaluated at the vertices of the voxel grid. The new zero-crossing of the combined implicit functions represents the best-fit surface of all scans (see fig. 2.19).

The isosurface is then extracted via the Marching Cubes Algorithm. Callieri et al. (2003) extended the approach of Curless and Levoy, to out-of-core techniques. They developed PlyMC, a tool which splits the volume first into smaller sub-volumes and processes these individually. This reduces the necessary amount of main memory to a manageable size by only loading the scans relevant to the sub-volume.

#### 2.3.2.4 Implicit Surface Reconstruction from Unorganized Points

Hoppe et al. (1992) proposed a method to approximate the surface on an unstructured point cloud. The neighbours to a sample point are gathered and used to fit a tangent plane in a least squares sense to the sample point. The distance between any arbitrary point and its projection onto the plane is used to create a signed distance function.

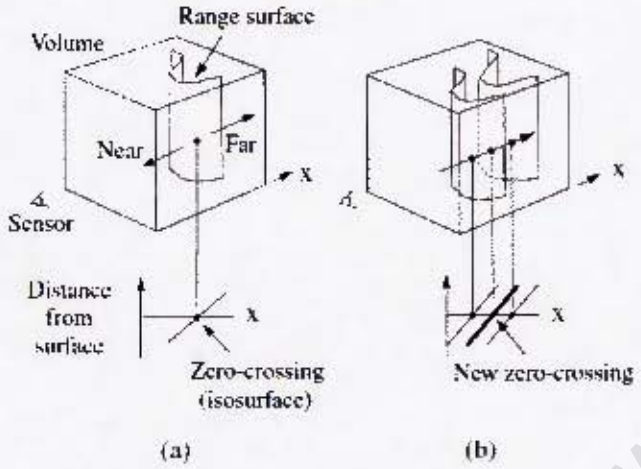


Figure 2.19: Image depicting the Volumetric Integration approach: Per scan, a weighted signed distance function describes each point in relation to the surface along the line-of-sight to the scanner. All functions of all scans are combined and the new zero-crossing of the combined implicit function is the final surface, which is basically the average between all input scans. Image from (Curless and Levoy, 1996)

As in the Volumetric Integration approach, the function is then evaluated on a regular volumetric grid and transformed via a variation of Marching Cubes into a triangulated mesh (see fig. 2.10). The approach allows the meshing of un-organized data samples and proves to be robust to noise.

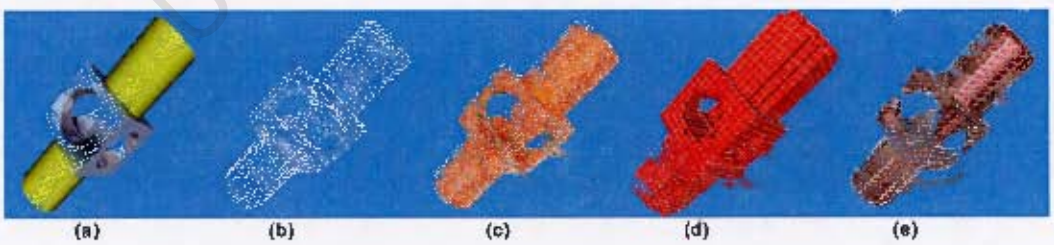


Figure 2.20: "Surface Reconstruction from Unorganized Points" by (Hoppe et al., 1992): (a) Original Constructive solid Geometry (CSG) model, (b) Point set derived from CSG model, (c) tangent planes fitted to point data, (d) voxels filled with data, (e) extracted surface. Images adapted from (Hoppe et al., 1992)

#### 2.3.2.5 Moving Least Squares

The Moving Least Squares approach, or also Moving Local System approach, (MLS), was introduced by Levin (2003) in 2003. For each point of a data set, a best-fit surface of the local neighbourhood is established. A projection operator re-adjusts each data point by projecting it onto the local best-fit surface. The point's new location is then used for the projection processes of the remaining points.

In a first step, a plane $H$ is fitted to a sample point $r$ and its weighted neighbours $p_i$ with least squares (see fig. 2.21b). The plane defines a new local coordinate system, of which the origin $q$ is the projection of $r$ onto $H$ along the normal $n$ of the plane. The neighbouring points $p_i$ are weighted according to their distance to $q$.

Secondly, a local polynomial approximation $g$ of the surface over $H$ is computed, onto which $r$ is then projected along the normal of $H$ in a final step. This is done for all points, so that in the end all points are representing the best-fit surface.

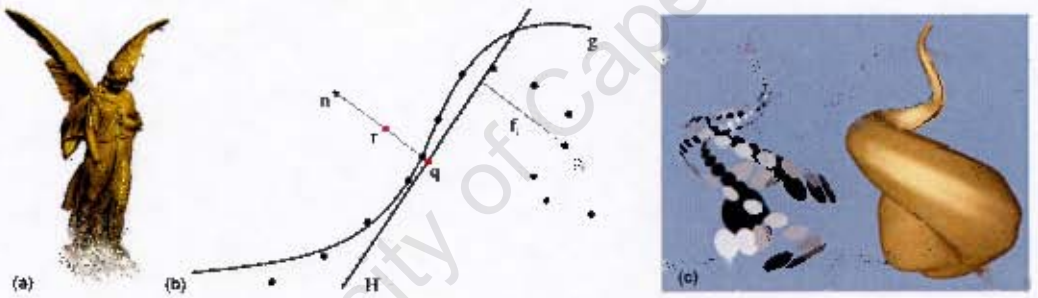


Figure 2.21: (a) a Point Set Surface. (b) a plane H is fitted to a data point r. The influence of the surrounding points is weighted based on their distance to the origin q, which is a projection of r onto H. A polynomial g over H is then computed and r is projected onto it. This process is repeated with all data points, so that in the end, their projection represent the best-fit surface. Images from (Alexa et al., 2001). (c) Points and the surface they define. Image from (Amenta and Kil, 2004).

Amenta and Kil (2004) showed that the polynomial fitting step can be omitted, which simplifies the computation of MLS surfaces. The surface is approximated, by iteratively (re-)fitting a plane and projecting the point along the normal onto it, as described above, until the energy function, which represents the quality of the fitting, converges.

Because the plane fitting can become unstable if the sampling density becomes too low, Guennebaud and Gross (2007) suggest to fit algebraic surfaces instead, such as spheres.

The MLS approach has found many applications and was adopted and extended in several ways. Amenta and Kil (2004) and Alexa et al. (2001), for example, employed it for the so-called "Point Set Surfaces", which are point based, mesh-less representations. The point data implies a surface which is calculated and rendered on-the-fly (fig.2.21a,c).

The main advantage is that these surfaces are easier to display than triangles and thus more point samples can be rendered at a time, describing more detail. Also, because the surface does not need to be converted to a mesh first, it simplifies the processing pipeline and reduces the processing time.

To translate the Point Set Surface into a triangle mesh, Fiorin et al. (2007) assign and evaluate a signed distance function, similar to the previous approaches, on the cell corners of a volumetric grid. The triangles are then extracted via Marching Cubes. With out-of-core, parallel and streaming techniques, Cuccuru et al. (2009) report to be able to extract a surface from 380 million points with the MLS method in less than 85 minutes.

#### 2.3.2.6 Poisson Surface Reconstruction

A recent idea is the Poisson approach by Kazhdan et al. (2006). They try to find an indicator function which is defined to be 1 inside the model and 0 outside of it. The function's gradient, which can be seen as a vector field, is approximated by the normals of the input sample points (see fig. 2.22). The relation in between the gradient and the vector field is defined as a standard Poisson problem:

"[...]compute the scalar function $\chi$ whose Laplacian (divergence of gradient) equals the divergence of the vector field" (Kazhdan et al., 2006, p.1)

Instead of computing one global function which fits the whole surface, for example Radial Basis Function, presented by Carr et al. (2001), Kazhdan et al. are computing a set of hierarchical local functions. Global approaches have usually the disadvantage that the final matrix becomes too large when representing a large dataset. But a positive by-product, and often required deliverable, is the extraction of a watertight surface.

Hierarchical local functions however, eliminate the space-complexity of a global function, by breaking the problem down into smaller local functions (which is more appropriate for large datasets) without losing the ability to create of a watertight surface. All data is sorted and stored in an adaptive octree, of which the nodes are equipped with local functions, providing the information about the covered surface in its branches. The octree combines functions of small local patches in the deeper regions

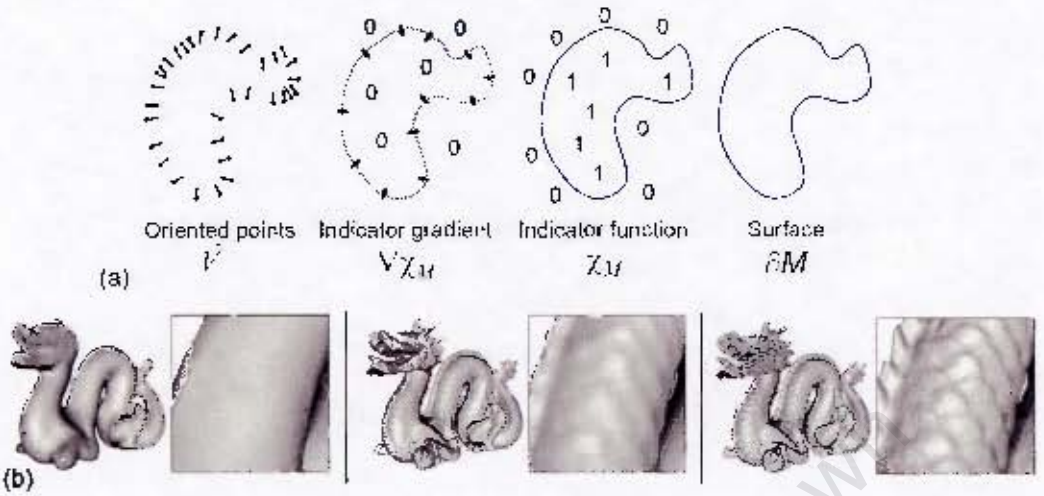


Figure 2.22: Poisson approach. (a) The aim is to find an indicator function defined as 1 inside the object and as 0 outside. The gradient of the indicator function can be visualized as a vector field, which can be approximated by the normals of the input sample points. (b) The extracted 3D models at increasing octree depth are shown. Images from (Kazhdan et al., 2006)

of the tree with the parent functions of the lower levels. The higher the octree level becomes, the more nodes for the same surface area are available. Thus, the amount of detail described by the functions of higher levels can be increased, without increasing the space-complexity of the functions. This adaptive structure produces fine detail in areas with many samples, while under-sampled areas are also approximated. The approach results in much smaller memory consumption than a global solution, but of course, the maximum octree depth is also depending on the available memory. In recent papers the idea is extended to multi-level streaming (Bolitho et al., 2007), subdividing the datasets into narrow slices and thus limiting the memory consumption to small, manageable pieces and was also enhanced by parallel computing (Bolitho et al., 2009).

## 2.4 Hole Filling

During data acquisition, especially while scanning larger buildings, it is not possible to entirely avoid data shadows, created by objects in the line-of-sight of the instrument, bad reflectance properties of the scanned surface, or simply because there is no qualified scanning position available. This is especially true for the top of walls and the roof, if there is no suitable, higher elevated point available to position the instrument. Hence

a hole filling or mesh repair step can become necessary.

Manufactured data, which is not based on accurate measurements, is a delicate matter in heritage documentation, and beyond the technical scope of this research. It shall thus only very briefly be discussed.

Smaller holes can be filled automatically. Davis et al. (2002) presented an approach for implicit surfaces by diffusing the implicit function with a low pass filter along the boundaries until the hole is closed. It can be seen as an extrapolation of the surface along the boundaries and can lead to a smooth patch.

Bernardini et al. (1999) suggest closing holes by running their ball-pivoting algorithm (see section 2.3.1.3, p.27) several times with increasing ball radius. New triangles will only be inserted on the boundaries and not replace any existing triangulation. This way, small detail will be retained, while more and more holes are filled during each iteration. But the larger the holes are, the larger the ball and thus the triangles will be, which are used to close the hole. Hence the patches will become more and more prominent in the final model.

There are also some watertight surface reconstruction algorithms, which fill all holes by definition, such as the Poisson-algorithm by Kazhdan et al. (2006), mentioned above. But also here, the larger the holes become, the more unrealistic the fill-in surface patch will be. In this case, the problem of closing holes is replaced by the problem of finding a solution for removing and fixing these inappropriate fillings.

All of these methods are just interpolating and the created patches are usually clearly identifiable either by large triangles or by the smoothness of the patch, compared to the rest of the model, as fine detail is not restored. Sharf et al. (2006) presented an iterative approach using an octree data structure. Similar to 2D image texture synthesis, they analyse the surrounding surface of a hole and copy and paste a surface patch from other parts of the model, which fits best the conditions around the current hole. In subsequent iterations, the process is repeated at higher octree levels, and thus the surface is refined with more detail. The results presented by Sharf et al. demonstrate that holes can be filled very convincingly, making it hard to distinguish between measured and artificially introduced data, which makes this approach not suitable for heritage documentation (see fig. 2.23).

## 2.5 Simplification Process

Simplification or decimation is the process of approximating a surface M with a less complex surface M'. This is an important part of the mesh processing pipeline, as the amount of triangles, generated by current meshing technologies, can easily become too

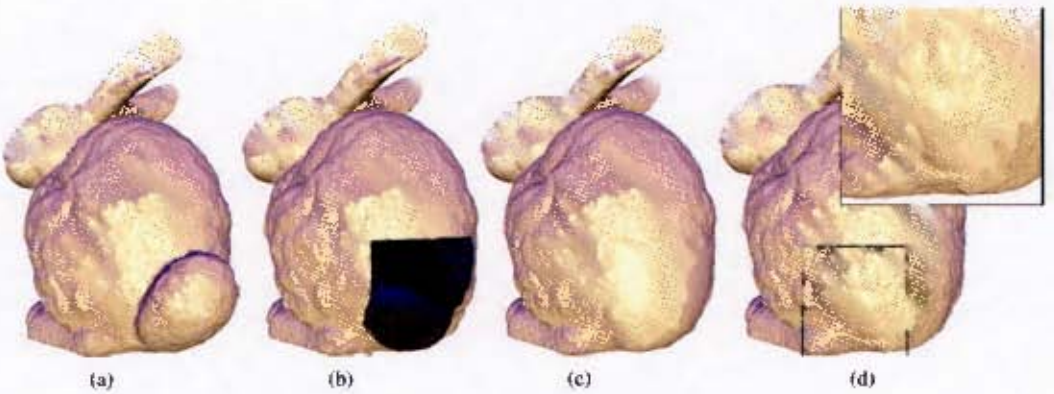


Figure 2.23: Context-based Surface Completion: (a) The Stanford bunny model. (b) A part of the back is removed, including the tail. (c) The hole is filled roughly. (d) The repaired surface simulating the fine texture of the surface surrounding the hole. Images from (Sharf et al., 2006)

large to fit into main memory. Especially Marching Cubes (see section 2.3.2.2), if a static grid structure as opposed to an adaptive approach is employed, creates triangles of more or less regular size, regardless of the detail the triangles actually describe.

Thus, simplification is a justifiable method to create an approximation which looks almost identical to the original but with a much smaller mesh, which helps editing and viewing the model. Over the past 20 years, simplification has developed to a mature technology. The literature published on this subject is substantial and for a more elaborate survey and classification, the reader is directed to Cignoni et al. (1998). Today, the tendency is not to simplify the model to certain sizes for different usages, but to use simplification techniques for real-time rendering of large datasets. In the following some important techniques to simplify meshes and to view large datasets are summarized.

Simplification of triangles basically means removing the vertices of the mesh. The two most important characteristics of a simplification algorithm are the way of selecting points for removal and how the resulting hole is filled.

### 2.5.1 Vertex Decimation

To select candidates for removal, Vertex Decimation, introduced by Schroeder et al. (1992), uses the distance between the vertex and the average plane defined by the current vertex' neighbours, or a line, in the case of a boundary or feature edge. The vertices close to the plane are eliminated based on a user selected parameter and the newly created hole is re-triangulated with less faces.

### 2.5.2 Simplification Envelopes

Simplification envelopes (Cohen et al., 1996) define the space between an inner and outer offset surface (the envelopes) as a valid range which should contain the final surface. It is independent of the actual decimation and serves as a method to provide a global error control.

### 2.5.3 Vertex Clustering

Vertex clustering, introduced by Rossignac and Borrel (1993) is a grid sampling method, removing all vertices of a (3D) grid's cell, and replaces them with a single vertex. The detail of the resulting mesh thus depends on the size of the grid's cells. Weights are assigned to the vertices, preferring points on the (view-independent) silhouette of the model and vertices that are part of large triangles. If smoothing is acceptable, the best position for the new vertex, replacing the others, is calculated as the centre of mass of the weighted vertices. Alternatively, if smoothing is not desired, the vertex with the highest weight is used.

Lindstrom (2000) later enhanced this approach by the use of quadrics -which will be described in the following section- and a slim out-of-core implementation, enabling the decimation of meshes in the order of several hundred million polygons. The algorithm seems to produce very satisfactory results even for very large meshes, but as Lindstrom notes, clustering artefacts become visible if the cell size is too large. Also, due to a fixed cell size, the approach often cannot decimate efficiently areas of low curvature, if the cell size is smaller than the flat surface. Lindstrom thus recommends using this out-of-core approach as a pre-processing simplification to decimate very large meshes to a more manageable size, which can be processed with other approaches.

### 2.5.4 Edge Collapsing

Hoppe et al. (1993) define several operations for an edge: collapse, split and swap (see fig.2.24). Even though originally developed for optimizing meshes after surface reconstruction, such as (Hoppe et al., 1992), to re-fit the extracted mesh closer to the input samples, Hoppe et al. (1993) show that the approach can also be used for simplification. They not only remove vertices by collapsing edges but also add some vertices, depending on the minimization of an energy function which attempts to balance between the distance of the vertices and the mesh and the total amount of vertices. The three methods, edge collapse, split and swap were then reduced to only the edge collapse method, which proved to be sufficient (Hoppe, 1996).

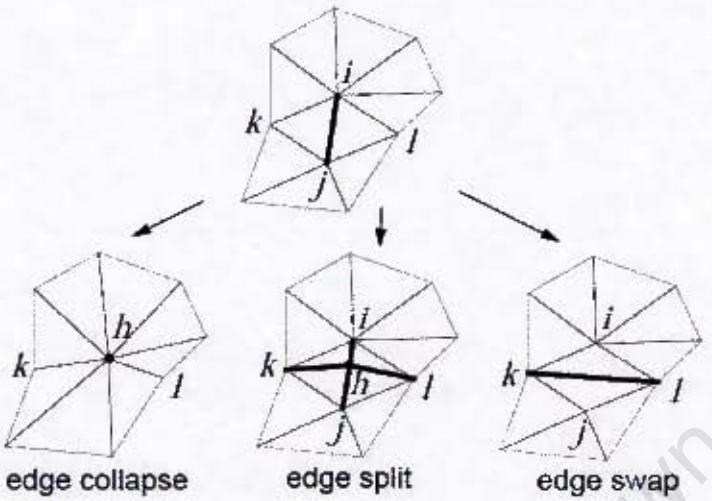


Figure 2.24: Simplification operations for vertices: collapse, split and swap. Hoppe (1996) later showed that only edge collapse is really necessary for simplification. Images from (Hoppe et al., 1993)

### 2.5.5 Quadric Error Metric

Garland and Heckbert (1997) define each vertex as the intersection of planes, formed by the triangles incident on the vertex. When two vertices are merged into one, they measure the squared distances of the new vertex' position and the previous set of planes. The error is then encoded in a 4x4 matrix and all matrices for each triangle-vertex distance are added together. This error matrix forms a quadric surface, containing all potential points where the global error for the new vertex is the same (fig. 2.25).

To decimate the dataset, only the edge-collapse technique is employed, which is the merging of two vertices connected by an edge. They further extend it to non-connected, but neighbouring triangles, called non-edge contraction (fig.2.25). This enables the algorithm to combine originally disconnected parts, which is often wanted, since some vertices might be very close to each other anyway. After each contraction, the error matrix of the remaining vertex is updated and the matrix of the removed vertex is added as well. Then a new vertex is inserted at a position, minimizing the error. The algorithm stops, when it reaches a user-set error-level or the targeted amount of triangles.

The algorithm is regarded as a de-facto standard and was extended to allow the support of textured meshes (Garland and Heckbert, 1998). Cignoni et al. (2003) chose quadrics as their standard simplification criteria for their Octree-based external mem-

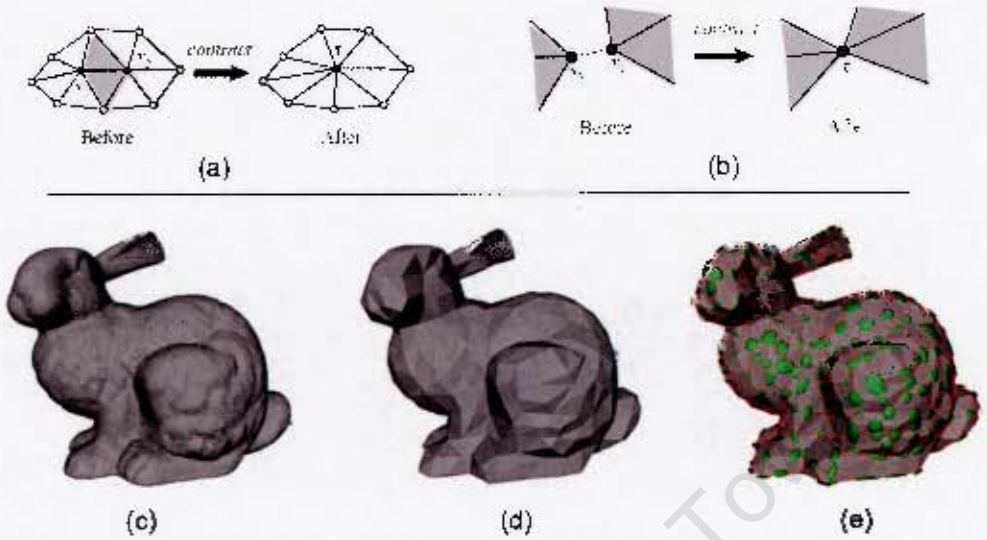


Figure 2.25: Simplification using quadric error metrics: (a) To reduce vertices, the edges are contracted or collapsed. (b) Also vertices close to each other, which are not connected by an edge, can be merged (non-edge contraction). Below: The Stanford Bunny Model, reduced from 69451(a) to 1000(b) triangles and the corresponding error ellipsoids per vertex (c). Images from (Garland and Heckbert, 1997)

ory simplification framework and thus enabled the decimation of very large meshes independently of system memory.

### 2.5.6 Progressive Meshes

Bearing the previous techniques in mind, the trend is not to produce simplified meshes as standalone products any more, but rather as part of a multi-resolution representation to enable real-time rendering.

Progressive meshes, proposed by Hoppe (1996) are hierarchical meshes, with the lowest resolution as a base mesh. It stores the inversion of the simplification process, or in other words, how to get from the base mesh to the highest resolution for the currently displayed part of the model. Here, the reverse of the edge collapse operation of the simplification process, vertex splits, are used to get from the lowest to the highest resolution (fig. 2.26a). This can drastically speed up the rendering time, as only the base mesh remains in memory at all times, and only the instructions, how to re-create the high resolution of parts, currently needed for display, are loaded and processed (fig.2.26b). To prevent the user from noticing, how the mesh is built-up, the

new vertices of the newly added triangles are morphed between old and new position over several frames. In a perspective view, this procedure enables a more advanced control over the displayed level-of-detail (LOD). For example, an object, very far from the viewer, does not need to be shown in the highest resolution, as the triangles might shrink to a size smaller than a pixel. Thus, a simplified surface can be displayed instead.

In a successive paper, Hoppe (1998) proposed the subdivision of very large meshes into smaller sub-models, simplifying them individually and then combining and simplifying their base meshes.

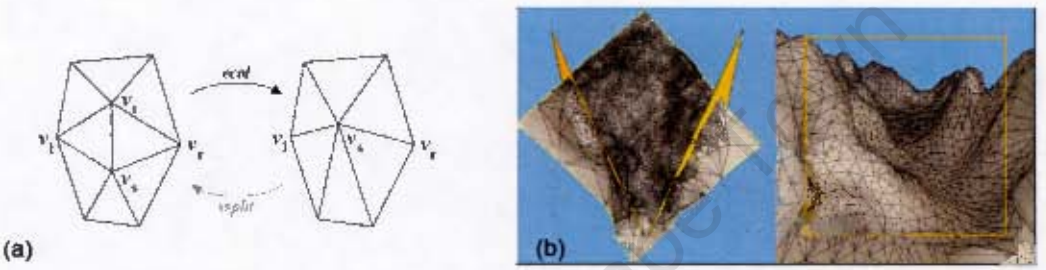


Figure 2.26: Progressive meshes. (a) 'Vertex split' as the reversion of the 'edge collapse' operation. (b) Image depicting the loaded level-of-detail of a specific view frustum (field-of-view in orange). Note that triangles outside the orange bounds are in a lower resolution than the ones within. Images from (Hoppe, 1996)

Since then, the idea of hierarchical displaying method was developed further. In 2000, Rusinkiewicz and Levoy (2000) proposed a hierarchical visualization tool, called QSplat. The mesh is represented by the vertices and a hierarchical structure of bounding spheres.

### 2.5.7 Adaptive Tetrapuzzles

Cignoni et al. (2004) and their 'adaptive Tetrapuzzles' approach, showed a way with quadrics and edge collapse to create a multi-resolution system for large meshes, which would not fit into main memory otherwise. They perform an out-of core simplification to create a multi-resolution mesh by subdividing the model into a tree of tetrahedral cells (fig. 2.27). They also created a visualization tool which is calling the different resolution levels of the individual cells, depending on the current view onto the model. They achieve more than 40 frames per second on a several hundred million triangles mesh, which allows smooth interaction with the model. In a later paper they generalize the idea and propose a framework for it (Cignoni et al., 2005).

Figure 2.27: Adaptive Tetrapuzzles: Images showing several levels of detail of the same model (depending on the camera position) and the corresponding fragmentation into patches. Images from (Cignoni et al., 2004)

### 2.5.8 Streaming Meshes

Multi-resolution meshes, or progressive meshes, are important for interactive viewing. On the other hand, many of these algorithms seem to increase the file sizes of the models, which is contra-productive, if the aim is for example the transmission over the internet. So in parallel to the creation of progressive meshes, there is research in progress, on how to best compress the data for efficient streaming.

The main problem with today's large meshes is that their file formats are encoding the model as indexed meshes (e.g. the PLY-format). In there, the vertices are listed and indexed first, followed by the triangles, which are stored as the three indices to the triangle's vertices. To read the mesh, all of the indices to the vertices for each triangle need to be dereferenced first, meaning the indices have to be replaced by the vertex-coordinates they link to. Faces which are topologically connected are not necessarily listed next to each other in the PLY-File. Thus, to display even just a portion of the mesh, the whole file needs to be read first. Isenburg and Gumhold (2003) propose to spatially cluster the vertices and to store more connectivity information, such as the next and the previous edge. They also define a compression format, which enables a compression rate of about 90-97 percent of the original file size, depending on the float numbers precision levels.

Isenburg and Lindstrom (2005) however, proposed an even more efficient and coherent file format to create "streaming meshes". They propose a format which provides additional information on for how long a vertex is needed. For example streaming meshes in "pre-order" list a triangle directly after all vertices, which it refers to, were introduced (see fig. 2.28d). This way, the model can be loaded sequentially and only a small set of active vertices needs to stay loaded. As soon as all triangles referring to it are drawn, the vertex can be removed from memory.

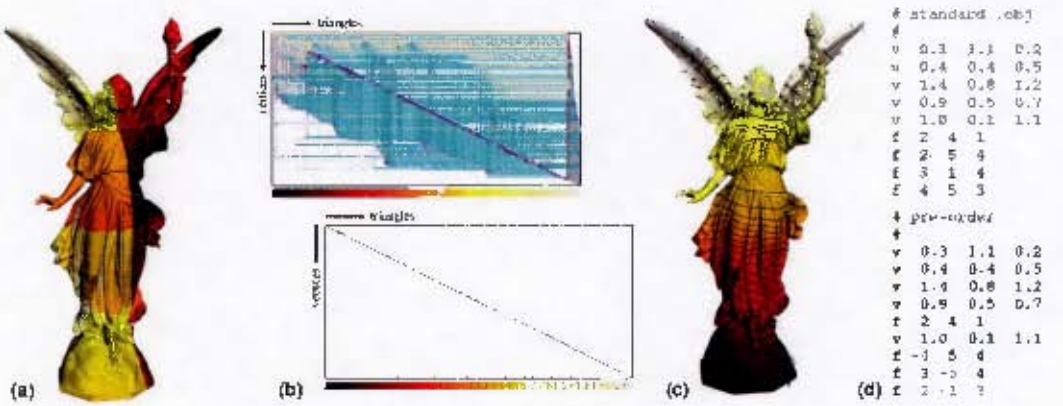


Figure 2.28: Streaming meshes: The colour codes of the statues show the position of triangles in the file before (a) and after (c) re-ordering. (b) The diagrams express the same in a different way: the blue lines in the diagrams connect triangles which share the same vertex, with respect to their position in the file. (d) File layout of a OBJ-file and a new file format optimized for streaming of meshes. Images from (Isenburg and Lindstrom, 2005)

The advantage seems to be obvious: much smaller files and a much more efficient way to access the mesh, or sub-parts of it. On the other hand the CPU needs to decompress the data first before it can be displayed. For interactive display, this seems to be a contradiction, since valuable CPU speed is lost. Jamin et al. (2009) however, tried to find a good compromise in between and designed a viewer, called CHuMI with an efficient data structure and lossless compression. The proposed format allows the data to be decompressed in stages, refining the detail at higher levels. Only those parts which are currently needed are decompressed to an extent showing the optimum amount of detail according to the distance of the camera to the surface.

## 2.6 Texturing

The 3D model, by standard shaded in neutral grey, can be augmented with colour information to enhance the visual appearance to enable a more holistic representation and examination. This process is commonly referred to as texturing, even though strictly speaking, the term refers to texture mapping, a specific technique of applying colour to a model, which will be explained in section 2.6.2.2.

To texture a model, the user can paint colours directly onto the surface, apply a repetitive texture pattern to a part or the entire model, or create a procedural texture, which is a computer generated texture, which can be influenced by several parameters.

Procedural textures are common to solid modelling as they can convey true 3D colour information and not simply pasting "posters" onto the surface. A procedural wooden texture for example would generate bark on the outside of a tree trunk and also the rings on the inside. Even though these methods mentioned so far, can significantly affect the users impression of the model, the benefit for the documentation of cultural heritage is limited, as they are manufactured representations and thus convey the wrong impression.

Many laser scanning instruments acquire photographic imagery in addition to scanning in a separate step with either a built-in or an externally mounted camera. For external cameras, producers of laser scanner often offer brackets, to keep the lens' principal point in a known, fixed position and close to the laser scanner's centre of rotation. This reduces parallax effects and minimizes the efforts of aligning the imagery manually to the 3D data as necessary when combining images and scan. In cases where laser scanners are able to scan a full 360 degree window, panoramic images are necessary to colour all sampled surface points.

There are a number of advantages in this kind of procedure, such as the direct alignment of 2D imagery and 3D data, minimizing the users-intervention time as well as the guarantee for the colourization of every data sample. Problematic however, is the restriction of taking images only at the time of the scan, which leads to discrepancies in the appearances of the scans, due to different lighting conditions throughout the day. It further restricts to scanning at day-times, unless artificial light sources are employed. Thus, especially for long scanning exercises, it seems more appropriate to take additional imagery with hand-held cameras, if possible, more or less around the same time of the day.

### 2.6.1 Image Alignment

According to basic photogrammetric principles, a photograph can be accurately projected onto the 3D model if the internal and external camera orientation is known. To determine these unknowns, one has to select corresponding points on the image and the 3D model. In a best case scenario, the camera is calibrated and only the camera's position and orientation need to be established, which reduces the amount of necessary corresponding points to a minimum. But calibrated cameras usually cause more or less blurred images, because the focus cannot be adjusted while taking the photo. For this reason it is desirable to also make use of images from un-calibrated cameras, which, however, would require selecting about 15 points, including some extra points for extra accuracy, on the model and on the image.

Regarding the fact that for texturing an entire building, dozens or even hundreds of images need to be placed, this seems inefficient. Targets could be detected automatically, but for similar reasons that targets cannot be used for registration, they generally cannot be used for texturing either. Firstly, targets usually cannot be attached to sensitive and fragile surface and secondly, they should not be visible on the final model.

To at least minimize alignment efforts, Ikeuchi et al. (2003) suggested refining an initial manual alignment automatically by exploiting the intensity information of the laser scans. Intensity or reflectance strength values contain information about changes in material and colour. When creating a 2D image of these intensity values, the material changes can be automatically detected with image processing techniques (edge filters). The edges coincide with the edges in colour images and can thus be used for the refining the initial alignment.

Liu et al. (2006) propose an automatic alignment method for multiple images. In a first step, it is tried to register 2D images onto a point cloud by identifying vanishing points on an image and matching them with the directions of the line features extracted from the model. As this method usually only works on a subset of images containing such line features, the authors further perform a relative orientation of all images with SfM methods in a separate step. As some images, which are also part of the SfM process, were individually aligned in the previous step, a 3D transformation can be established to scale and align the entire SfM image set to the point cloud (fig. 2.29). Because this approach relies on line features, it works best on modern, urban scenes.

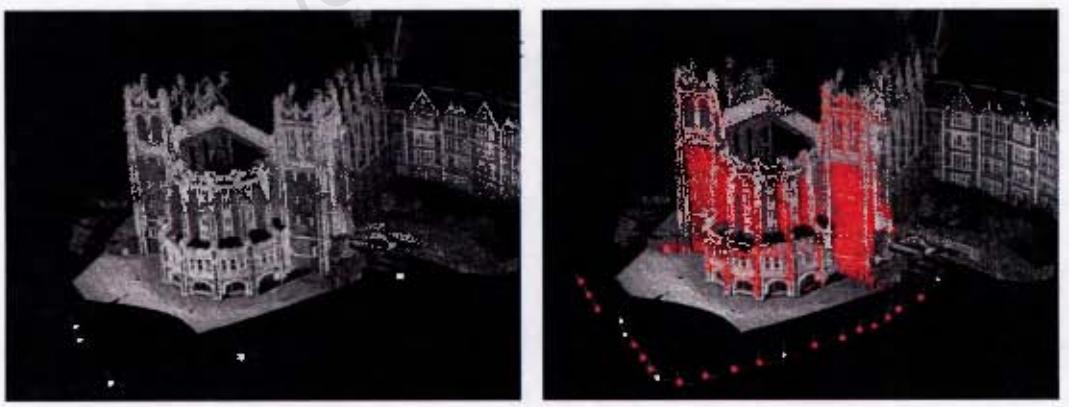

Figure 2.29: (a) Alignment of several images (white dots) via identification of vanishing points on the 3D model. (b) More images (red camera stations) are registered via SfM to the previous set. 3D point cloud of the SfM process in red. Images from (Liu et al., 2006)

Lensch et al. (2000) suggested an automatic method to align the images automatically based on the silhouette of the object. However, this method works best if the complete silhouette of the object is visible on the image and well separable from the background, a requirement which usually cannot be fulfilled for larger structures.

Corsini et al. (2009) also refine an initial alignment with their mutual information approach, a statistical rather than a feature-based method. Mutual information is a measure which expresses the similarity of two random variables. Corsini et al. (2009) extend this approach, so far widely used in medical imaging, to general image alignment (fig. 2.30). Their solution evaluates the joint histogram of a photograph and a rendering of the 3D model in an iterative process. After the evaluation, the model's position and orientation is adjusted and re-rendered until the mutual information evaluation converges. For a good correlation between rendered image and photograph, the shading of the model is essential. Not shading the model would correspond to a silhouette. To recover more detail, Corsini et al. (2009) propose to use normal mapping, a widely used computer vision technique, which encodes the direction of the model's surface normals into different colours. Alternatively, they also suggest a specular reflectance map, which is dependent on the current viewpoint and, according to the authors, works better with reflective objects. They also propose a combination of these techniques with additional shading, such as ambient occlusion, which darkens hidden areas (fig. 2.31). The mutual information approach was later refined to also include manual, user-set correspondences (Sottile et al., 2010) and a combination with gradient maps (Palma et al., 2010).



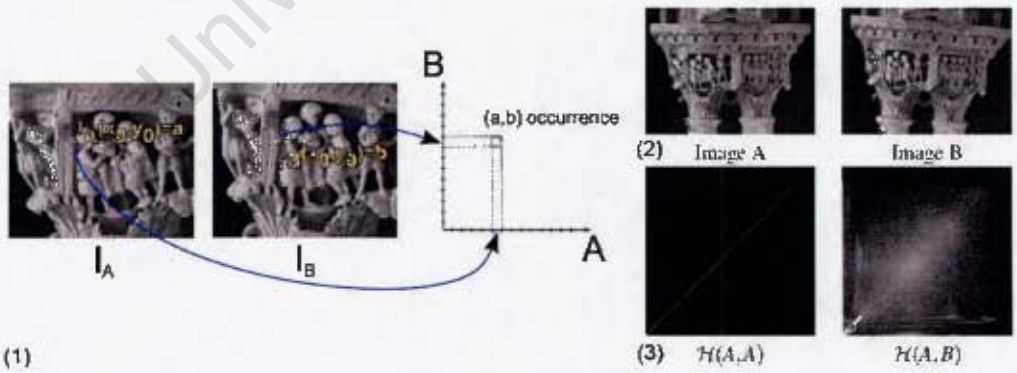


Figure 2.30: Mutual Information approach. (1) the correlation of two images can be evaluated from a joint histogram. (2) Two images (A) and (B). (B) is a slightly rotated version of (A). (3) two examples of joint histograms. Images from (Corsini et al., 2009)

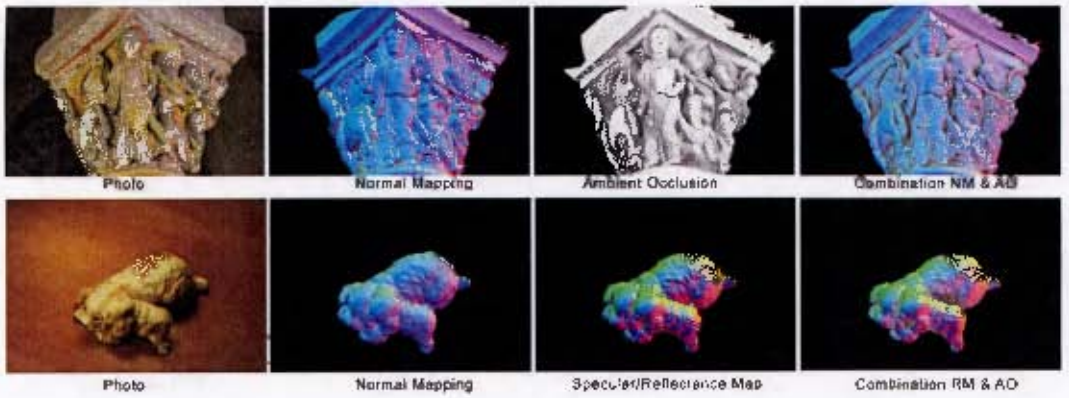


Figure 2.31: Similarity of photograph and renderings of the models with different shading techniques for mutual information analysis. Images adapted from (Corsini et al., 2009)

Recently, Corsini et al.[10] combined the Mutual Information approach with SfM-techniques for a fully automated registration pipeline of large image sets. In a first step, the images are oriented relative to each other with SfM methods. The obtained SfM model is automatically aligned with the laser scan model with an adapted version of the 4PCS-approach by Aiger et al. (2008), mentioned in section 2.2.3.4. To remove small misalignments during the SfM process, each image is individually fine-aligned with the Mutual Information approach, described above. In a last step the images are further globally fine-aligned with a graph based system. All images, which belong to the same node of the graph are projected onto the model and supposed to be a guide for the following iterative process, consisting of projecting and realigning each image until convergence. A similar approach, but still involving manual work, was developed independently by the author during this research, as described later.

Inaccuracies in the alignment or camera calibration, even though very small, can result in visible ghosting artefacts, which can be fixed with small warpings of the images (Eisemann et al., 2008; Dellepiane et al., 2011; Gal et al., 2010)

### 2.6.2 Colour Mapping

#### 2.6.2.1 Vertex Colouring

The simplest way to store colour information on a triangulated model is to assign a colour to each vertex of a triangle and interpolate the colour for the surface in between.

[10] M. Corsini, M. Dellepiane, F. Ganovelli, R. Gherardi, A. Fusiello, R. Scopigno - Private Communication with M. Dellepiane, November 2011

This so-called colour-per-vertex method is a simple and safe technique, as it allows the creation of simple import and export functions for processing tools, but the encoded texture detail decreases and increases with the resolution of mesh, and is fixed to the resolution set at the time of the projection (see fig. 2.32).

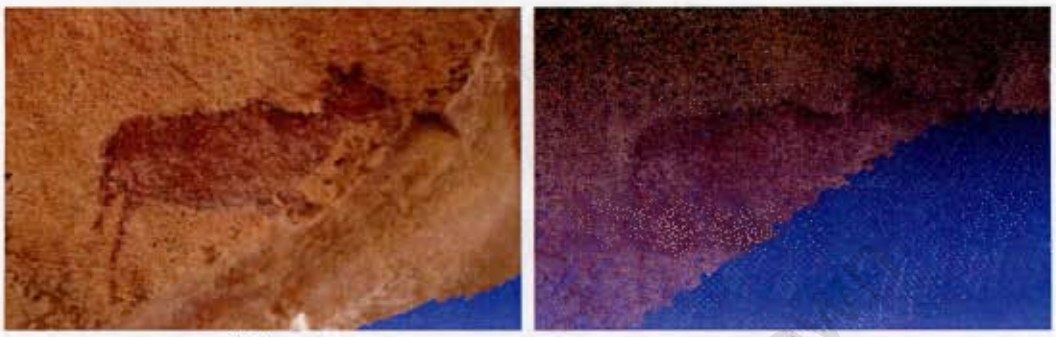


Figure 2.32: Limitation of the vertex-colouring method: the amount of visible detail depends on the density of vertices.



### 2.6 2.2 Texture Mapping

Another standard approach to assign colour information is texture mapping, or to use terminology corresponding closer to the above method, colour-per-triangle[11]. In this approach each triangle is linked to its corresponding area on an image. This is achieved by parametrisation, meaning to assign 2D image coordinates, also referred to as UV-Coordinates or texture coordinates, to each vertex of a triangle. This is a rather simple process if each triangle receives colour information from only one image. As soon as several images need to be blended together, an artificial map, a so-called texture atlas needs to be generated (see fig. 2.33b), which replaces the source images. This requires a much more complicated parametrisation of the entire mesh, if the meshes do not resemble any geometric primitives, such as spheres, cylinders or boxes. In these simple cases, a primitive projection (spherical, cylindrical, planar) ensures that the triangles are mapped distortion-free and receive the correct amount of pixel-area on the texture map, in proportion to the surface the triangles cover on the mesh. This is necessary to make sure that the amount of detail does not vary across the model. A simple per-triangle parametrisation does work, but it is preferable to have as many triangles connected together on the texture atlas to allow for manual editing in 2D

[11] This approach works also for quads and polygons. The title in here was chosen because the native mesh structure used in here is based on triangles

image software, such as Photoshop, and to make the texture atlas independent of the resolution of the underlying geometry.

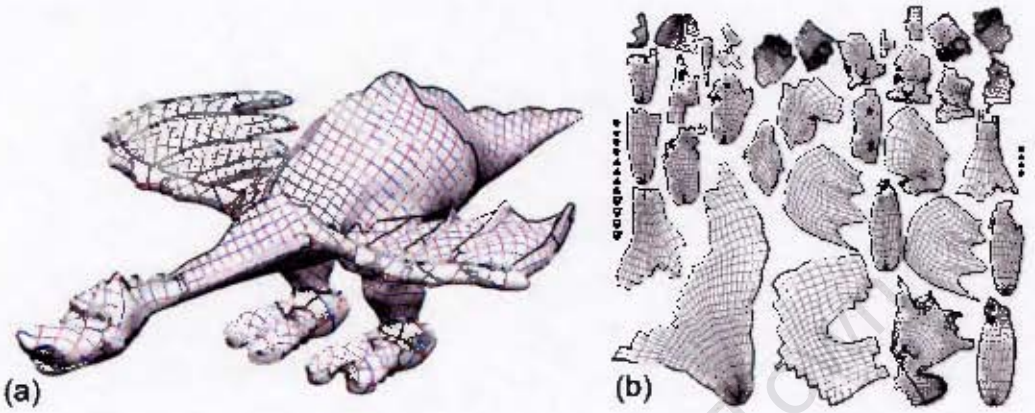


Figure 2.33: Parametrisation in the least-squares sense. (a) the segments of the models are illustrated by thick black lines, red and blue lines demonstrate the 2D parametrisation. (b) unfolding these segments on a planar map, a texture atlas. Images from (Lévy et al., 2002)

This widely researched topic of parametrisation is too complex to be discussed in here in detail, but to summarize, at present, the approach of Lévy et al. (2002) is still a quasi-standard for automatic texture atlas generation, also used in 3D modelling tools, such as Blender. They propose to segment the models along high curvature lines and unfold these segments on the image plane in a least-squares sense (fig. 2.33). It is interesting to note, that their established parametrisation solution is independent of the underlying resolution of the mesh, and thus could be generated on low-resolution models and applied to denser meshes.

Besides this complicated step of parametrization, it is also difficult to transfer the data between different software packages. Textures are only supported by a few file formats, and in addition, the ability of a viewer or editor to read and write the format does not necessarily include textures. For example, to the author's knowledge, the open source tool Meshlab is the only software which is able to read ply files with texture map support. In addition, texture maps need to be loaded into the graphics card memory and a large image set or high resolution texture maps might not fit into memory on low-end computers. Hence it is not guaranteed that the model and the textures can be viewed on other machines. For these reasons, a per-vertex-colouring seems much easier to manage.

### 2.6.3 Image Blending

When projecting multiple images onto a 3D model, a blending or weighting scheme for the images needs to be established. A simple criterion is angle of incidence (Lensch et al., 2000), where the triangle is assigned with colour information of the image, where the viewing direction is most parallel to the normal of the triangle. Another criterion is distance of the triangle to the camera or even a combination of distance and angle (Pulli et al., 1998). Callieri et al. (2008) propose a flexible framework for blending of textures with out-of-core capabilities for large image datasets. The underlying concept is that for each image, several grey-scale masks are created, each encoding different criteria, such as an angle mask, depth mask (distance), border mask (the distances from discontinuities in the depth mask and from image borders), focus mask and stencil mask. The latter is a manual mask, where the user can mark areas to be left out, such as unwanted objects in the photographed scene. The combination of all masks provides the weight for each image pixel (fig.2.34).

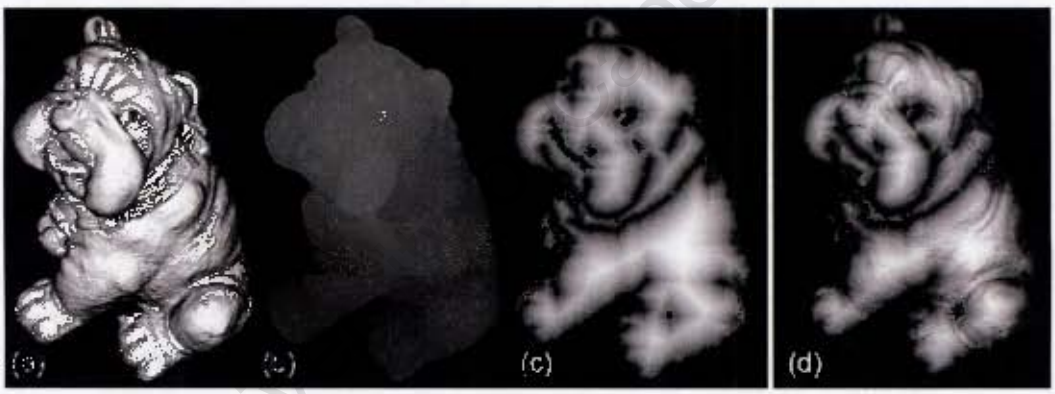


Figure 2.34: Masked Photo Blending. (a) angle mask, (b) depth mask, (c) border mask and (d) the combination of (a,b,c). Images from (Callieri et al., 2008)

Inconsistent colours can be prevented by using colour checker cards during the field campaign, which help to match the appearance of colours throughout a large set of images, possibly from different cameras, as suggested by Callieri et al. (2008).

Shadows however, are more difficult to remove from photographs as a simple brightening would also increase the colour noise and luminance noise. Dellepiane et al. (2010) suggested calculating the position of the sun at the time, the photograph was taken. This would allow the simulation of artificial shadows on the basis of a geo-referenced 3D model. The shadows on an image can then be masked out and be filled with colours from other images, where these parts are not in shade.

Debevec et al. (2004) demonstrated a more sophisticated method on the example of

the Parthenon in Athens, Greece, in which they suggest to calculate the correct colour reflectance values of the surface. This allows the 3D model to be rendered realistically under any lighting condition. To estimate the reflectance values, they identified a small sample surface area containing all different materials of the building (fig. 2.35c). During night-time, several images of this sample area were taken with a moving light source, which allowed the calculation of a bidirectional reflectance distribution function(BRDF), simulating the reflectance properties of all materials. In addition, they also took conventional images of the building, and projected them onto the 3D model, creating a texture map. While photographing the object, they also continuously took images of a set of reflective spheres, capturing the sky cover, the light intensity and the light direction (fig. 2.35a,b). During post-processing, the 3D model was rendered under the same lighting condition as at the time of the photograph. This grey shaded model and a textured version were compared and a reflectance map without shadows was estimated (fig. 2.35d,e,f). Each pixel of the estimated reflectance map was then replaced with the best matching material of the sample area and its corresponding reflectance function. This final reflectance map allowed very realistic renderings (fig. 2.35g,h).

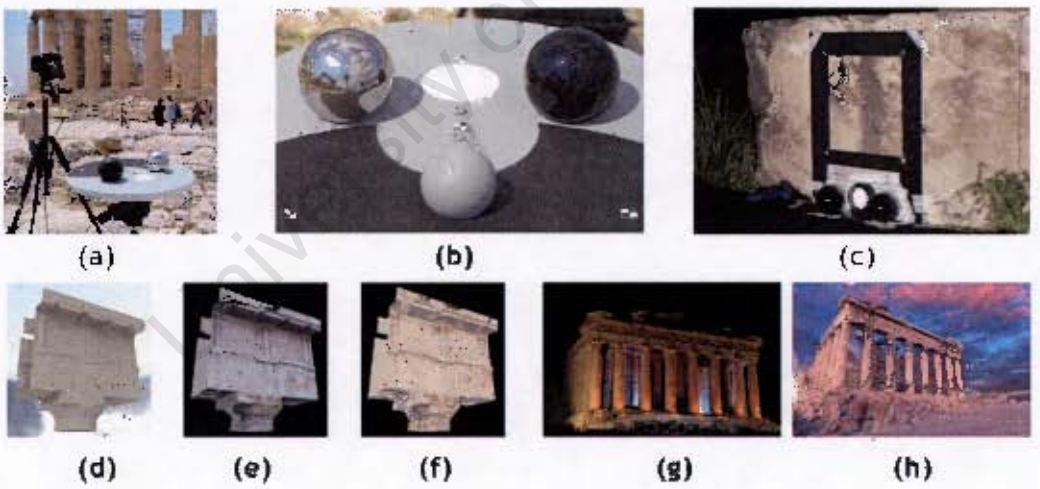


Figure 2.35: Estimating Surface Reflectance Properties of the Parthenon, Athens: a+b) set of reflective spheres recording the sky coverage, light direction and intensity. c) sample area containing most materials appearing on the building, d) sub-block of the model under the same lighting conditions as the photograph which was projected onto the 3D model in e). Image f) shows the estimated reflectance properties without shadows based on d+e) which are later replaced by best-matches from c). g+h) Examples of artificial renderings. Images from (Debevec et al., 2004)

Using BRDF seems to be the best documentation approach for cultural heritage sites, but the requirements make this approach much more time consuming than using standard photographs.

The next chapter will discuss and incorporate some of the approaches presented within this literature review into an efficient and flexible workflow.

# Chapter 3

# Establishing the Workflow

The central aspect of this work is to establish an efficient routine to create a coloured, triangulated high-resolution model of a scanned structure. In the following the order of the pipeline was reversed, now beginning with the output, a coloured surface model, because often the succeeding steps enforce some requirements onto the preceding parts of the work-flow. The only exception is hole filling, which largely depends on the choices made during surface reconstruction.

In each section the available methods are examined and discussed. Also, some new ideas on how to optimize the tasks are explored and the recommendations for the workflow are presented per section. In the next chapter, the established workflow will be verified by processing an entire dataset.



## 3.1 Pre-requisites

### 3.1.1 Computer Hardware

The experiments of this chapter were partly executed on an Intel Quadcore system with 8GB of RAM, Nvidia Geforce 8800 (512 RAM), Windows Vista, and partly on a Intel I7 system with 24 GB of RAM, 2x NVidia Geforce 590 and Windows 7. Both of these systems are normal work-stations, no server machines. For the next chapter, the verification of the found workflow, the latter machine was employed.

### 3.1.2 Software

To be able to create a workflow, which is directly usable and addressing various problems, such as file conversions and other compatibility issues, it is vital to mention which software was used. For the ease of the reader who plans to apply this workflow, it is thus usually referred to specific software packages as opposed to technical principles.

However, referring directly to specific software does not mean that there is no better implementation available or being developed. For a quick overview of the underlying technical principles of the employed tools, the reader is referred to the glossary.

#### 3.1.2.1 Commercial Software

It is the intention of this research to use as many freely available software tools as possible, so that the pipeline in its core is transferable to other projects as well. However, if not possible otherwise, these following commercial software packages are available to the Zamani Project:

- Innovmetric Polyworks 11
- Geomagic Qualify 8,9,10
- Leica Cyclone 7
- AgiSoft Photoscan
- Z+F Laser Control
- Trimble Realworks

#### 3.1.2.2 Meshlab

During all stages of the pipelines, problems do occur. Scans or meshes need to be cleaned of artefacts or problematic triangles, or need to be split into smaller parts. The open source software Meshlab[1], created and maintained by the Visual Computing Lab of the ISTI-CNR, is a powerful and versatile, freely available mesh processing tool and is often used within the following pipeline. It comes with a large collection of filters, for example, to clean, reconstruct, smooth, re-sample or simplify triangle meshes. It is available in 64-bit and can thus open and process very large meshes. Meshlab can also serve as a viewer which allows to measure on the model. However, at present even though very large meshes can be opened on a 64-Bit system, the viewing experience in these cases will be poor.

### 3.1.3 File-Formats

The conversion between file-formats is a time-consuming task, but no single format could be determined which is supported by all software packages employed in this

[1] `http://meshlab.sourceforge.net/`

workflow. At present, two file formats are found to be the absolute minimum, one for scan-data (PTX) and one for 3D-meshes (PLY).

#### 3.1.3.1 PLY

PLY, an open-standard file-format, was developed by the Stanford University for an easy and efficient storage of polygons or triangles. It is structured into a header, containing information about how many and which elements (vertex, triangle, etc.) and element-attributes (colour, normals, etc) are stored in the file.

The format proves to be very flexible, even for own additions and even though it was developed as a polygonal format it is now also often used to store point clouds. Because of PLY being an open source file format, it is quite likely to find converters to any other formats, or at least allows the development of own tools. The majority of the software packages, mentioned above, support the PLY-file format. Only Leica Cyclone is not very flexible regarding the in and export formats. As no other format is common to Cyclone and all others either, PLY is regarded as the standard in and output format for meshes.

#### 3.1.3.2 PTX

PTX was designed by Cyrax or Leica to store point data of a laser-scanner, such as the point coordinates (x,y,z), the intensity of the laser-beam reflection and colour (r,g,b). The ASCII-format enables customized converters and guarantees readability for the future. PTX allows storing a transformation matrix with registration information and retains the scanning grid, which is important for surface reconstruction (see section 3.4.2.1). Even though PLY files can also store point data, it seems more appropriate to store all original scans in a laser-scan specific format. LAS was developed as a standard format for laser-scanning data, but so far it is still lacking support. At present, PTX, introduced by Leica seems to be more suited for the task, as it is well recognized as a point-format in most laser-scanning software, such as Polyworks, Cyclone and Meshlab.

#### 3.1.3.3 File Converter

A command line converter from PTX to PLY is available, called ptx2ply, created by the Zamani-project, but partly based on source code of the free Scanalyze software[2]. Its main purpose is the triangulation of PTX-scan files to store the meshes in PLY-files.

[2]http://graphics.stanford.edu/software/scanalyze/

However, during this research, several modifications were done and conversion options were added to better fit the found workflow.

In addition, Meshlab can read a large variety of mesh-formats and could also serve as a converter to PLY.

## 3.2 Texturing

### 3.2.1 Manual Image Alignment

Even though the 3D modelling from laser scan data has matured and quite a few commercial packages are available on the market for registration and surface reconstruction, only very few offer texturing options for photographs taken with hand-held cameras, such as Leica Cyclone, Trimble Realworks, JRC Reconstructor, 3D Reshaper or Geomagic Studio. Most of these packages provide only very basic alignment tools, not optimized for large sets of images. The only commercial package of these, which is available to the project, is Leica Cyclone which offers to colour the point cloud[3] from individual images.

#### 3.2.1.1 Alignment with Leica Cyclone

Even though only four common points on the point cloud and the image are required to align an individual image, one can drastically improve the alignment result, by selecting more points. But finding the correct points on point clouds can be quite hard, as features are not well visible. Considering this, it becomes obvious that it is an enormous effort to do this with a large image collection.

For these reasons Leica recommends to use panoramic images, which are 360 degree, full-dome photographs, usually produced by taking 5-7 individual, overlapping images with a fish-eye lens, while rotating around the centre of the lens' entrance pupil[4] to minimize parallax issues. The images are stitched together and the output is mapped to (rectangular) image formats (fig. 3.1). For this procedure, different projections exist, such as the equirectangular/spherical projection. By projecting an equirectangular panorama onto a sphere, and positioning the viewer at the centre of the sphere, the user can turn around and experience the panoramic image without distortions. Besides a spherical projection, computer graphic applications often work with cube-maps, where the panorama is mapped to the six sides of a cube. As part of the creation process of

[3]Cyclone is able to colour meshes as well, but not able to handle such large meshes, generated within this research

[4]http://en.wikipedia.org/wiki/Panoramic_tripod_head

panoramas, the lens distortions are removed, allowing a better fitting alignment during the texturing process.


Figure 3.1: Illustration of the process to create panorama pictures. Images are stitched to a create an equirectangular/spherical panorama (a), or cube-maps (b)

Panoramic images can decrease the efforts drastically as only a single image needs to be aligned to colourize an entire scan at once. To cause no parallax issues, this method requires panoramic images to be taken from the scanners location, right before or after the scanning process. As the scanning usually takes places throughout the entire day, the additional panorama photography will cause blending problems as the lighting condition outside cannot be kept constant throughout a field campaign (see fig. 3.2).

Figure 3.2: Blending artefacts caused by moving shadows, when using panoramas, which were recorded under varying lighting conditions.

A limitation of Cyclone is its inability to accommodate for the occlusion of objects. Independent of the point visibility from the camera station, Cyclone applies colour to

all points of the currently selected point cloud. The only practical option for texturing is thus the texturing of individual scans with panoramic images from the laser scanners location. This however would affect the entire workflow as the texturing would need to be one of the very first steps of processing, before or after the cleaning and does not allow changes at a later step.

#### 3.2.1.2 Alignment with TexAlign and TextAlignSuite

A more appropriate texturing solution seems to be the research software packet of TexAlign, TextAlignSuite and TexTailor of the VCG lab, ISTI, Pisa, Italy, (Callieri et al., 2008; Corsini et al., 2009; Dellepiane, 2009; Franken et al., 2005) to which the Zamani project was granted access. According to the authors of the software of the VCG-lab, the packages are being implemented at present into the open-source software Meshlab. Similar to Cyclone, TexAlign allows the alignment of individual photographs by selecting corresponding points. For our purposes here, a rough alignment with four points on image and model are sufficient (fig. 3.3a). Unlike in Cyclone, this alignment serves only as an initial alignment.

TextAlignSuite, based on the mutual alignment approach (Corsini et al., 2009), described in section 2.6.1, can refine this initial, rough alignment in a semi-automatic process. The software tries to rotate and translate a virtual camera around the model and to estimate its focal length, so that a rendered image of the model from the viewpoint of this camera resembles the photograph best (fig. 3.3b-e). The user only needs to adjust the focal length in some cases and re-start the process if the result is not satisfactory.

The matching results depend on evaluating the joint histogram between the rendered image of the model and the photograph and thus on how similar the model and the image appear. The software offers two rendering techniques, such as normal mapping and specular mapping, which can be calculated in real-time, but still recovers many of features.

For further improvement, the software also allows the evaluation of already existing colour values attached to the model or a combination with the previous rendering or shading techniques. In most cases, a colour value will not be present, but another shading technique, such as ambient occlusion could be used for recovering more detail. Ambient occlusion darkens hidden areas and thus simulates a more realistic lighting condition. Because, at present, ambient occlusion cannot be rendered in real-time in sufficiently high resolution, it needs to be pre-calculated in other software such as Meshlab and stored alongside the model as vertex colour attributes.

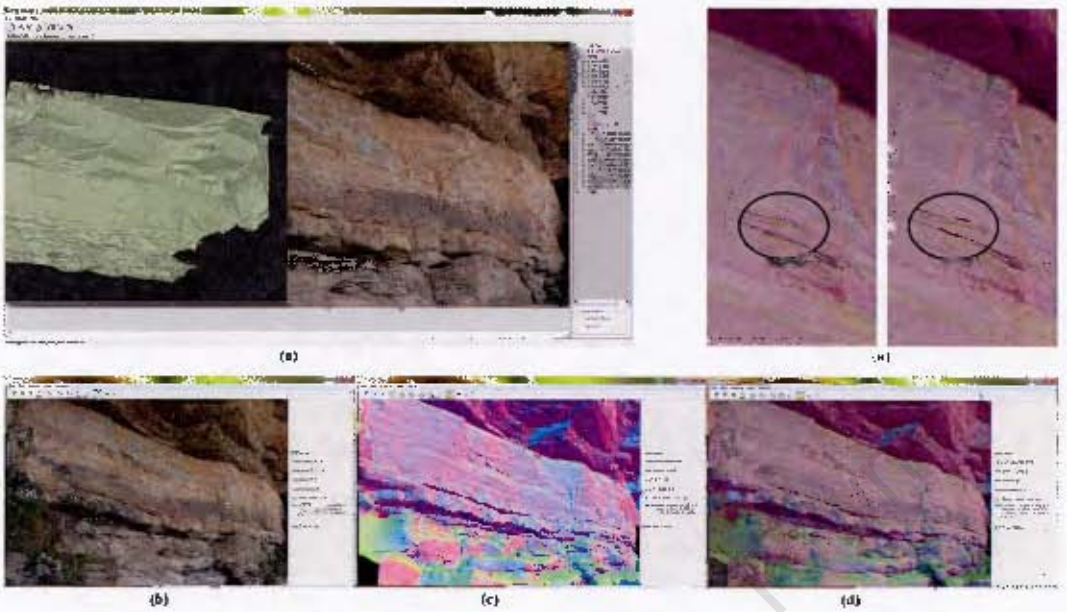

Figure 3.3: Overview of the texturing process with the VCG software. (a) Common points are selected in TexAlign to provide an initial alignment. Images (b-e) depict the fine-alignment process in TextAlignSuite: (a) photograph, (b) model, shaded with normal-mapping, (c) the photograph overlaid with the model, (e) difference of the alignment before and after the process.



Alternatively, the colour channel could be used to store colour values from an already aligned image and to use this projected version as a guide for the remaining images. For example, close-up photographs of fine rock art are usually difficult to align because they often depict only very little geometric features, visible on a laser-scan model. In these cases, an image taken from further away, which aligns well with the geometric features of the cave, should be used as a guide for a close up of the paintings (see fig. 3.4).

Figure 3.4: Using long-distance shots as a guide for the alignment of close-ups: a photograph (a) is aligned to a model (b) and projected onto it. The projected version(c) is then used as a guide to align a close-up (d)

A very useful, alternative guiding information is the intensity value, which the laser scanner acquired for each point. As these values reflect changes in the scanned material, a mapping of these values (usually 0-1) to grey-scale (0-255) provides useful additional information, independent of geometric features (see fig. 3.5). It is thus highly preferable, if the intensity information can be retained throughout the entire pipeline. A general disadvantage of the mutual alignment approach is its dependence

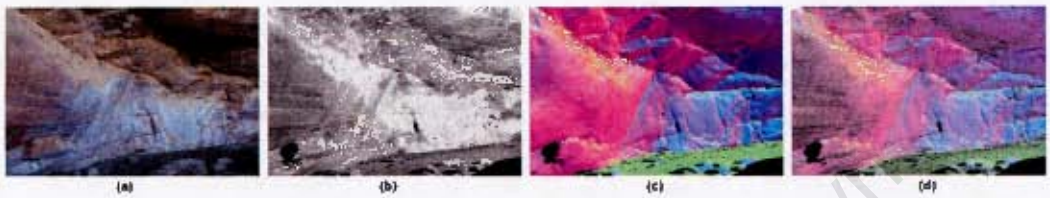


Figure 3.5: Similarity between photographs and intensity information of the laser-scan: (a) photograph; (b) intensity values mapped to greyscale; (c) rendering of the model with a normal-mapping-shader; (d) combined image of (b) and (c), which merges the geometric detail of the model and information about the material of the surface

on the resolution of the underlying mesh geometry, and it is recommendable to work on the highest resolution possible, but this will slow down the software. Working with only a subset of the model in high resolution, can speed up the process significantly. Hence, it is preferable to have the mesh split into sub-blocks during the surface reconstruction step.

#### 3.2.1.3 Image Projection with TexTailor

TexTailor is the last step of the VCG software suite and enables the projection and blending of the now aligned photographs, based on the principles described in 2.6.3. It allows the automatic calculation and application of a depth, border and angle mask for each image, which ensures that only the colour values from the best suited images are mapped onto each vertex of the mesh. It also provides an option to specify a stencil mask: a grey scale image, provided by the user to mask out certain objects, which are visible on the image but not on the model, such as people or bushes. This mask has to be manually created in external software, such as Adobe Photoshop or GIMP. It is recommendable to use a feathered brush along the edges of the masked area to avoid any harsh transitions in the blending between several images. TexTailor is able to project out-of-core, enabling the projection of large sets of images. It is also able to fill in small gaps in the texture coverage with interpolated colour values.

### 3.2.2 Texturing with Panoramas

Similar to the approach in Cyclone, it seems to be desirable to also use panoramas with the VCG's software suite. Using cube-maps, which are basically similar to images taken with a very wide-angle lens, the user would only need to align one image of the six sides of the cube. The camera position will remain constant for the other images, only the camera's orientation will change by a rotation of 90 degrees around two axes. For this purpose a little tool was programmed within this research, called PanoAlign, which reads the alignment (XML) file of one of the six cube-image (Top, Bottom, Front, Back, Left, Right), which was aligned with TexAlign/TextAlignSuite. PanoAlign then creates five XML files, one for each of the remaining cube-maps, by simply rotating the camera 90 degrees further. Due to unavoidable small misalignments of the initial cube-map, the other five images should be fine-aligned in TextAlignSuite via the mutual information approach. As opposed to the Cyclone approach, where panoramas can only be used to texture single scans and require to be taken from the scanners point-of-view, any panorama can be used with the VCG-software to texture any part of the model.

### 3.2.3 Using SfM for Image Alignment

Structure from motion could be a much more efficient way to align a large image dataset, if the overlap between images is large. The idea is to create a low resolution SfM model, and register it to the laser scan model and thus transform and align all camera stations found during the SfM process as well. The idea is based on Liu et al. (2006), but their software is not publicly available and performs best only in urban scenes. The open-source software Bundler[5] or similar tools, such as SfM-Toolkit[6] or VisualSfM[7](Wu, 2011), are well suited for this task. For the experiments within this research Bundler v.0.4 and SfMtoolkit v3 were used. SfMtoolkit is built on the source of Bundler, but supports 64-bit systems and works on the GPU supporting larger images and includes the executables of CMVS and PMVS as well, described later. VisualSfM is similar software, entirely optimized for high-end graphic cards and is likely to replace the other two software toolkits sooner or later, mostly because of performance reasons, but also because it comes with a graphical user-interface.

The commercial software Agisoft Photoscan, also available to the project, is an additional alternative to the free SfM-methods above and can fit nicely into any work-

[5]http://phototour.cs.washington.edu/bundler/
[6]http://www.visual-experiments.com/demos/sfmtoolkit/
[7]http://www.cs.washington.edu/homes/ccwu/vsfm/

flow, as it allows the export to the same output formats produced by Bundler and SfM-toolkit. Its main advantage is the easy-to-use graphical user interface.

The results of these packages can vary, for example, in some cases Bundler is able to align more images than SfM-Toolkit and sometimes it is the other way around, most likely due to their differences in the feature detection and matching algorithms. Because SfM-Toolkit is able to match more images in shorter time, due to its GPU support, it is recommended to prefer its usage over Bundler and to only use the latter, if results are poor. In the following, the term Bundler will be generally used for the process of bundle adjustment, independent if the original implementation is used or the one inside SfM-Toolkit.

Any image taken by a digital camera can be used with these SfM tools, as long as the sensor sizes of the various cameras are provided. Bundler and SfM-Toolkit already come with a large camera database, but often the user has to specify some cameras manually. Other parameters to be set are the amount of features, which should be extracted (octaves) and a matching sensitivity threshold. In most cases the standard parameters produce sufficiently good result and hardly need to be adjusted.

The output is a "bundle.out" file, an ASCII file, which lists the estimated focal length in pixels, two radial distortion coefficients, a 3x3 rotation matrix and a translation vector for each camera, and also a list of all matched 3D points.[8].

The matched 3D points are just a sparse point cloud representation of the scene. In an additional, optional step, the user can create a much denser point cloud with the open-source tools CMVS and PMVS. As PMVS is quite a memory intensive process, which can easily require more than 8GB of RAM, depending on the level of detail of the reconstruction and the amount of input images, it is recommended to use the clustering software CMVS, which partitions the reconstruction process into several, overlapping clusters, which can be processed sequentially, but automatically in a batch-file. The user only needs to specify how many images should be part of a cluster.

The output is a PLY file with a dense 3D point cloud of the scene. The point cloud can now be meshed via software tools, such as the Poisson tool in Meshlab. This model is in the same coordinate system as the sparse point cloud, including the camera positions, established by bundler.

[8]A detailed specification of the output format can be found online: `http://phototour.cs.washington.edu/bundler/bundler-v0.2-manual.html`

#### 3.2.3.1 Alignment of the SfM-Model

Aligning these SfM models to the laser-scan model could be done similar to an ICP registration of two scans, as explained in more detail later in section 3.6.1. The user would need to select a few common points on both models in a software application, such as Meshlab, which would roughly align both surfaces and provide scale via a simple 3D transformation. The ICP inside Meshlab or Scanalyze could then be employed to refine the alignment. The output would be a 4x4 transformation matrix, which could be applied to the external orientations of the cameras.

Alternatively, similar to Liu et al. (2006), one could align a few images to the laser scan model first, and, because these images are also part of the SfM process, a 3D transformation could be established between their positions in the laser-scan- and the SfM-coordinate system. This would allow transforming all other images of the SfM process as well. Liu et al. (2006) automatically aligned a few images by extracting and matching line features, which would not work well on collapsed walls of ruins and irregular, natural shapes of rock art caves, often documented within the Zamani Project. However, one could manually align a few images of the SfM dataset with the help of the Mutual Information approach in TextAlignSuite, mentioned above.

#### 3.2.3.2 SfM2Texture

To the author's knowledge, there is no software available allowing these SfM alignment methods. For this reason, a small tool was developed, called sfm2Texture, which, essentially, performs a 3D transformation between control points (fig. 3.6) and creates XML-files for each image part of the SfM process, which can be opened with TexTailor for projection, or with TextAlignSuite for fine adjustment, if necessary.

The 3D transformation used here is based on the concept of a quaternion transformation[9] which is a non-iterative method without the need of initial, approximate values. The source code was adopted from the java applet available under the same link as above. The output is a quaternion which is then converted to a rotation matrix, plus a translation vector and scale.

The input for sfm2Texture is a "bundle.out" file, as well as all images, undistorted by the Bundler/PMVS process. For the transformation process, it offers the user several options:

1. Matrix and/or scale - the user can specify a 4x4 matrix in a XF-file, established with, for example, Scanalyze. If the matrix is only a rigid-body transformation,

[9] `http://diegeodaeten.de/quaternionentransformation.html`

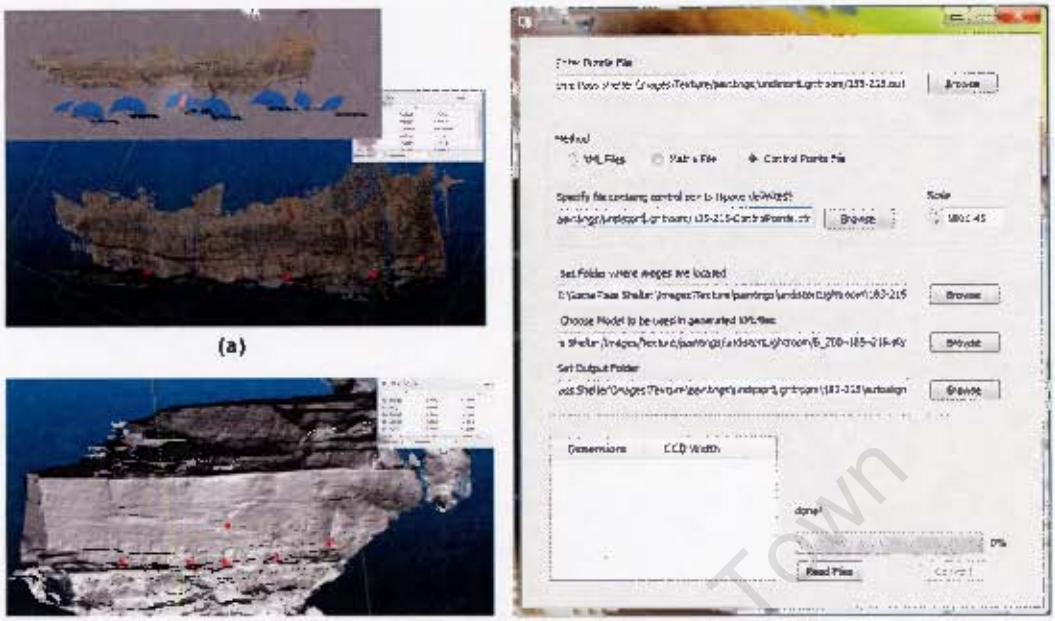


Figure 3.6: SfM-texturing workflow. (a, top-left image) the camera parameters for a set of images are aligned via SfM methods. (a, lower image) common points are chosen on the SfM model and the laser-scan model (b). (c) Screenshot of SfM2Texture

the scale can also be specified separately

2. Control points - the user can list control points, at least three, on the laser-scan model and the SfM model

3. Pre-Aligned Images - XML files with camera parameters produced by TextAlignSuite. At least two XML-files (two images) need to be specified to be able to estimate scale.

Option three uses the camera translation vector of the XML files and the according translation vector of the "bundle.out"-file as control points for the 3D transformation process. In addition, to get more accurate results with only a few files, the orientation of the cameras is also taken into account, which can be expressed by the three vectors defining the camera's coordinate system: "Look", along the cameras viewing direction, "Up", facing upwards and "Right", which is orthogonal to the other two vectors[10] (see fig. 3.7). These vectors can also be interpreted as 3D points and used as control points for the 3D transformation. Hence, with only two XML-files, the user has specified eight common points in both systems. But as there might be slight misalignment errors, it

[10] http://knol.google.com/k/matrices-for-3d-applications-view-transformation

is recommendable to provide more XML-files of more camera stations. For this reason, sfm2texture allows to run the transformation process incrementally. The user can start with two XML-files to estimate a rough camera position for all other images. Some of these new camera positions can then be refined in TextAlignSuite again, followed by another iteration of sfm2Texture, until the alignment fits well for all images. The user still has to open a couple of XML-files, but with each iteration of sfm2Texture, the mutual information approach of TextAlignSuite converges faster, as only minimal adjustments need to be done.


Figure 3.7: Images illustrating the three vectors defining the camera's coordinate system.

Experimenting with all three options, it seems that specifying a set of control points or a matrix refined by ICP produces best and fastest results. The XML-file option requires some time for manual loading, checking and refining the various alignments in TextAlignSuite, but is certainly an alternative, if the step of reconstructing a dense point cloud with PMVS was omitted.

### 3.2.4 Acquiring SfM-images

For best SfM results, it is recommended to take many images in small steps in a circular movement around the object as described in the Photosynth Guide[11](see fig. 3.8a), a strategy which could be confirmed on several tests within the Zamani project. If a circular movement is not possible, for example in the case of a large, straight wall, shots taken in a line parallel to the wall, produce very good results as well. In addition, to

[11] http://photosynth.net/help.aspx

increase the overlap, a second row of images should be taken a couple of steps further away (see fig. 3.8b). To go around a corner, even smaller steps should be made. The rule is to rather take more images than too few.

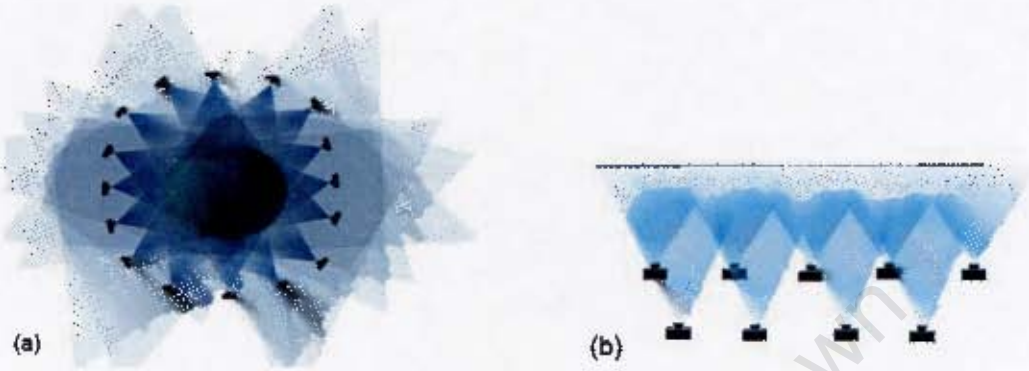


Figure 3.8: Recommended acquisition procedure for entire 3D objects (a) and for flat objects, such as walls (b). Source: Photosynth Guide v8 (http://cdn.photosynth.net/documentation/Photosynth%20Guide%20v8.pdf)

For very large sites a structured photography approach is vital. Each wall of a building should be done on its own and image sequence numbers should be noted on a map. Otherwise the user tends to walk around a site and take a "hand-held panorama" every couple of meters. This results in a very large image dataset, which would take weeks to be matched, as the matching time-complexity grows exponentially with every image added. Also, it is more likely that the entire SfM process fails, due to too little overlap, if the photographer lost track of what was actually captured.

If it is necessary to reduce the amount of images, just orthogonal, stereo shots produce sufficient results as well. But it proved to be useful, to also take a couple of images from further away, covering most of the surface in one shot. These additional shots serve as a base where the other images are matched to.

### 3.2.5 Texturing-Results

The alignment process with TexAlign and TextAlignSuite is much faster and allows much finer alignments than a process only guided by common points on model and image. Most of the remaining alignment errors are due to lens distortions, which are not being compensated with this approach. However, this manual process might work well on small sites or objects with less than 20 images, but it will still take too long for more complex sites, requiring easily over 100 images. Panoramas can greatly decrease the necessary image alignment task and also fit the model well, because the radial

distortion was removed as part of the panorama stitching process. However, as the creation of panoramas also takes some time and requires a special tripod, it seems to be an enhancement of the process, but not a replacement for the texturing with hand-held images.

The proposed SfM-approach is the most promising solution for large datasets, if enough overlap is present. It would also take care of misalignments due to the lens distortions. However, there is still a lot of room for further improvement. Inaccurate estimation of the camera parameters leads to distortions of the model and thus it will not coincide well with the laser-scan model. A promising solution however was proposed by Corsini et al.[12] who suggest an automatic fine adjustment of each camera position with the mutual information method. In principle, their SfM approach is similar to the one proposed in here, but their method is entirely automated, including the alignment of the SfM-Model to the laser-scan-model. The authors announced that the technology will be implemented into the open source software Meshlab.

An alternative solution, proposed here, but beyond the scope of this research, is to modify the SfM matching algorithm to also match rendered images of the laser-scan model, enhanced with greyscale intensity information. Because these additional images are rendered images, their cameras' in- and external orientations are known and could serve as a reference for the un-calibrated images. In addition, the images of the SfM process would be registered automatically into the same coordinate system as the laser scan model.

Apart from the alignment, no software solution is known to the author to globally colour-match all images (see fig. 3.9). In general, this problem could be prevented by using colour checker cards, as suggested by Callieri et al. (2008), but no other post-processing solution could be found.

### 3.2.6 Resulting Workflow - Texturing

SfM is clearly the easiest way of aligning a large entire image set and thus should be the first option to explore. It might not be possible to texture the entire model at once, but even a SfM-model of sub-parts can be very helpful. To create the SfM model basically any SfM-software can be used with sfm2Texture, if it produces the required output-files, similar to Bundler. Agisoft Photoscan can also create these files and can thus be employed instead. If the SfM process fails, it is worthwhile to try other tools

[12]M. Corsini, M. Dellepiane, F. Ganovelli, R. Gherardi, A. Fusiello, R. Scopigno - Private Communication with M. Dellepiane, November 2011

Figure 3.9: Image demonstrating the need for a global colour-matching method of all photographs used for texturing

as well, as the quality of the SfM-models vary from case to case between software packages. No clear recommendation can be given.

The easiest solution to texture with SfM, at present, is to run SfM-toolkit or Agisoft Photoscan on a set of images and select a few control points on the sparse SfM model and the laser scan model, for example in the free software CloudCompare[13], which allows to load and work on point clouds. For the alignment, the control points (four to six points) can be selected quite roughly. Experience with several test-cases showed, that the SfM-model is often distorted (fig. 3.10) and thus, even a fine control point selection would not perfectly align the two models either. Hence, it is recommended to rather fine-align the needed images with TextAlignSuite again before they are projected with TexTailor.

Obviously the effort of going through producing a SfM model is only worthwhile from a certain amount of images onwards. In case of less than three to four images, it will most likely take the same time to go through a manual alignment process.

To fully document a small room with Structure-from-Motion, a large amount of images are required, even with a wide-angle lens. Besides that, it is unlikely that these rooms can be linked accurately to a larger SfM-model of the entire building, as the connection is usually a narrow door or passage. Hence, often sub-SfM-models for each room might be necessary. For these cases it might be worthwhile to use panoramas of the rooms for the texturing instead.

[13] http://www.danielgm.net/cc/

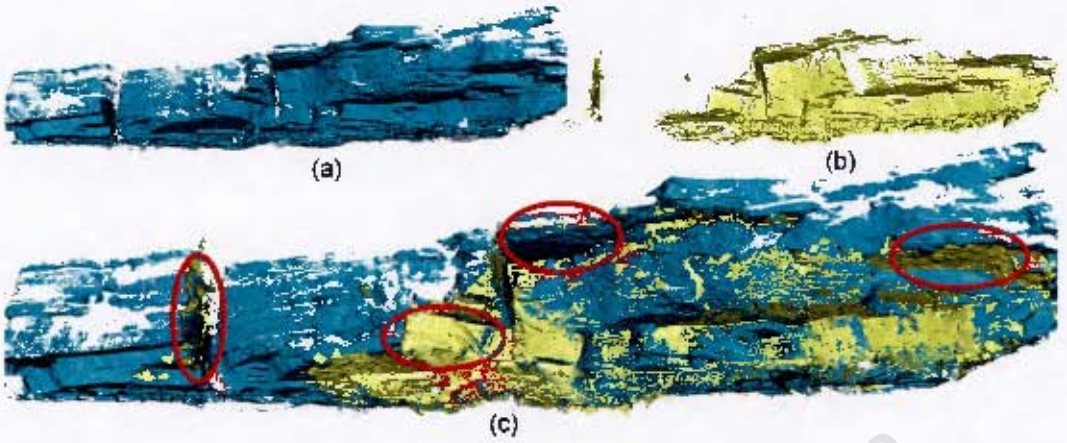


Figure 3.10: (a) Blue model is based on laser-scan data, (b) yellow model is a reconstruction via Bundler and PMVS. (c) Both models were aligned with ICP. The mismatches can only be explained by inaccuracies and non-uniform distortions of the SfM-Model.

An interesting side to texturing with the VCG's software suite is the ability of combining automatic- and manual methods: A large portion of the model can be textured with the SfM-approach, and wherever needed, single images or panoramas can be aligned manually with TexAlign and TextAlignSuite to fill in gaps in the texture. In both cases, automatic or manual, XML-files are created, which can be read and processed at the same time with the same software, TexTailor.

Please refer to figure 3.11 for an overview of the workflow for this section.

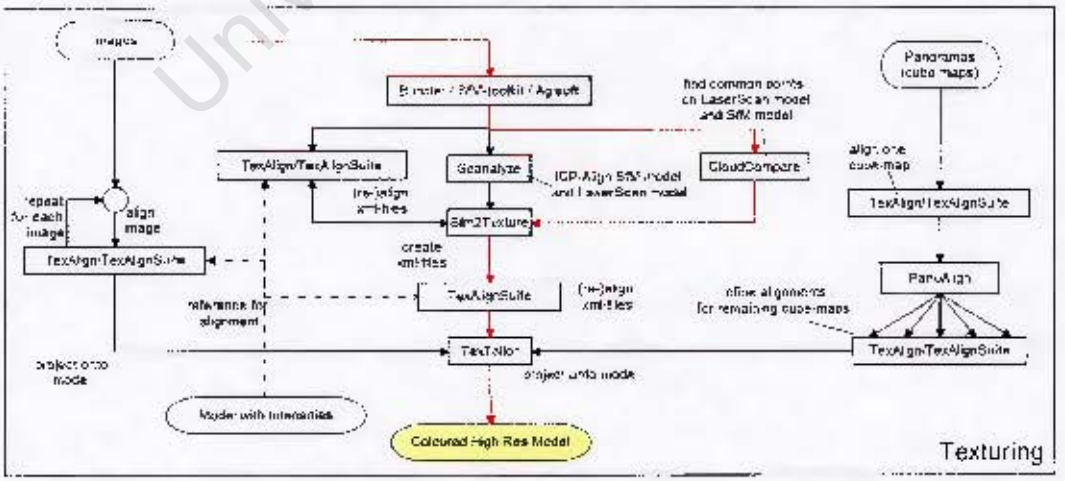


Figure 3.11: Diagram of texturing workflow. Preferred routine in red, if possible

## 3.3 Simplification

The viewing experience in 3D model viewers and editors is depending on the size of the model and on the available computer hardware. It is thus essential to reduce a high-resolution model to various lower resolutions, accommodating the need of different applications. A good compromise between quality and a smooth navigation experience, even on slightly outdated dual-core machines, is a model consisting of three to five million polygons.

The Quadric Error technique of Garland and Heckbert (1997) became a widely used standard simplification algorithm. It simplifies the mesh with respect to the error, which is introduced by removing triangles/vertices. Because Quadric-Error decimation is a rather slow and memory intensive technique, it is not optimized for very large meshes. For these cases, it is wise to also consider a vertex clustering technique (see section 2.5.3), which simplifies the model based on a regular grid, removing all but one vertex of each cell. As detail deteriorates quickly with the grid size, this method is usually used to reduce a very large model to a level manageable with Quadric Error, as suggested by Lindstrom (2000). The clustering decimation needs only little additional memory space, when the model is loaded, while Quadric Error requires more memory with every edge being collapsed. 64-Bit software (e.g. Meshlab) and larger RAM resources however, allow larger meshes (> 10 million triangles) to be simplified.

If not enough memory is available to load and simplify the entire model, but at least, the model is split into sub-blocks, an alternative is to simplify each block individually with a Quadric Error technique. If the blocks match up along the boundaries, it is wise to enable an option to preserve the boundaries and to reduce the blocks based on a maximum error, rather than a triangle count. This is very important, as detail can vary drastically between blocks. Reducing them all to the same number, or to a certain percentage of the original model size, can result in different appearances, some blocks smoother than others. Tridecimator, a tool part of the VCG library (open source) allows to set such a maximum error, however, the interpretation of the value is not very intuitive for the user. To obtain the correct value, it might help to reduce one block to the desired size and use the error value at which the algorithm stopped for the remaining parts of the model.

Holes can strongly influence the quality of the result. When simplifying sub-blocks, the boundaries of each block need to stay intact, which also means that triangles around any hole will remain unchanged. Thus, to reach the targeted amount of triangles, the algorithm needs to reduce triangles somewhere else, eventually also in important, detailed areas (see fig. 3.12, middle row). As a work around, it is recommended

to reduce the sub-blocks only to an amount which allows the merging to one single model, which can fit into memory again and be simplified as a whole model, without the restriction of keeping boundaries intact.

### 3.3.1 Resulting Workflow Simplification

The open source software Meshlab offers both techniques for decimation, Quadric Error and Vertex Clustering. As the latter can produce topology changes it is recommended to employ the Quadric Error method, if enough memory is available. No significant complications, other than memory limitations could be found in the newest 64-bit version (1.3.1). For a small comparison between these methods, please refer to figure 3.12

Also the subdivision approach in connection with the command line tool Tridecimator worked without complications. For cases, where the boundaries need to be preserved, it is suggested to fill at least very small holes during the hole-filling or surface reconstruction step.

Out-of-core simplification tools are still rare. Polyworks has such a function onboard, but simplification experiments with very large meshes were not successful. In future, with progress in the development of viewers and editors, optimized for streaming or progressive meshes, the user will not have to worry about simplification any more. For example, first demos of Nexus, a VCG-lab development based on the Adaptive Tetrapuzzles approach by (Cignoni et al., 2004), is capable of displaying meshes in the order of several hundred millions of triangles.

Please refer to figure 3.13 for an overview of the workflow for this section.

## 3.4 Surface Reconstruction

Surface reconstruction or meshing is the technique of extracting a continuous surface from discrete input data. Even though algorithms might create a parametric or implicit surface internally, tools implementing these approaches and used for processing usually output a surface, represented by polygons. In computer graphics the most common entities of polygons are quads or triangles, defined by vertices and/or edges. In the following, it is focused only on triangles, also called faces, as they are still the de-facto standard for 3D computer graphics. The future seems to lie in Quads (or maybe only points), due to their advantages during simplification, texturing and viewing. Tools however, producing and processing Quads are still rather rare.

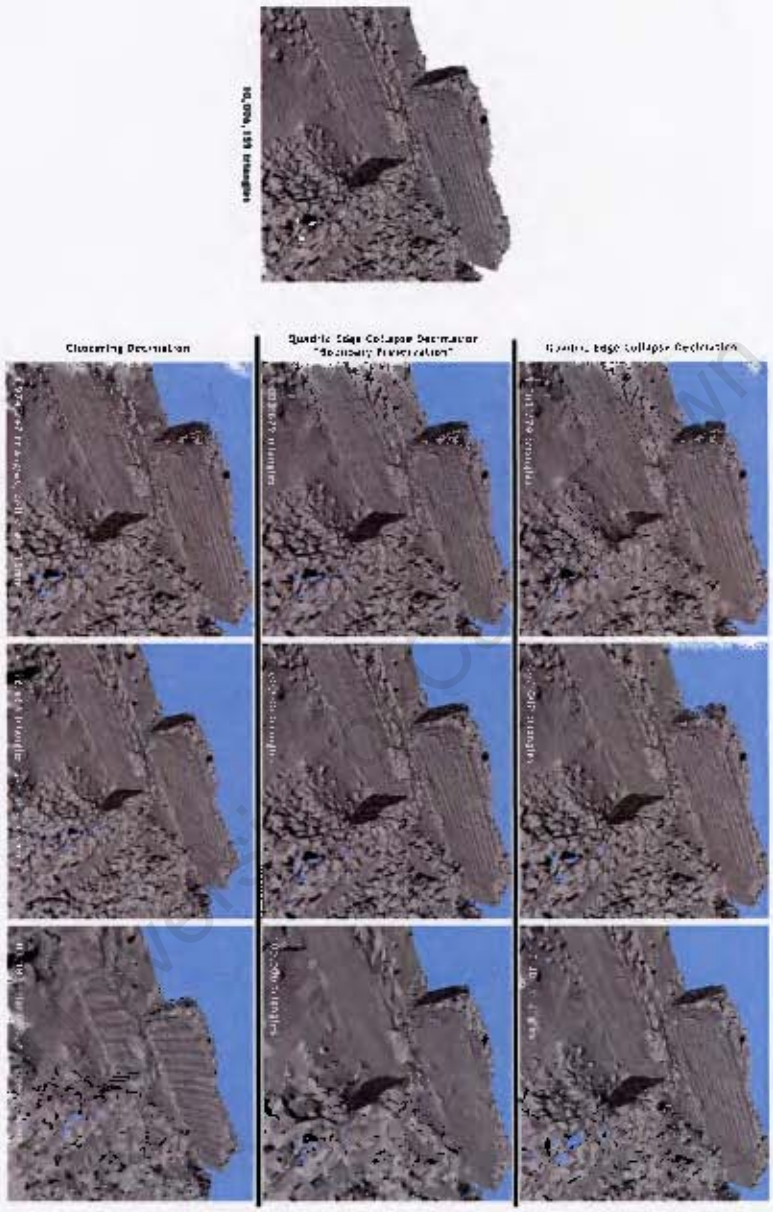


Figure 3.12: Comparison-results obtained with Meshlab v1.3.1 for the following filters: "Quadric Edge Collapse", "Quadric Edge Collapse" set to preserve boundaries, and "Clustering Decimation". As clustering decimation works with cell sizes and not with a target-amount of triangles, the cell size was set to produce approximately the same amount of triangles, used for the other two methods

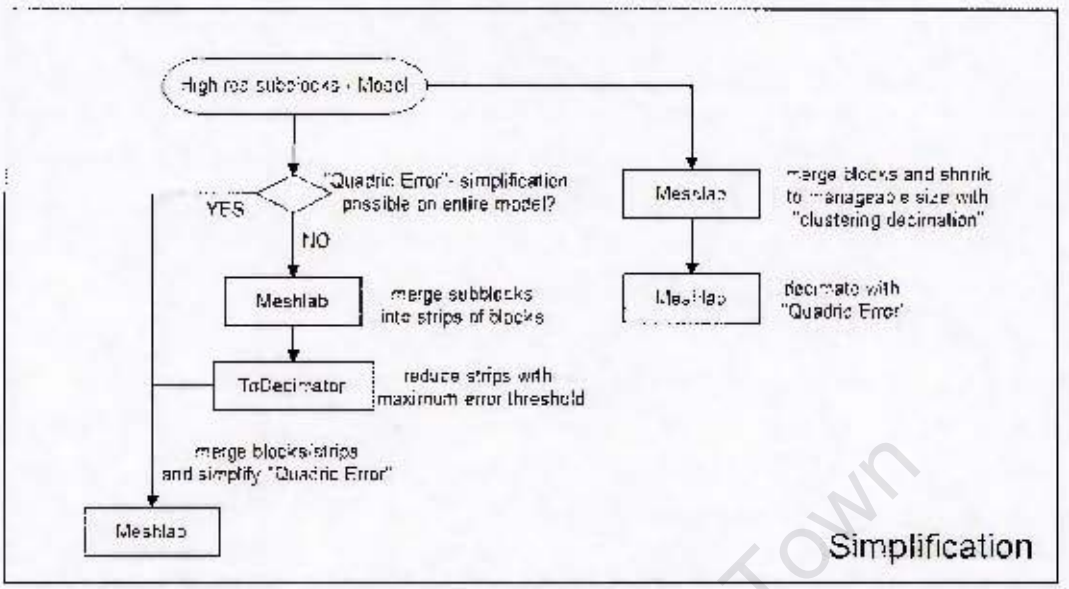


Figure 3.13: Diagram of simplification workflow.

Manifoldness and non-intersection of triangles is an important criterion when creating a surface, as many algorithms and software packages cannot handle complex surfaces when editing them. If possible, manifold outputs are preferred.

Meshing algorithms can be classified in many ways, but since the main objective of this work is to find an efficient work-flow it seems obvious to do it based on their input data structure. Considering this, there are two classes of algorithms. Either they are merging individual scans, usually requiring the scans to be triangulated already, or extracting a surface from a single point cloud, including all scans.

### 3.4.1 Tessellation Depth Problem

One of the most essential problems in meshing, besides for watertight reconstructions, is here referred to as the tessellation depth problem, or short TDP. It is a threshold defining which distance between two points is acceptable to form an edge as part of a triangle and from where on a gap is too large to be bridged, leaving a hole in the model behind.

### 3.4.2 Prerequisites for Meshing Algorithms

#### 3.4.2.1 Pre-Triangulation

As mentioned before, some algorithms or software tools require the input to be in triangulated form. This allows them to either work on the triangles directly, merging

them together, or, to extract more information about the vertices, such as normals. Pre-triangulating scans is not a difficult task, as long as the scanning grid is still intact. Every laser-scanner sweeps across the surface in lines and records a certain amount of points per line (vertical axis). This can be thought of as a matrix- or grid-like structure, listing all points in the order in which they were scanned, line by line. If the laser beam does not hit a surface point, nothing is reflected back. These cases, here referred to as "empty points", also have to be accounted for to keep the grid intact. This structured representation is also often referred to as a range image, storing a depth value for every point in the x,y grid. The regular grid structure allows connecting point neighbours to triangles, as the closest points are always the eight direct neighbours in the grid (fig. 3.14). If the scanning grid is lost, each scan needs to be treated as an unstructured point cloud, which requires a more time-consuming, full meshing process.

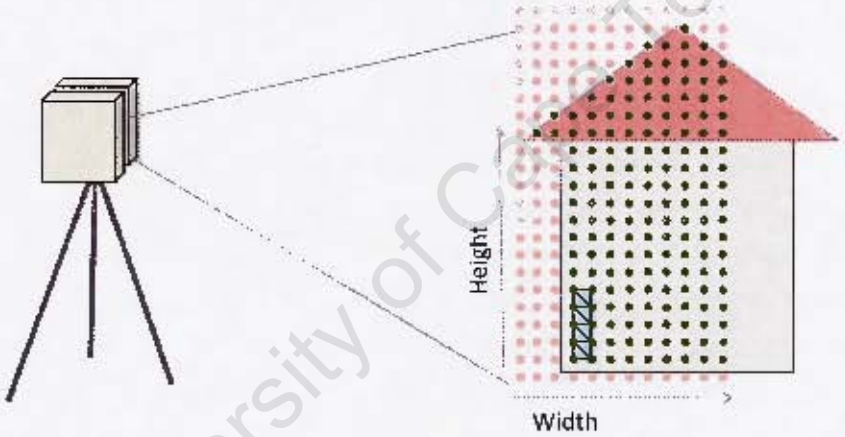


Figure 3.14: The diagram depicts data samples (green points), which are collected by the instrument in a "gridded" or matrix structure, defined by the amount of lines and the amount of points per line. Points without a return signal also need to be accounted for, to maintain the matrix (red points). This matrix representation simplifies the search for point neighbours and thus the triangulation drastically.

The main disadvantage of triangulating individual scans is the tessellation depth problem (fig. 3.15), which is more dominant in this case than in other meshing algorithms, which work on a single point cloud. As the resolution of the scan drops with further distance of the surface to the scanner, triangles on the outer areas of the scans will be quite large. Hence, the points in the outer regions are less likely to be connected. If, however, the whole dataset is considered, combining points from all scan positions, it might turn out that these, otherwise rejected points, have close neighbours from other scans, which could be used to form more triangles.

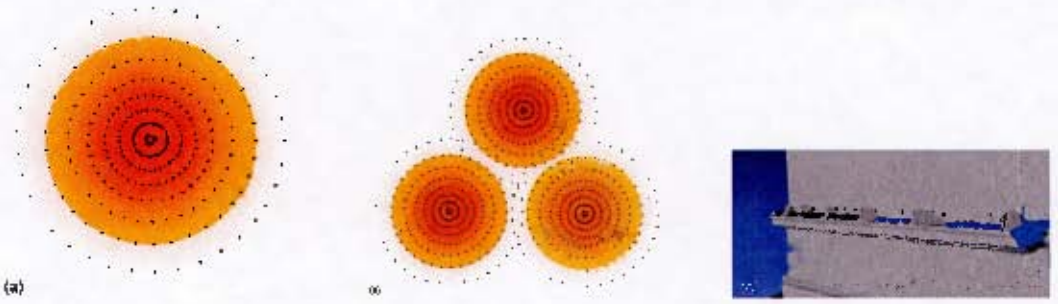

Figure 3.15: (a): the point density of a 360 degree scan, which decreases with the distance from the centre. The TDP is obviously more prominent on the outer areas, and thus, the outer points usually will not be triangulated and rejected. (b) the previously rejected points could provide useful information and fill in gaps when combining data from other scans. (c) Artefacts, which appear when the tessellation threshold is set too large.



#### 3.4.2.2 Point-Normals

As opposed to point-coordinates and colour values, the normal of a point is indirectly determined by its point neighbours or by considering the normals of all triangles, incident on this vertex. It is quite easy to derive them from single, pre-triangulated scans. However, their retrieval then also depends on the tessellation depth threshold: as no triangles will be created in sparser areas of the point cloud, no normals for these points will be calculated either.

The only alternative is to extract the normals from the whole dataset, which, however, is not trivial, besides maybe the spatial sorting of the points for faster access, which is straightforward but time-consuming. The major problem arises from the noise present in the data, which leads to incorrect determination of point neighbours. If the wrong points are paired, the orientation of the normals cannot be determined and often the incorrect orientation is propagated throughout the rest of the dataset. Also alignment errors and vegetation can be regarded as noise. Thin grass blades for example will often only be covered by one single point, which makes it practically impossible to find any correct pairing at all. One solution is to extract the normals on a sub-sampled dataset and interpolate the missing normals on the high-resolution data, which has a smoothing effect, and will thus influence the level of detail in the final model.

### 3.4.3 Review of Applicable Meshing Algorithms

Meshing is a well studied topic and there are many different types and adaptations, as shown earlier. For this research, only commercial packages available to the project were considered and the more recent research software tools, available online, which can handle at least several million points. The selection was narrowed down to five representatives of well established approaches. Better implementations or workarounds which reduce the problems described below might exist, but are not known to the author or could not be tested. It is however attempted to keep the workflow as flexible as possible, so that the individual meshing tools can be replaced by better techniques, if they become available.

The tested software packages are:

- Geomagic Qualify
- Octree Merge (OM)
- Poisson Reconstructor
- PlyMC
- Zipper

The main criteria for this comparison are:

- preservation of detail
- the effects of noise in the scans
- the effects of misalignment of the scans during the registration step
- software stability
- completeness

#### 3.4.3.1 Reference data set

To meaningfully compare the different meshing solutions, a representative test case was created, by extracting a small area from the Gereza dataset, a Portuguese fort on the island Kilwa Kisiwani, Tanzania. The top of the NE-tower (fig.3.16) features typical, problematic scanning-situations, especially true for 360-degree, full-dome scans:

- Non-uniform sampling densities - The tower and its platform can be seen from many instrument setups, from various distances.

- Flat angles of incidence - as the tower platform can be seen from further away, but not from higher places, the angle of incidence of the laser beam from most positions will be quite flat.

- Under-sampling, data voids - the structure is made from coral and often mortar has disappeared, leaving irregular, uneven surface behind, which cannot be scanned from all sides.

- Vegetation - in some places, the surface is covered by grass. Although this is very little vegetation compared to other projects, it is likely to be enough to show how the different approaches respond to vegetation.

Figure 3.16: The NE-Tower of the Gereza

For the conversion from scan data into the various file formats, required by the following meshing tools, the software converter ptx2ply, mentioned earlier, was modified. With its help, the test case, a subset of the entire dataset, was extracted. For output formats, which require being in triangulated form (PlyMC, OM) or the estimation of point normals (Poisson), the tessellation threshold was set to 5cm. In the case of Poisson, all vertices with normals, where the viewing angle towards the scanner instrument is larger than 85 degrees, were ignored. OM is a MLS tool and therefore requires normals, but the software is extracting them from triangles on its own.

The target resolution of the reconstruction process was set to 1 cm, or as close to it as possible. Some software tools, such as Poisson and OM are using octrees to spatially structure the input points. The resolution is thus depending on the cell size of the corresponding octree depth, chosen for extraction.

The Delaunay surfacing tool in Geomagic, allows setting a point spacing. This could lead to a visible grid pattern, which could affect the quality assessment. The feature has thus been turned off.

#### 3.4.3.2 Delaunay

Delaunay methods can work on unstructured point clouds. No additional information than the point coordinates themselves is usually necessary. This makes these tools very powerful and universal. They are able to consider every point from all scan positions, reducing the tessellation depth problem to a minimum. Hence, the final model will be more complete than with other methods, working on individual scans.

Geomagic Qualify provides a "Wrap" tool, which allows the meshing of unstructured point clouds and is known to be related to Alpha shapes (Edelsbrunner and Mücke, 1992), which fall into the Delaunay category. It offers two options, "surface" and "volume", of which only the first one is used here.

Figure 3.17: Delaunay model with Geomagic, full dataset, no noise reduction



Experimental Results The resulting surface of the test dataset is disillusioning. The surface appears noisy, with many holes, intersecting and non-manifold triangles, causing serious surface orientation problems (fig. 3.17).

The strongest pro of Delaunay variants, their ability to process unstructured vertex data, seems also to be a weakness. As point neighbours need to be established from scratch, noise can affect that search drastically. This also affects the surface orientation, which can thus easily flip.

These problems often occur in noisy, convoluted areas. It is suspected that instrument noise and alignment errors, are negatively affecting the Voronoi diagram and thus also the search for correct point neighbours. As can be seen in figure 3.19, a wrong triangulation causes a helix or spiral shaped surface. The orientation of the surface

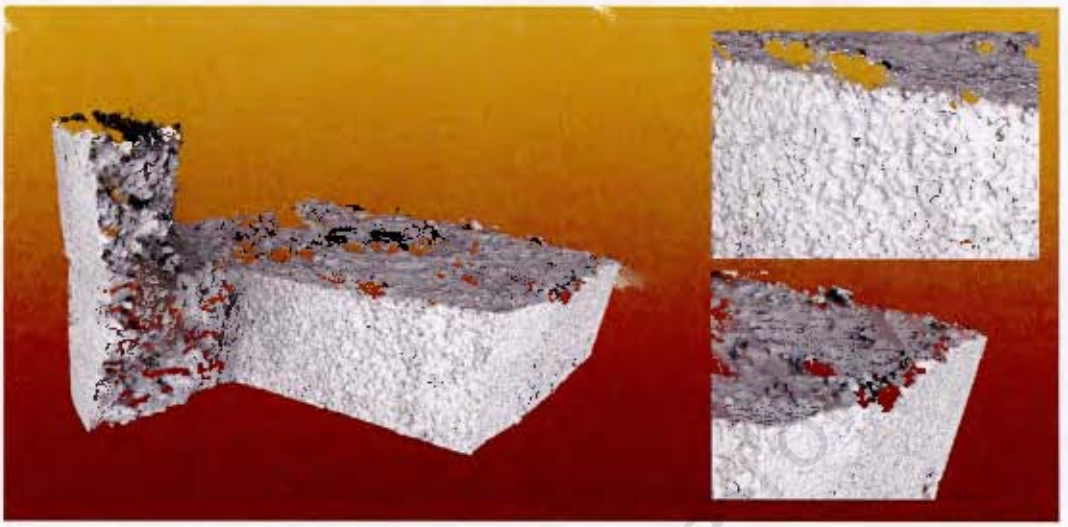

Figure 3.18: Delaunay model with Geomagic, full dataset, maximum noise reduction



Figure 3.19: Example demonstrating the effect of noise and misalignment on the orientation of the surface normals after a Delaunay triangulation. (a) The black spots are problematic triangles, where the surface normals point away from the viewer. (b) If a false edge connection between points is made, the surface orientation, illustrated by either a bright or dark yellow colour, can flip, even though the orientation is correctly propagated from triangle to triangle (c).

thus flips, despite a correct propagation throughout the geodesic neighbourhood of each triangle. The software allows setting a noise filter to "off" (fig. 3.17), "minimum", "medium" and "maximum" (fig. 3.18). Activating this filter results in a much more complete surface with less orientation problems of the surface normals, but still no perfect result is achieved. Even with the noise filter on maximum, there are still areas where the surface orientation is flipped. This suggests to apply several noise filtering steps to the dataset beforehand and/or to sub-sample the dataset to an amount below the noise level.

Geomagic Qualify V10, does not support the surface reconstruction of arbitrary large meshes. According to Dey et al. (2001, pg.1), the space and time complexity of the Delaunay triangulation can be quadratic with respect to the number of input points, which can be seen by the computation time and the amount of RAM needed. As a consequence, Dey et al. (2001) employs an octree to reduce the complexity. Following this example, to manage large datasets with Geomagic, one could subdivide the original dataset into smaller, regular blocks and process them individually.

#### 3.4.3.3 Moving Least Squares

MLS approaches, represented here by OctreeMerge (OM, developed by Fiorin et al. (2007)) also work on the point cloud, but usually require pre-determined point normals. To circumvent orientation problems in the determination of point normals, it is preferred to extract them on a per-scan basis, accepting the fact that the result is more affected by the tessellation depth threshold, as described in Section 3.4.2.2. In this case, however, it is quite safe to use a very large threshold, because only normals of the vertices are used, not the triangulation used for their estimation. The impact of a few inaccurate point normals will be small when merged into the whole point cloud. If there are no additional points from other scans in that area, it provides the algorithm with at least precise point positions and some vague point normals to create a surface. Single, isolated points will usually be disregarded as the algorithm needs at least a couple of close points to fit a surface to.

**Experimental Results** The surface, created by OM is very detailed and reasonably complete (fig. 3.20). Compared to the PlyMC result however, the holes seem to be larger. This effect however seems to be related to the cell size of the octree, which, in this case, equals 7mm (level 10). Another surface which was created at level 9 (14mm) appears to be much more complete (fig. 3.21).

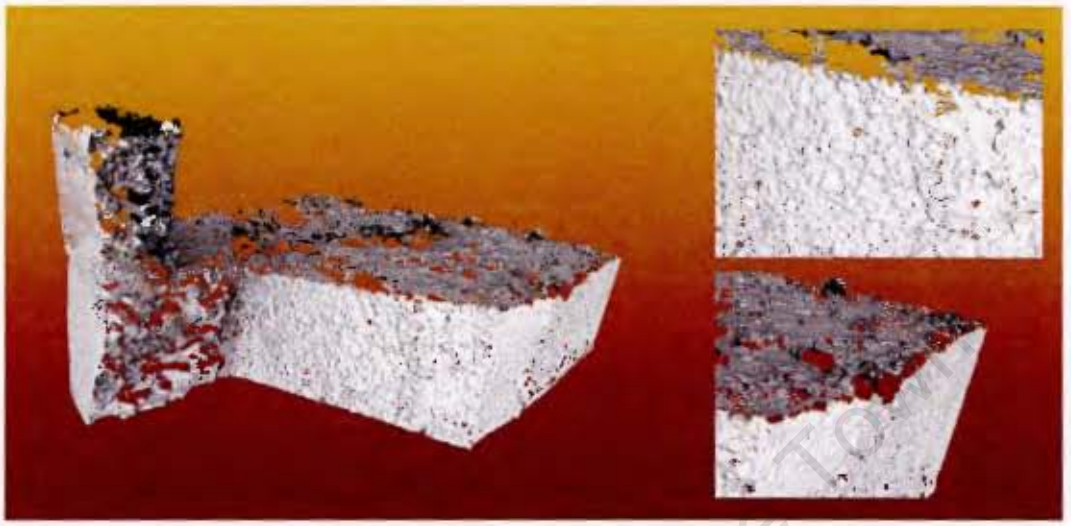

Figure 3.20: Result Octree Merge, Octree Level 10 which equals a cell size of 7 mm

Figure 3.21: Result Octree Merge, Octree Level 9 which equals a cell size of 14 mm

OM allows setting a threshold for the amount of points in the proximity of the current point, which are used for the fitting of the surface (the standard is 15 points). The more points are used, the smoother the surface will become, and thus this option serves as a noise filter. The standard setting seems sufficient for most cases. The smoothing works well on vegetation. While detail is preserved in the rest of the model, the tips of grass blades, usually only sparsely sampled, are treated as outliers hovering above the real, more densely sampled surface. The result appears as a slightly irregular surface, still recognizable as grass surface, but without large spikes. However, it is often also vegetation, which leads to program crashes, as experienced in other cases. Without these stability issues, OM could be also used for large datasets, because it is based on an out-of-core approach, with the ability to process sub-blocks of the dataset.

#### 3.4.3.4 Poisson

The power of the Poisson approach lies in the creation of detailed surfaces, which are also watertight. Kazhdan et al. (2006) released free software, which was also implemented into Meshlab. For large datasets, they added a streaming version (Bolitho et al., 2007), which builds the model slice by slice. In the following this streaming-enabled version was tested. It also relies on an octree data-structure and thus the output resolution is determined by the cell size of the corresponding octree-depth-level.

Experimental Results Comparing the result (fig. 3.22) to other approaches, especially point-based methods, the created surface appears to be very smooth. However, there is not much control about the smoothness of the surface. Via a parameter, one can combine several point samples into one, which makes it smoother, but, even on its lowest setting (used here), the result seem to be still less-detailed than the surfaces, produced with OM.

This test-case also shows that a watertight algorithm bears the danger of conveying the wrong impression to the viewer. Small holes might have been filled quite plausibly and the user will not be able to differentiate between artificially created and true, measured data, unless it is somehow encoded into the 3D model, for example via a quality flag assigned to each vertex. For the time-being, the only way to obtain a vague idea of the reliability of a triangle, is to measure the length of its edges. The current Poisson implementation uses an adaptive resolution approach and coarsens the resolution of the mesh in less-densely sampled areas. Hence, the longer the edge is, compared to the average resolution of the current mesh, the more likely it is that the

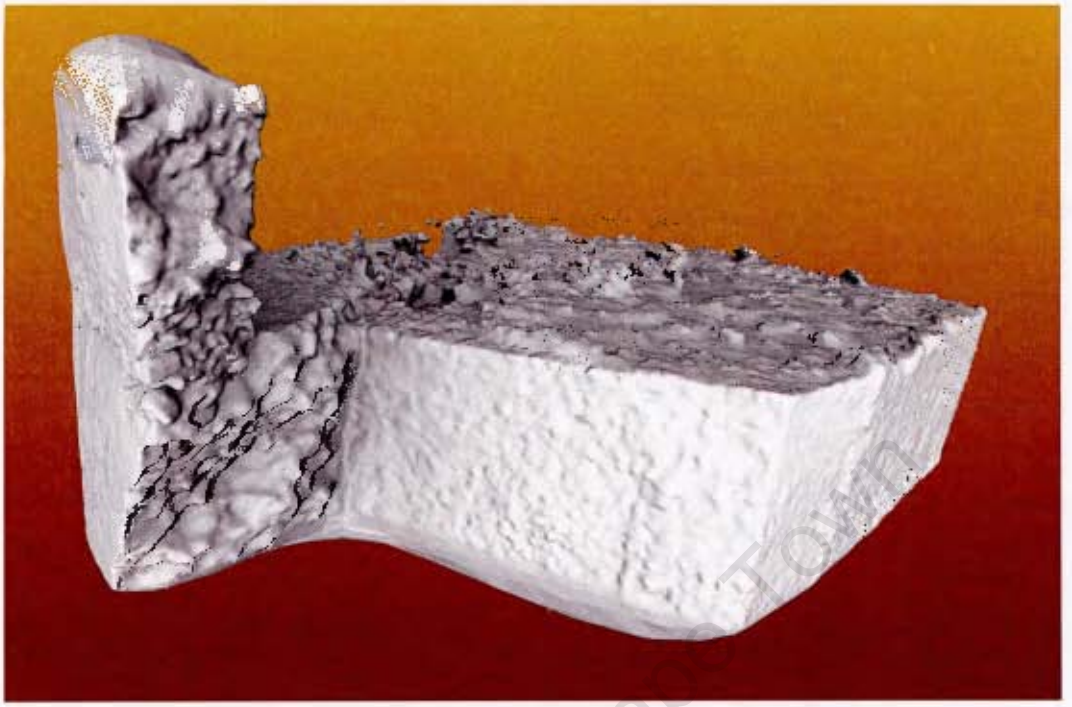

Figure 3.22: Result Poisson, Octree Level 10 which equals a cell size of about 10mm



surface was artificially introduced. If the length is about average, it can be assumed that the triangle is based on measured points, or at least, close to some (see fig. 3.23).

The larger the hole, the more likely is an entirely incorrect filling. As can be seen in figure 3.24a, where one of the larger holes was not filled correctly, even though there were some points available in the centre of the hole. This might not necessarily be a shortcoming of the Poisson approach in general. It could also be just misbehaviour of the current implementation.

As larger structures are always build on ground, there is at least one side of it, which cannot be scanned: underneath. Due to its watertight nature, the Poisson approach will close off these "holes" as well, producing excess surface which needs to be cut away (fig. 3.24b). These patches are characterized by its large triangles. Thus, one could select them based on the length of their edges, but this usually also would affect other fill-in patches and re-open holes. Alternatively, one needs to cut these triangles manually in editing software. As the used software tool cannot output the result in sub-blocks, editing of very large meshes can become very difficult.

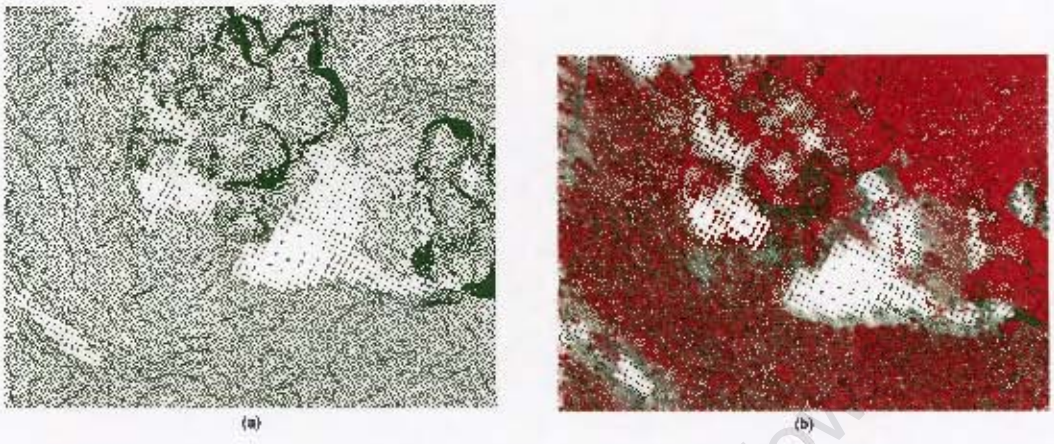

Figure 3.23: (a) Sampling density of the Poisson-model (green), which can give an impression about the reliability of a triangle. (b) shows the original dataset in red and the Poisson-model-vertices in green. The sampling density of the Poisson model is decreasing in areas which are not supported by any measured data. This method is not accurate, but serves as an indication in the absence of other quality measurements

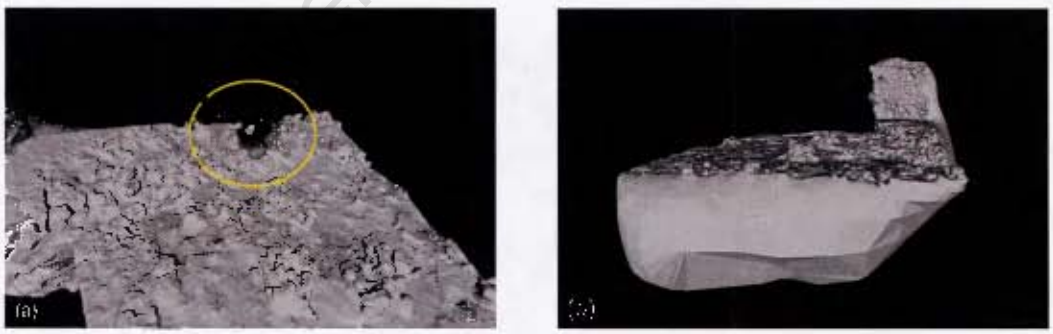

Figure 3.24: (a) Filling Holes with Poisson: Small Holes are plausibly filled, but some larger ones have not, even though there were some points available in the centre of the hole, which were reconstructed here as small floating islands. (b) Excess surface created by Poisson

#### 3.4.3.5 Volumetric Integration

Volumetric Integration relies on triangulated scans, which are then "merged" together, essentially by interpolating between overlapping areas. As explained in section 2.3.2.3, a cube containing all scans is created and divided into small cells, called the voxels. Each voxel corner is fitted with a signed distance function. The marching cubes algorithm, is then extracting the iso-surface by evaluating these distance functions, and converts it to a triangle-mesh. Hence, the size of a voxel determines the resolution of the output mesh. Details smaller than a voxel will not be reconstructed. The resulting triangles are basically averages of the overlapping surfaces within one voxel.

One of the advantages of this method is its scalability and thus it is applicable even for large datasets. Meshlab 1.3.0, an open-source editor and viewer for meshes comes with an implementation of this methods, called the "VCG Surface Reconstruction" or -when compiled as a command line version- PlyMC. Based on tests by the author, it is assumed that Polyworks ImMerge works on similar principles, but so far was not officially confirmed by Innovmetric. In the following review of this algorithm, it is focused onto the experiences with PlyMC, as it is open source.

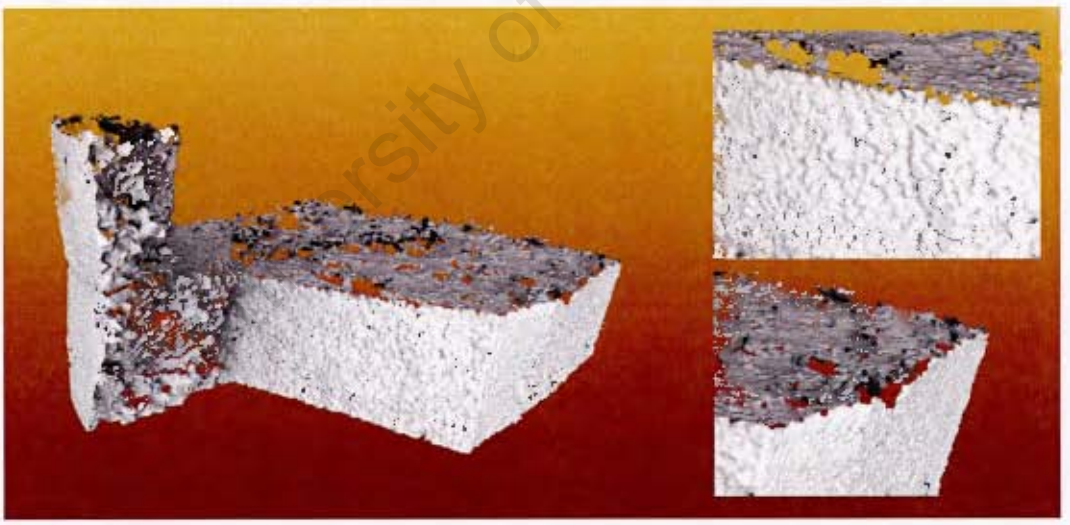

Figure 3.25: Result of PlyMC, voxel size 1cm, geodesic quality activated. The quality of each triangle is determined by its distance from the nearest boundary.

Experimental Results The output of PlyMC looks very detailed and complete, but a little too smooth in areas around holes (fig. 3.25). This is due to an internal smoothing function, which affects triangles based on their quality. If no quality values are provided, a geodesic quality is assigned, classifying triangles on their geodesic

distance from a boundary. If this filter is turned off and the input surfaces are assigned with quality values, for example, based on the distance and angle to the instrument, the surface appears much more detailed (fig. 3.26). Ptx2Ply was thus modified to estimate these quality values during the file conversion process.

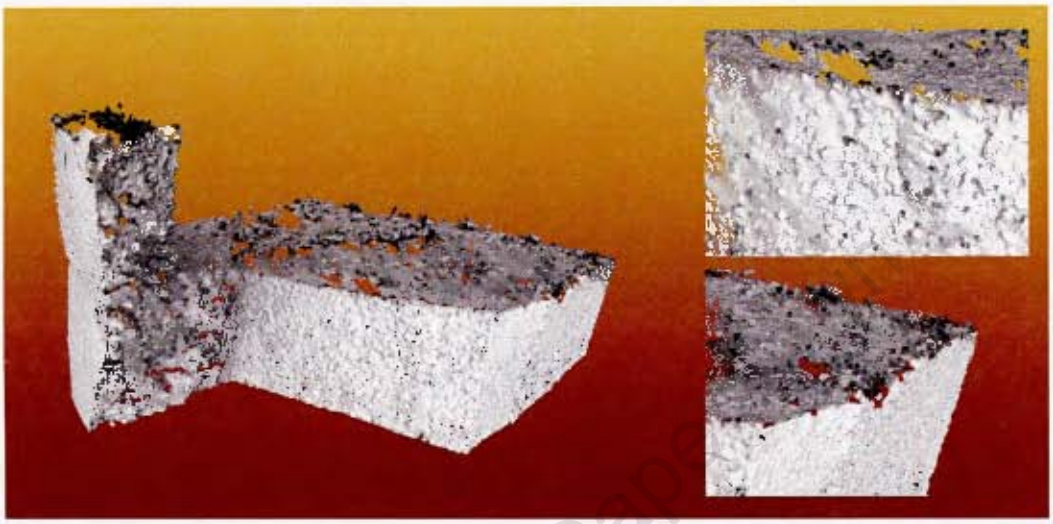

Figure 3.26: Result of PlyMC, voxel size 1cm, geodesic quality turned off. The quality of each triangle is determined depending on the distance and angle to the instrument

Vegetation is problematic for PlyMC, but the problem already starts before the actual merging. As leaves or grass blades cannot, or rather, will not be scanned in adequately high resolution, the pre-triangulation will connect several of them together or only produce small, isolated pieces. PlyMC is then attempting to close these holes by volumetric diffusion, extrapolating the surface along the boundaries. If the hole was not closed after several extrapolation steps (in here only one step was performed), the extra surface will not be removed again and thus even more complex boundaries are created.

Smoothing the scans before the merging could be a workaround for this problem, but also detail would be lost. In non-overlapping areas, it is however questionable if the detail reconstructed is actually based on measurements or just noise. If noise is the reason, smoothing could be justified. But in overlapping areas, noise (and also misalignment errors) would be smoothed automatically, as PlyMC is averaging between several surfaces. In this case additional smoothing would decrease the level of detail even further.

If the sampling density of different scans is more or less the same, the Volumetric Integration can produce very high-resolution, detailed surfaces (fig. 3.27a). If two

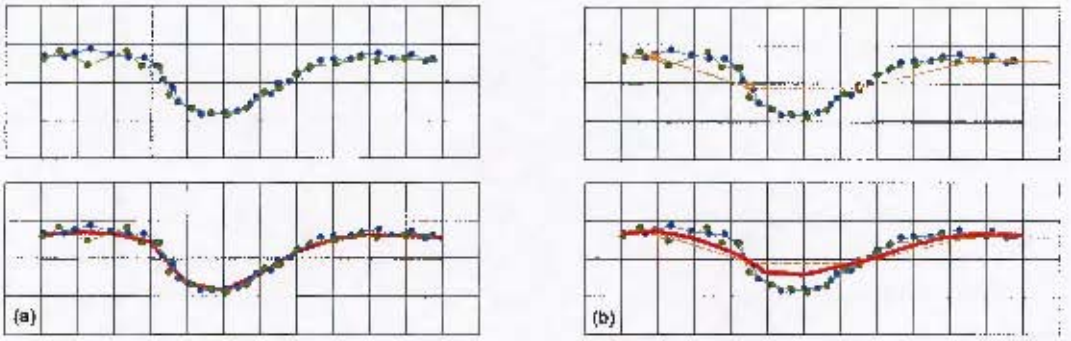


Figure 3.27: (a) Surface extraction if the sampling density is more or less the same on all scans. Final surface in red. (b) If the sampling density varies largely between scans, the resulting surface might lose detail.

scans overlap, and one of them is sampled less densely than the voxel size or than the other scan, it will result in a smoothed surface and details are lost (see fig. 3.27b). The features recovered with PlyMC are thus depending on the scan with the lowest quality in the overlapping area. This is mostly not a problem for close-range scanning applications ( $< 10m$ distance, e.g. relief, archaeological artefacts, statues), for which PlyMC was developed. Unlike medium distance scans of large structures, the variation in the distance of a close-range scanner to the object is nearly constant for all scans. In medium range scans however, especially spherical, 360 degree scans, the variation can be quite large, ranging for example from 10 cm up to 80m in one scan of a Leica HDS6100 (fig. 3.28). As the resolution of the scanner does not adapt with distance, it is impossible to keep the resolution constant on all surfaces. Hence, the only way to retain as much detail as possible is to lower the tessellation depth threshold, which results in more holes.

#### 3.4.3.6 Zippering

The Zippering algorithm removes overlap between different scans and re-triangulates in between them. The result is a single surface. The decision of which scan is to be prioritized is either up to the user or automatically made by evaluating the angle and distance to the scanner. It is not suitable as a reconstruction algorithm, because a lot of the information in the overlap is rejected even though it could be used to reduce noise, as done with the Volumetric Integration approach. Besides that, registration errors can lead to visible seams between the scans: As it just re-triangulates the gap in between two surfaces, a large gap will result in a harsh transition on the merged surface. It is assumed that the merging tool in Polyworks ImEdit is a basic implementation of

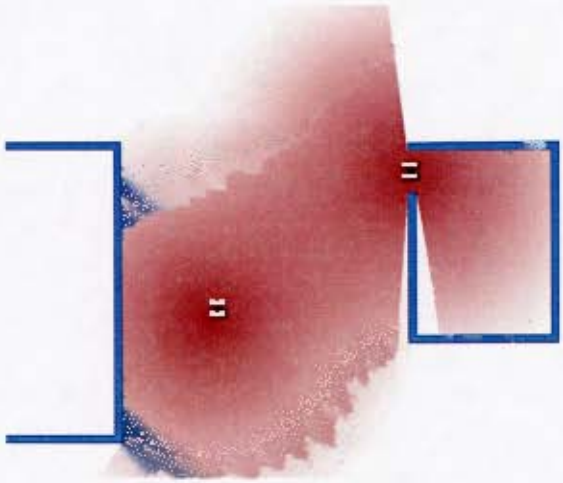

Figure 3.28: Distribution of points in a small scanning campaign with two laser-scanners, which are able to scan full-dome. Blue colour indicates a wall. The darker the red colour, the denser the point cloud is in that area.

this method: only two scans at a time can be merged together. The effects of the tool are similar to the "Zipper" by Turk and Levoy (1994) but no official information was found. The original "Zipper" source code is available to the public, but so far, there is no version for Windows platforms. The new releases of Meshlab (>1.2.3) contains a new implementation based on the original with some additions (Marras et al., 2010) and also only merges two surfaces at a time.

Theoretically this approach should be robust, but the tested implementation in Meshlab often crashed. The merging tool in ImEdit merges the surfaces well, but since the result contains both surfaces, the model becomes quickly too large to remain loaded in memory. A solution where the result is automatically split into sub-blocks is lacking. For these various reasons, it was not included in the comparison, but it is mentioned here for the next section of combining the results of different meshing approaches.

#### 3.4.3.7 Discussion of Results

**Influence of Tessellation Depth Threshold** As mentioned above, the tessellation depth threshold, if set too loosely, can be responsible for artefacts or less accurate point normals. But the more restrictive the threshold is chosen, the more holes will appear. Due to different characteristics of each site, for example, the amount of vegetation or the height of buildings, no recommendation for a specific value can be given.

Setting the threshold for the test-case to an overly extreme value of 30 cm, reveals

clearly the advantages of Point-Set surfaces. PlyMC retains the artefacts from the single, pre-triangulated scans and merges them into the final model (fig. 3.29a). In areas, where the distance between these surfaces is larger than the voxel size, PlyMC will create double-surfaces (fig. 3.30). Hence, if PlyMC is used, it is very important to set the tessellation threshold for the pre-triangulation of the individual scans not too far off the voxel size, which will increase the likelihood of holes in the model.

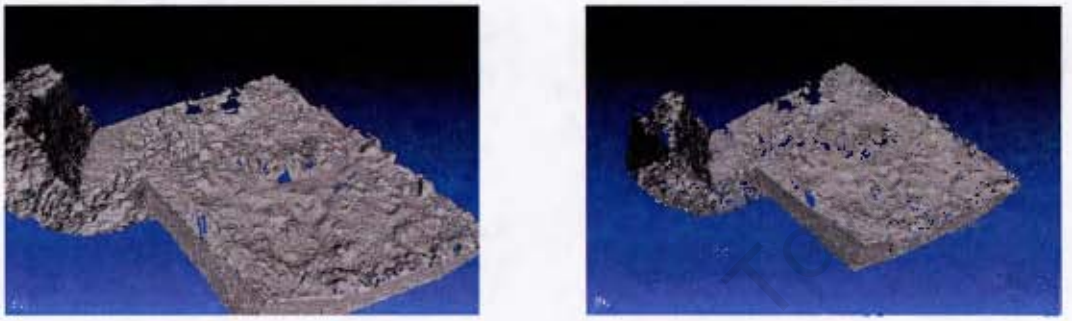

Figure 3.29: (a) PlyMC result, voxel size 1cm, tessellation threshold for input data 30cm. (b) OM result, 14mm, tessellation threshold for input data 30cm

Point-based algorithms however, do not seem to be negatively affected by less accurate point normals, as demonstrated on a surface extracted with OM (fig. 3.29b). As expected, these extra vertices actually enable OM to create a more complete surface.

Figure 3.30: Double surfaces, which appear with PlyMC, if the tessellation threshold is set too large.

| | Geomagic | OM | PlyMC | Poisson |
|---|---|---|---|---|
| Handling of large datasets | - | ± | ± | o* |
| Reconstruction of Detail | o | − | ± | o |
| Software stability when processing noisy datasets | ± | - | + | - |
| Output quality of noisy datasets | - | + | | + |
| Output quality with respect to TDP | + | o | | o |
| Watertight output surfaces** | - | - | - | + |

(−) not good, (o) acceptable, (+) good

Table 3.1: Comparison of different surface reconstruction methods.
*Poisson can handle large datasets, but produces a single, large model instead of splitting the model into small, manageable blocks.
**Watertightness is a useful feature, but not essential to this comparison

### 3.4.4 Surface Reconstruction Workflow

#### 3.4.4.1 Surface Reconstruction Algorithm

Even though these are today's most prominent reconstruction algorithms, none of them seems to be an all-round tool to process all kinds of input data. The Delaunay shows problems with dense, noisy datasets, but produces a maximum of surface, in terms of size. PlyMC is very robust and scalable, but does not use all data because of the tessellation depth problem. OM is not very stable and also relies on pre-triangulated scans, but produces nice details without too many holes even in vegetation-areas. Poisson creates perfectly watertight surfaces, which are too smooth, compared to other algorithms.

Vegetation, or strong noise, seems to be problematic for all tools. It causes vertex normals to flip (Delaunay), the creation of very irregular and complex boundaries and holes (PlyMC), or program crashes (OM, Poisson). It is thus necessary to clean out as much as possible in a preceding cleaning step. This, however is not a trivial task and, at present, involves a lot of manual work. Hence, it is recommended to rather work with more robustly programmed tools, such as Geomagic and PlyMC.

Despite all the stability issues with the software and just regarding the quality of the produced surface, it seems that OM, and respectively MLS-approaches would suit best as the meshing approach for terrestrial, medium range laser-scanning. The point-based approach produces reasonably complete surfaces while recovering fine detail. Each additional point will contribute to enhancing the detail, as opposed to PlyMC,

where the maximum level-of-detail is determined by the lowest quality scan in an overlap area. To receive maximum benefit out of MLS-approaches, the process of obtaining pre-determined normals, and thus the influence of the TDP, needs to be addressed and improved. To make use of all input points, exact normals need to be extracted from the entire point cloud, not per scan.

However, as shown above, a global solution for point normals estimation is not easily achievable, because the result will be influenced by noise (see section 3.4.2.2). Thus, a two step approach is suggested:

1. Calculating approximate normals for all vertices in a scan, even if the result would not be reliable. Each recorded point implies it was in a clear line of sight to the scanner. Thus, the true point normal will face more or less towards it, but not away. This will serve as an indicator of the orientation.

2. Merging all points into a single point cloud and only re-evaluate normals of poor quality.

However, to the author's knowledge, a software solution implementing the above is not available at present.

#### 3.4.4.2 Combining Different Meshing Solutions

As shown above, MLS-approaches are believed to work best on terrestrial laser-scan data. However, OM is not too reliable and cannot be recommended for the inclusion into this workflow. As all of the other algorithms have their pros and cons, and no clear recommendation for either of them can be given at this time, it is suggested to create two surfaces and merge them, or parts of them, similar to Callieri et al. (2009): one to recover fine details, which will most likely also lead to more holes, and a less detailed but more complete "fill-in" surface (fig. 3.31).

PlyMC is very scalable and produces very detailed surfaces, and will thus serve as the main surface reconstruction algorithm. Poisson and Delaunay produce the most complete models, however, Poisson also introduces new surface, not based on any measurements, which is not acceptable for cultural heritage applications. Hence, wherever possible, a Delaunay method should be preferred to produce the fill-in surface.

To combine models produced with PlyMC and Delaunay, two options are available: either the Zippering method, represented here by Polyworks ImEdit, or PlyMC itself. The zippering appears to be a well-suited solution as it will only stitch both models together along boundaries, closing holes without smoothing detail. But applying this

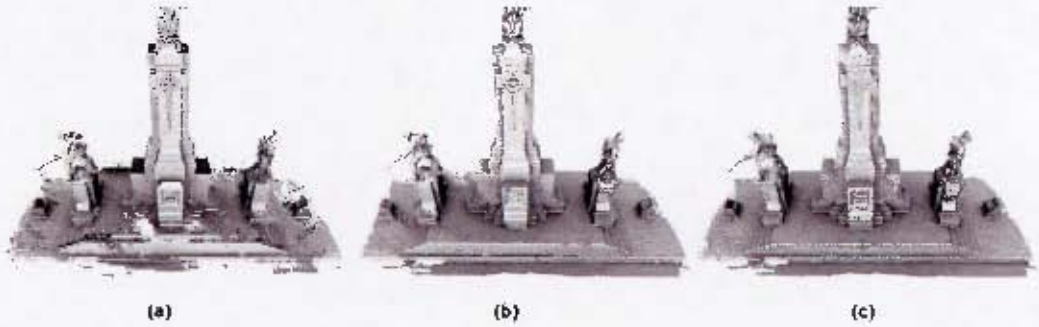


Figure 3.31: (a) PlyMC model. (b) Geomagic model (c) Merged version of the Geomagic- and PlyMC-model. The combined surface carries the detail of (a) but is also as complete as (b)

method to large datasets might become difficult, as no software is available which could handle arbitrarily large datasets.

PlyMC, on the other hand, can easily handle large datasets and, because of the command line nature, is also easy to script. The "fill-in" surface could be ingested by the normal merging process and treated as a normal scan. This way the merging happens as part of the surface reconstruction process. The user can set a scan-specific weight in PlyMC for each input scan, to soften the influence of a particular scan, or in this case, the "fill-in" surface. If the "fill-in" surface differs more than the voxel size/output resolution from the "fine-detail" mesh, for example due to different resolutions or different noise levels, the output mesh can contain double surfaces. In addition, both surfaces need to be oriented the same way. Incorrect orientation, as often occurring with Delaunay algorithms, would result in cancellation of two opposing triangles during merging.

#### 3.4.4.3 Large Datasets

With advances in laser-scanning technology, the amount of data acquired per scan increases permanently. Most available algorithms, even if they are designed as out-of-core tools, do reach a limit at some point in time or become inefficient. One solution is to sub-sample the input scans, either by ignoring scan lines and rows, or by re-sampling the scans in regular intervals (grid-sampling), which often leaves visible grid-patterns on the final model. However, it is always preferable to reconstruct a surface based on the highest available scan-resolution, as any simplification reduces detail.

Theoretically, any surface reconstruction algorithm can be used to create large models, if the dataset is split into small enough pieces, before processing. Special

attention, however, needs to be paid to the points and triangles along the boundary of each sub-block, as no triangles will be created in between two parts. One solution is to slightly extend each block to create some overlap. But unifying all blocks later into one large file without further processing will keep the boundaries of each sub-block intact and often create intersecting triangles. This will complicate the hole-filling and simplification process and can cause rendering artefacts (fig. 3.32).

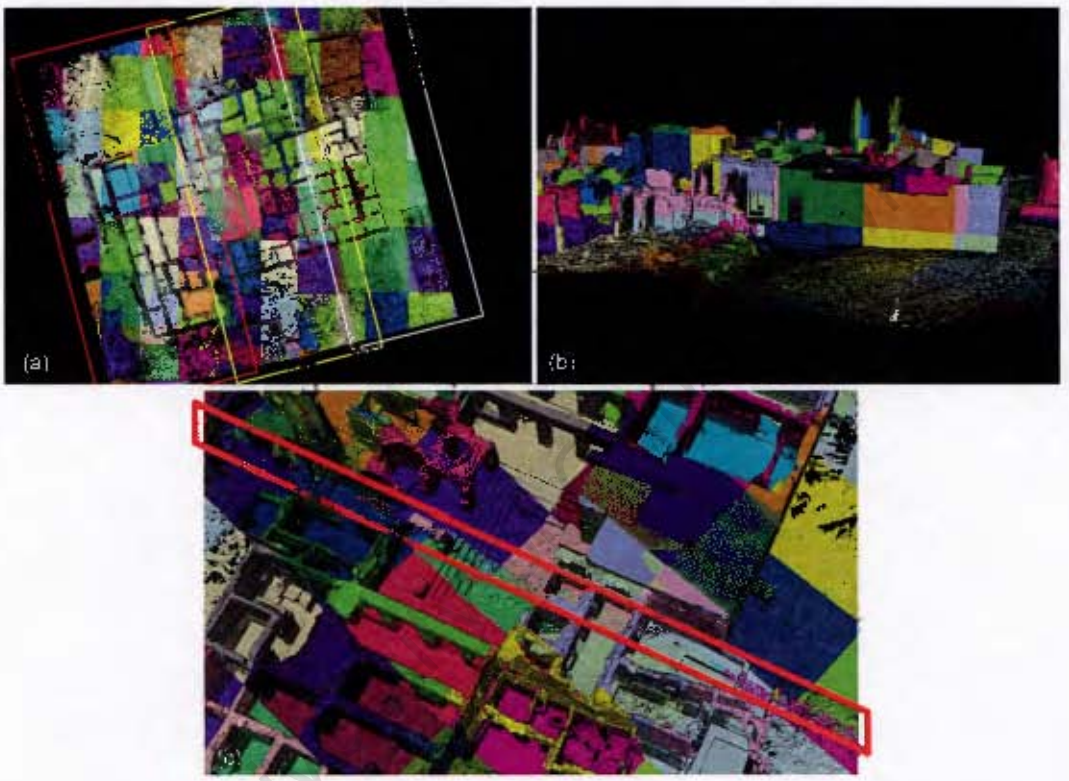


Figure 3.32: (a) Dataset, which was subdivided into three larger parts. (b) Each of these parts was processed with PlyMC, which further subdivided the datasets into smaller portions. This allowed parallel processing of the three larger areas. The smaller blocks match perfectly along their boundaries, while the three larger areas are overlapping. If the overlap (c) is not removed, it can cause rendering artefacts as well as problems during simplification, hole filling.

To cope with extremely large datasets, PlyMC (32-bit software) was designed to subdivide the dataset into pieces, with perfectly matching boundaries in between the sub-blocks. However, it slows down, the more subdivisions become necessary. This is because PlyMC reads entire scans, even though just a very small part of them might be required for the processing of the current sub-block (fig. 3.33a). This will become more and more time-consuming the higher the resolution of the scans and thus the

larger their file size is. The disk access will thus be more efficient by splitting the data into sub-volumes prior to the actual merging with PlyMC (fig. 3.33b).

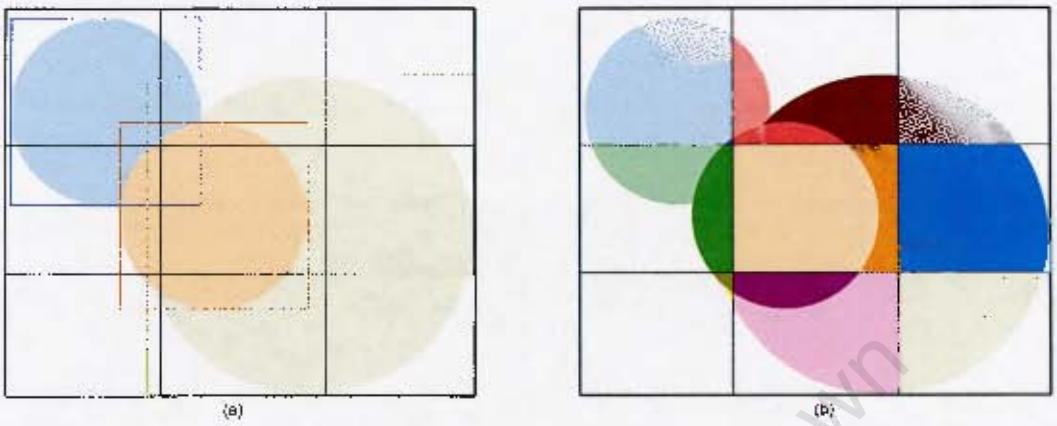


Figure 3.33: Illustration how the hard-disk access of PlyMC can be optimized by subdivision. (a) The bounding box of the entire model is subdivided (black grid lines) during processing with PlyMC. Even though the orange scan only marginally falls into the top left sub-block, the entire scan needs to be loaded. The green scan actually has no data in that block at all, but its bounding box intersects with the boundaries of the sub-block and also needs to be loaded. (b) Disk access can be optimized by subdividing the input scans with ptx2ply before they are processed with PlyMC.

For this reason, the PTX converter Ptx2ply was modified to subdivide the scans into blocks, prior to the merging with PlyMC. To use this feature, only a bounding-volume (limit-box), which encloses the structure, needs to be specified, as well as the amount of subdivisions, the volume should be split into. In addition, ptx2ply allows adding an overlap in between the blocks.

The sub-division feature proved to be a very powerful tool within the Zamani-project, as it allows the processing of the data with any available algorithm in the highest resolution. Figure 3.34 shows the increase in quality, when the input data set is of maximum resolution. It is a comparison of two models created with PlyMC, one based on data which was sub-sampled by only using every third line and row of a scan, while the other one is based on scans in highest resolution. Both models were extracted at 2cm-resolution. The dataset in highest resolution consumes 29.4GB of disk space, while the simplified dataset is only 3.1GB large.

Another advantage of subdivision is the ability to process several blocks in parallel, either on a multi-core system and/or on several machines. Many meshing tools, especially research tools are not optimized for parallel processing. Hence, the subdivision of the input dataset serves as a practical alternative.

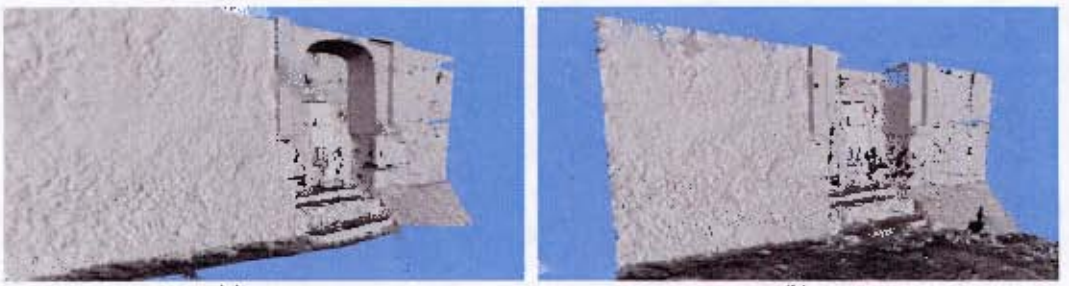


Figure 3.34: Two models extracted with PlyMC at 2cm resolution. (a) the input dataset was sub-sampled. (b) the input dataset was not reduced.

After the surface reconstruction with PlyMC, special attention needs to be paid to the overlap between the ptx2ply-blocks, as mentioned above. To remove it and produce matching boundaries between sub-blocks, it is suggested, to run PlyMC again to remove the overlap in between. This however, can result in a slight loss in quality due to re-sampling. The additional merging process with PlyMC can also be set to split the final model again into small, manageable pieces. The vertices on a boundary of a block will then perfectly match with its neighbouring blocks and, when merged later into one file, can easily be connected by removing duplicated vertices.

If subdivision is employed, pre-alignment of the entire dataset is advantageous, so that the axes of the bounding box are parallel to the system axes. This avoids unnecessary processing of "empty" blocks and simplifies the access later.

Some reconstruction methods, such as Poisson, are designed as an "at-once" process, as they try to create the whole model in one go. Especially the watertight feature can become problematic when sub-dividing, as shown in figure 3.24. If a hole is extending over two sub-blocks, the Poisson tool can only close part of it. This however will most likely result in non-matching overlap and merging will become complicated. The only alternative is to allow for a larger overlap area in between the blocks (20-30 percent), to be sure that all holes extending over the boundaries of neighbouring blocks are closed identically. After reconstruction of the individual blocks, one simply cuts away a large portion of the excess overlap and merges these smaller blocks. For this purpose another tool was created, plyCut. It allows the specification of a limit-box and an extension or shrinking factor.

#### 3.4.4.4 Retaining Colour Information

As recommended during the texturing step, the intensity values of the scans can be of significant value for the image alignment. However, most methods mentioned above do

reject colour information during the meshing, which is not a limitation of the method itself, but rather of the software implementation. Only Geomagic retains the colour values by standard.

Alternatively, PlyMC in its current version (4.0) offers a "splatting" technique (parameter -p). Enforcing this parameter will also create a smoother surface, than without this option, but at least the colour is retained. Thus, two models need to be produced, one with and one without the splatting option. Both models, high resolution- and colour-model, can be loaded into Meshlab where the colour can be transferred onto the more detailed mesh by using the "vertex attribute transfer" option.

Ptx2ply was extended to map the intensity values in PTX files to grey-scale information (parameter -c2). While the general range for intensity values in a PTX-file is [0...1], it is important to know, that the various instruments only use a subset of this range. For a similar grey-scale appearance the following ranges should be specified for ptx2ply:

- Leica Scanstation C10: 0.10-0.35
- Leica HDS 3000, Scanstation 1&2: 0.35-0.65
- Leica HDS 6000, 6100: 0.001 - 0.5

#### 3.4.4.5 Resulting Workflow - Surface Reconstruction

The most ideal workflow depends drastically on the available input data. Generally the most stable software is PlyMC, which also produces very good results. However, if there are many holes in the final model, due to vegetation or TDP problems, it is recommended to produce another model with Delaunay methods, here Geomagic, and merge the two with PlyMC. The Delaunay model can also be employed to transfer the intensity information to the final model with Meshlab, as described above.

To manage arbitrarily large datasets, it is found to be of great use to split the data first into overlapping sub-volumes and process them, with the preferred surface reconstruction algorithm, here PlyMC. The resulting overlapping blocks can be merged again with PlyMC. During this second merging step, the Delaunay model and some other hole-filling patches can be inserted into the PlyMC process to be merged with the main model.

Poisson can also be used as a surface reconstruction method, if watertight meshes are required. However, if the process is unsuccessful due to problems with vegetation, or if the result is overly smooth, it is recommended to use Poisson as a mesh-repair-tool, as described in the following section.

Please refer to figure 3.35 for an overview of the workflow of this section.

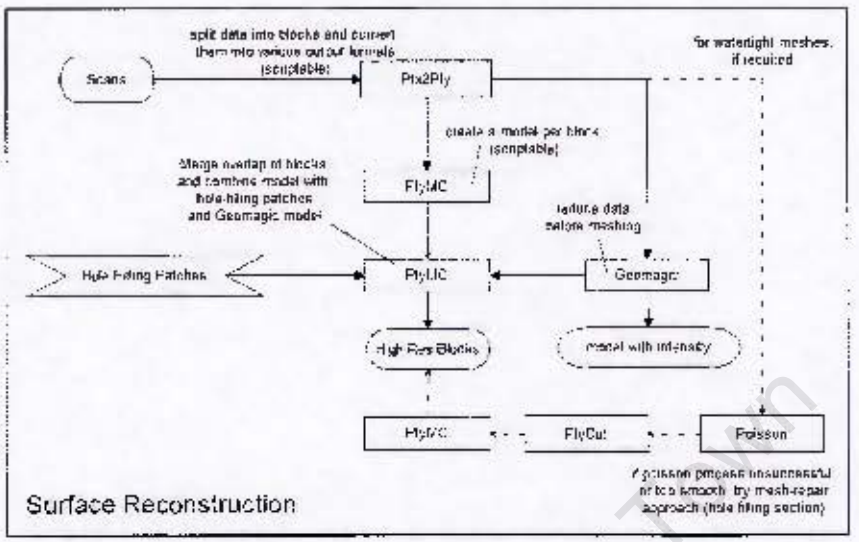


Figure 3.35: Diagram of Surface Reconstruction workflow.

## 3.5 Hole Filling

Based on the experience by visually comparing models, users will always prefer a watertight 3D model to a model with many holes, even if the latter is more detailed. Water-tightness is also a precondition for 3D printing and in virtual reality environments. For cultural heritage purposes, hole-filling is however a delicate matter, as conservation and restoration experts need to rely on recorded data to be sure their analysis and planning is correct. The introduction of non-recorded, artificial patches is thus non-desirable. The question is however, which patch size is regarded as critical? In principle, creating a triangulated model can always be regarded as some kind of hole filling, because we move from a discrete dataset to a continuous surface. However, there is no absolute answer to this problem, because the availability of such reliable data in high resolution is new to these targeted professions. So far, architects are still mostly working with CAD-Drawings, with only a minimum amount of points, lines and triangles. Believing that this might change in the future, the Zamani project thus adopted a policy for hole-filled models, to provide the un-edited, original version as well. Ultimately, this could be avoided by assigning a standardized quality tag to each vertex of the 3D model, which holds information about its reliability, which could be determined by its distance from an actual measured point. But so far, no such standard is available.

### 3.5.1 Automated Methods

Even though hole-filling is usually not desired for cultural heritage applications, it might become necessary to produce a watertight surface, for example for 3D printing. The most efficient approach to obtain a hole-free model is to employ a watertight algorithm for the surface reconstruction, such as Poisson.

However, Poisson often suffers from software crashes, due to vegetation and produces overly smooth models. To work around these issues, one could employ it to re-mesh the model, once created with a non-watertight algorithm. The model, constructed with other techniques, such as Volumetric Integration, has a maximum of surface detail and potential problematic point samples of vegetation were already reduced to a minimum (fig. 3.36a).

To retain the detail of the surface during re-meshing, it is suggested here, to refine the input mesh by subdividing every triangle, for example with the "Butterfly" or "Catmull-Clark" filter in Meshlab (fig. 3.36b,c). After the refinement, all triangles can be rejected as only the vertices and their normals are required by the Poisson software. The subdivision of triangles provides more samples for the Poisson-tool, and thus produces more constraints during the fitting of the functions to the point cloud. Using this approach, the detail of the input mesh can be transferred almost entirely to a watertight mesh (fig. 3.36d).

Other surface reconstruction tools, even though they are not designed to create watertight meshes, offer hole filling as well. PlyMC and OM for example, employ an approach, similar to Volumetric Diffusion (Davis et al., 2002). The surface along a boundary is extrapolated for -a user defined- amount of steps (fig. 3.37). While this can be useful to close very small holes, larger examples prove to be more difficult, especially, because the software does not revert the hole back to its previous state, if it was not closed by the extrapolation step. In areas with vegetation, this behaviour often creates cone-like structures, as shown in figure 3.38. It is thus recommended to set this parameter to the minimum (=1). This does not disable the extrapolation, but lower values will lead to a lot more holes, because of a principal limitation of the marching cube algorithms.

### 3.5.2 Semi Automated and Manual Methods

Alternatively to entirely automated approaches or to fix some more complex areas, one can also employ semi-automated tools, such as in ImEdit, which is part of Polyworks, or Geomagic Qualify. These tools allow the user to click on a hole which is then either filled in a "flat" or "smooth" mode. Smooth filling will extrapolate the surface along

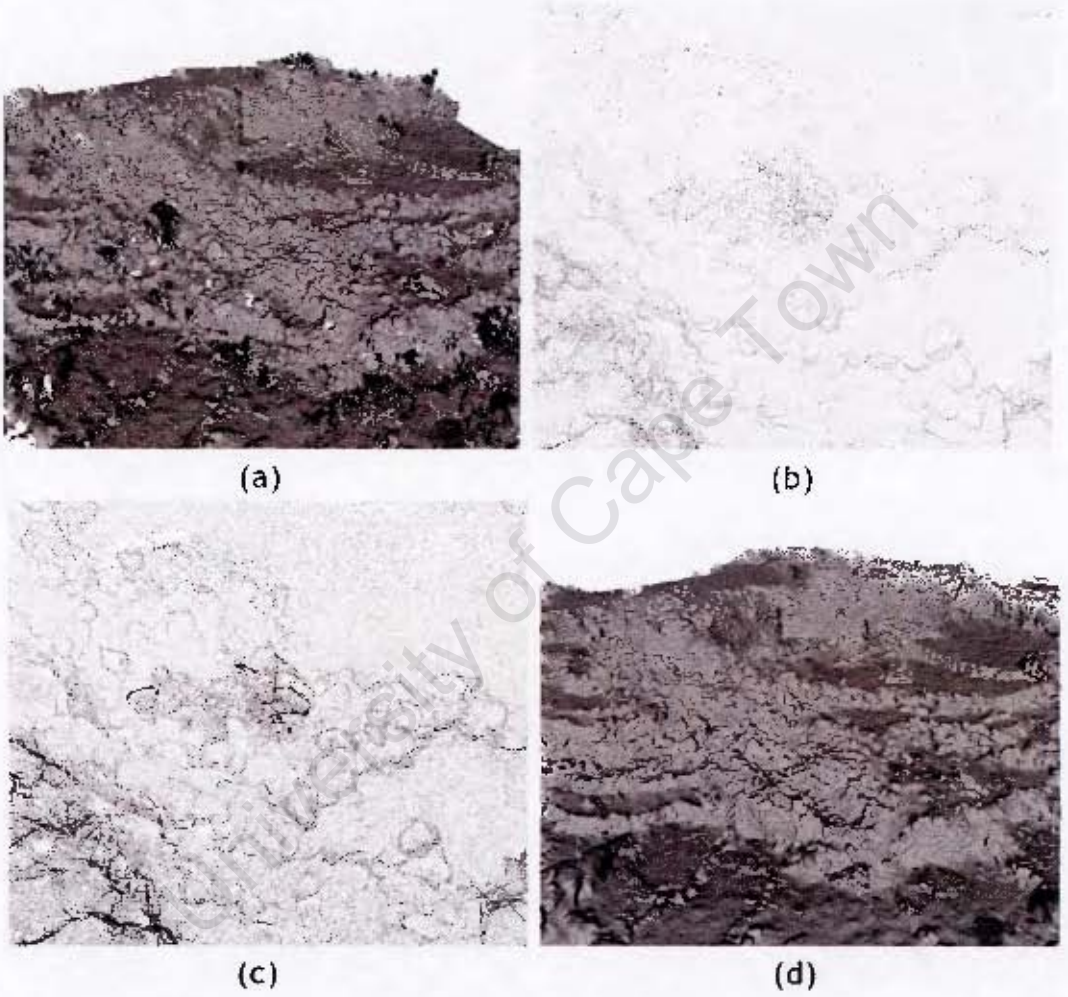




Figure 3.36: (a) Geomagic Model of the terraces of DGB II, Cameroon. (b) Vertices of Geomagic Model. (c) Refined vertices. (d) Re-processed model with Poisson, based on the refined vertices of image(c).

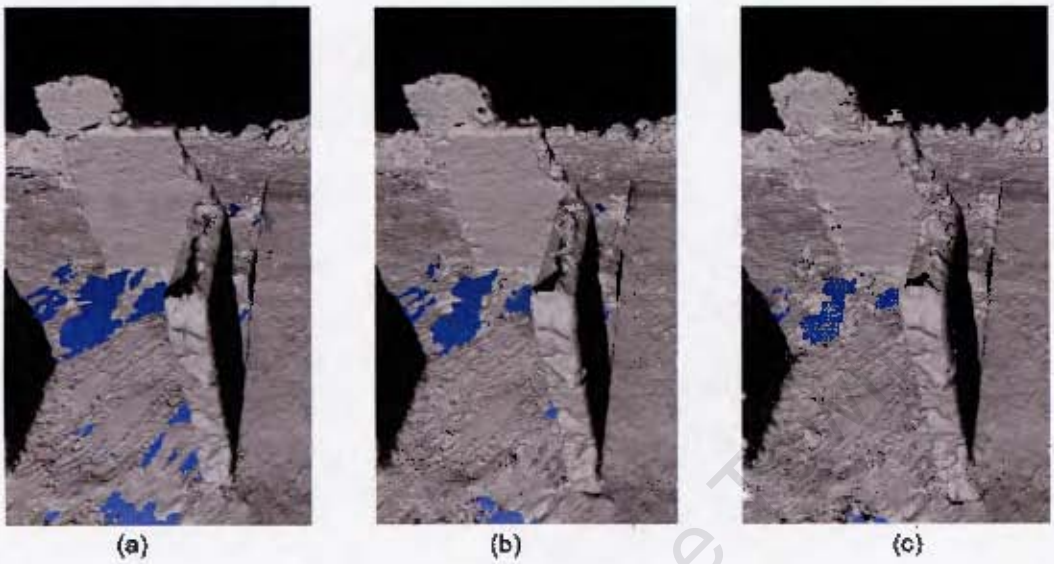


Figure 3.37: Volumetric Diffusion in several steps with PlyMC, (a) expansion level=1, (b) expansion level=2, (c) expansion level=3


Figure 3.38: Artefacts caused by Volumetric Diffusion. Hole before (a) and after (b) the process.

the boundary, while flat filling just performs a simple re-triangulation in between the vertices on the boundary. Both approaches can be used to fill smaller and simpler holes, but the larger and more complex the holes become, the less reasonable the result will be. For these large holes, ImEdit offers the possibility to define a surface patch, which the user can sculpt by positioning anchors. If the anchors are close to existing surface, they will snap onto it to ensure smooth continuation. Once the patch looks satisfying, the hole will be triangulated according to the sculpted patch (see fig. 3.39).

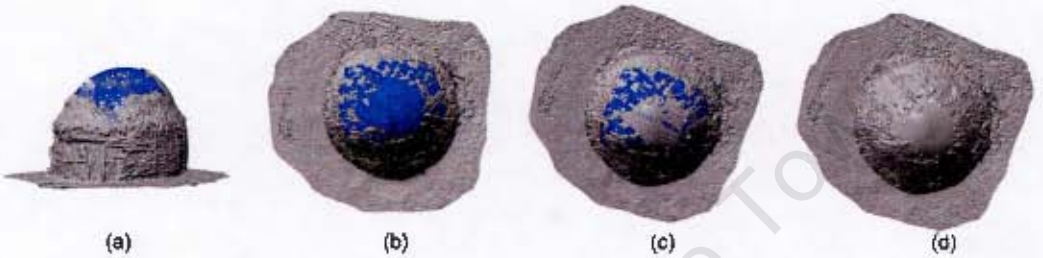


Figure 3.39: Sculpting tools of ImEdit for hole filling. (a+b) A blue patch is fit to the surface around the hole. (c) The hole is filled by using the blue surface as a guide. (d) Filled model



### 3.5.3 Including Other Sources of Data

Photogrammetry, if available, is a very valuable source of input. Especially with structure from motion tools, surface is quickly reconstructed, which can be scaled and registered to laser scan models to fill holes. Agisoft's PhotoScan Professional, Autodesk 123D Catch, My3DScanner and also Bundler in connection with PMVS, can be employed to automatically reconstruct patches, even from uncalibrated images. Using un-calibrated images might create less accurate surface reconstructions. However, these will still be more accurate than entirely fabricated patches.

Once the surface is created, control points have to be defined on laser-scan and SfM model to find the correct scale, translation and rotation. The latter two can be refined with an ICP-process in registration tools, such as Meshlab and Scanalyze, as described in the following section. If the patch is correctly positioned, it can either be merged with a "Zippering"-method or treated as a laser scan during the surface reconstruction process with PlyMC, as described above.

### 3.5.4 Resulting Workflow - Hole Filling

As mentioned before, hole filling is not essential, and actually questionable for the documentation of heritage sites. While it might be acceptable to fill very small holes during surface reconstruction by extrapolation of the surrounding surface, the only accurate filling for large holes is the creation of patches with photogrammetry/SfM. If that is not an option, it is recommended to sculpt patches with Polyworks, because of its large toolset, but a second model should be delivered then.

However, if the aim is a 3D print of the structure, a watertight model is required. The quickest method to achieve this, it to re-mesh a previously constructed mesh with Poisson, as described above.

Please see figure 3.40 for an overview of the workflow of this section.

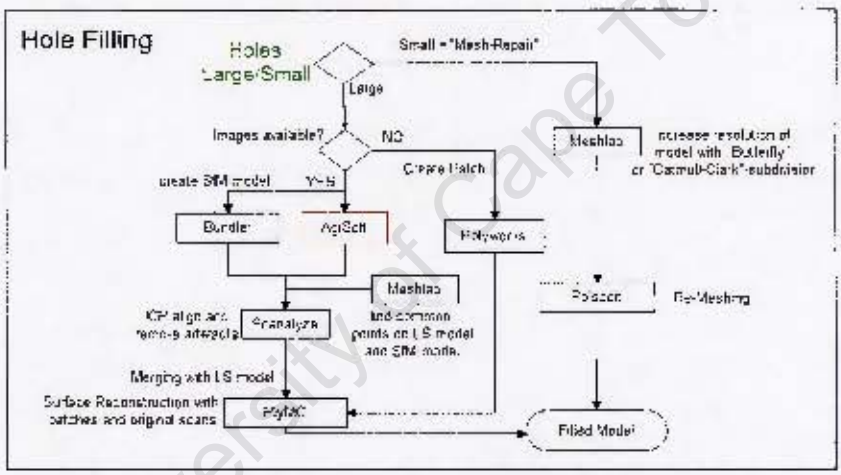


Figure 3.40: Diagram of hole filling / mesh repair workflow.

## 3.6 Registration

In section 2.2.2.1, it was pointed out, that target registration is too time-consuming and difficult to be used within the Zamani project. In the following, it is thus focused on the surface matching method, which relies on matching geometric features in the overlap area of two scans. As mentioned earlier, the standard solution for this problem, ICP, is looking for closest points in two scans and calculates a transformation which minimizes the distance between all correspondences in the least-squares sense. This process is iterated until convergence or until a user-defined threshold of iterations was reached.

### 3.6.1 Surface Matching with ICP

The ICP is controlled by several parameters. The most important one, common to all variants, is the search distance for common points. If a point on scan A has a closest point on scan B, which is as close, or closer than the search distance threshold, the points are considered as valid samples for minimization. The user can usually also define how many of these samples may be collected and how many iterations should be performed. At the end of an iteration, the ICP will provide, depending on the software, an error value (maximum or average), or an error distribution statistic. This result needs to be treated with care. A small value does not necessarily mean a good fit and does not give an absolute error between both surfaces. A low maximum error only expresses, that all or most distances between valid sample points within the search distance were minimized to below the maximum error. Corresponding points above the search distance will not be included in this statistic. It is thus recommended to also visually inspect the result, and to check whether both surfaces interweave evenly (fig. 3.41).

Figure 3.41: ICP-Alignment: Scans are only well-aligned, if their surfaces interweave evenly (b).

To find the best-fit transformation, the ICP minimizes the errors between points with least squares. As described in the previous chapter, mostly two error-metrics are used: point-to-point and point-to-plane. Point-to-point is generally regarded more resistant to noise, while point-to-plane converges faster and seems to converge less likely only to local minima. Thus, if the option is available, one should prefer the latter for a "rough" registration and point-to-point for a last "fine" alignment.

As can be seen from the procedure, the search of the ICP for corresponding points is simply driven by distance between the two surfaces and thus it is very likely that wrong pairings are made. The ICP, as it is not guaranteed to find the global minimum, thus requires the scans to be already roughly aligned. This is usually a manual process, requiring the user to identify three or more points on each of the two targeted surfaces.

Alternatively, the user can shift and rotate one surface until it is in an approximate position.

After an initial alignment was found, the ICP can do its work. A manual user alignment is considered a "rough" position and many of the closest points on both surfaces are still wrong pairings. Hence, care needs to be taken, while setting the search distance threshold, which should be large enough, to find enough correct pairings. Setting the search distance of the ICP straight to the desired final matching tolerance error can cause the ICP to converge to a local minima or even cause the surface to drift away from its correct position. Good experience was made with the following routine: starting with only few point samples for faster iterations and a very high search distance (e.g. 20cm), several iterations should be performed until one cannot see any more significant movements. The error should have dropped significantly during one of the previous iterations, and the surface should now be, more or less, in the correct position. The search distance is then reduced in additional iterations, e.g. 10cm, 5cm, 2cm, 1cm, "tightening" the two surface more and more together (fig. 3.42). When the last iteration is performed, the amount of samples should be increased to what memory and time allows. In the end, one should also consider to switch to Point-to-point mode, to obtain the best fit of as many samples as possible. The search distance should not be set to below the accuracy level of the instrument. Usually 1cm search distance is providing good results for the final ICP iteration.

Registration with ICP in its original approach is a pair-wise process; only two scans at a time are aligned to each other. The software Scanalyze allows to group two aligned scans together and register a third scan to this collection. Noise in all scans belonging to a group will "thicken" the, theoretically infinitely thin, surface. As it is computational expensive to extract the correct averaged surface after the grouping, and thus not possible to calculate it on-the-fly, Scanalyze will register the new scan to the top of the crust, formed by all scans within the group. Thus, after grouping all scans together, the scan which was added the last, will not perfectly match the very first. Hence the amount of scans within a group should be kept as small as possible.

### 3.6.2 Global Registration

To reduce the accumulated error during pairwise alignment, there are software solutions available, which perform a global registration, either distributing the error between all scans or even performing a simultaneous multi-view alignment. While this might sound good, tests within this research with Polyworks ImAlign or Meshalign (VCG-lab, I.S.T.I, Pisa) showed a disillusioning and disturbing clustering behaviour, at least

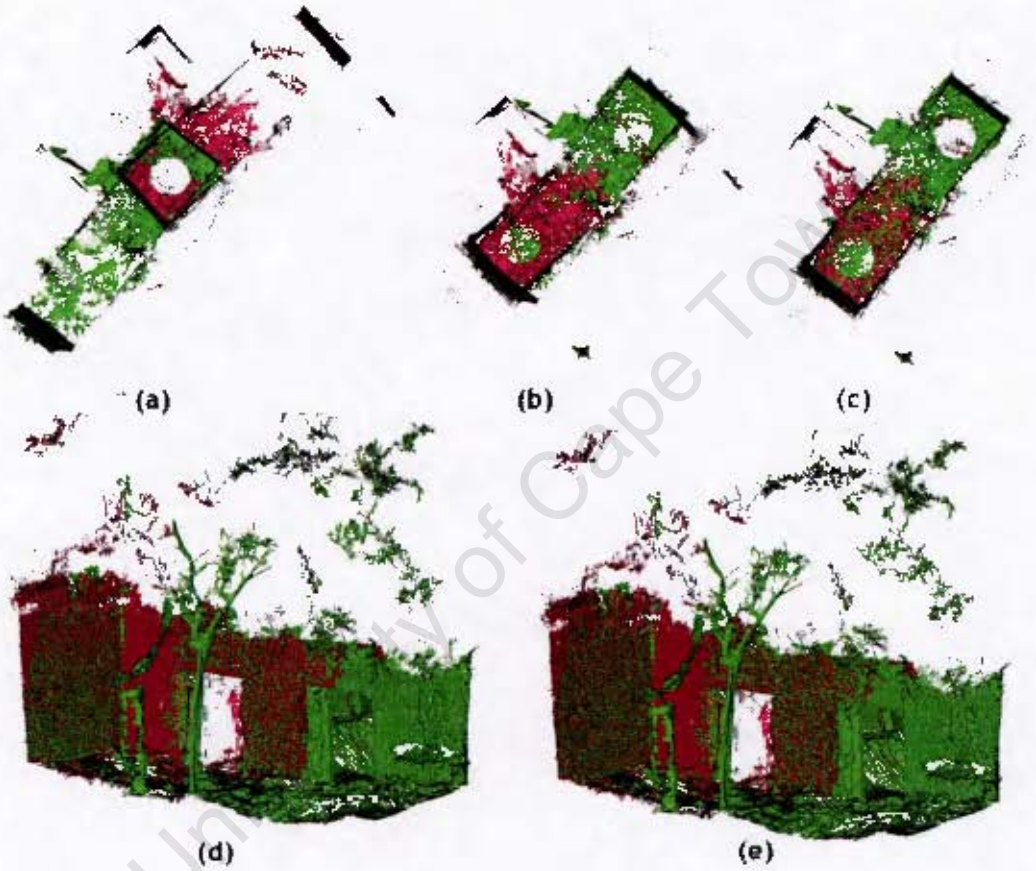




Figure 3.42: Alignment process with ICP: (a) two scans in their local coordinate systems. (b) The red scan is manually, approximately aligned by shifting and rotating the scan. (c and d) ICP was run on the initial alignment with a search distance of 20cm. (e) After decreasing the search distance in small steps to 1cm during several iterations of ICP, the two scans are well aligned.

true for terrestrial, medium range laser scans:

The purpose of the ICP is to minimize the distances between the majority of corresponding point samples on both surfaces, indifferent of their distribution across the scan. As mentioned before, the sampling density on all objects within a 360° scan cannot be kept constant, which is why objects far away from the scan centre will be in a much lower resolution and contributing less samples to the ICP process than closer objects. If we now imagine a chain of four scans, where each scan is only linking to two other scans, but one of these links is only based on a very small overlap of surfaces in a very low resolution, the strong constraints between the other links will outweigh the constraints of this weak link. Primarily the strong links will be tightened more and more, while the scans, connected by the weak link, are moving further and further apart (fig. 3.43).

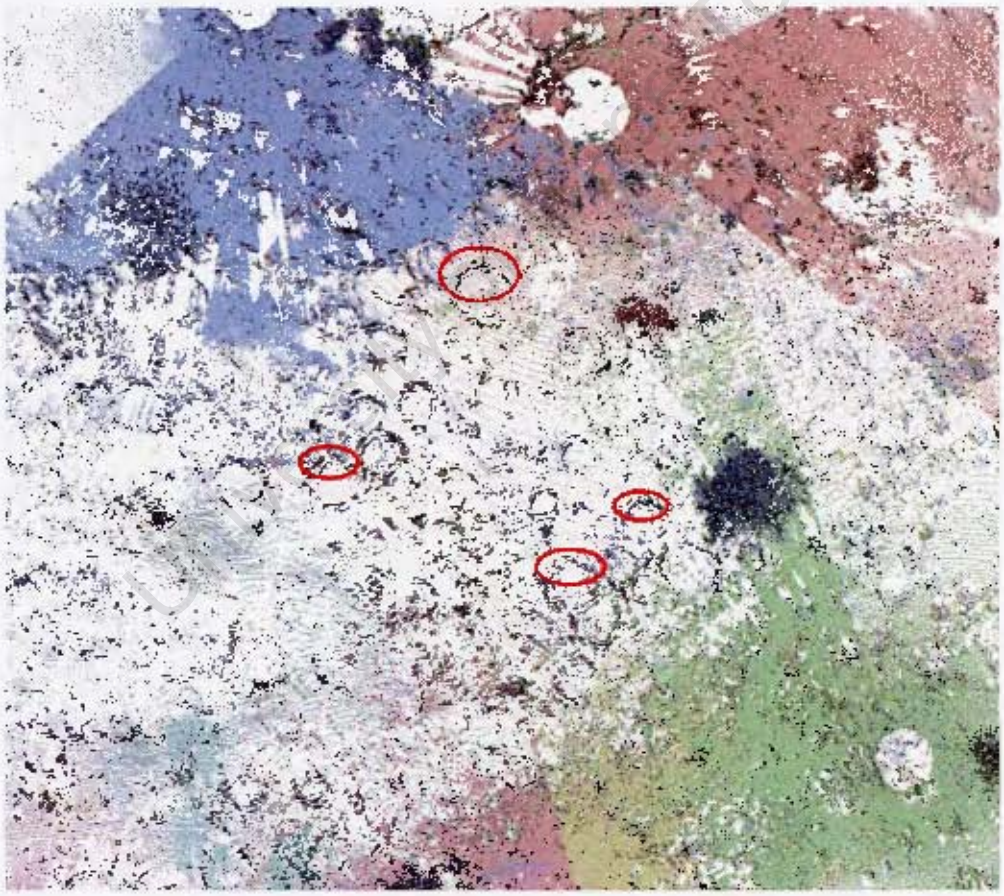

Figure 3.43: Result of global registration process with Meshlab. Note the misalignment (some areas are highlighted in red) between the two major groups of scans, even though they were well-aligned before the process was started.

### 3.6.3 Registration with a Skeleton Model

It is thus suggested, to follow another procedure. First, all scans are reduced to a lower resolution to speed up the process. Afterwards they are roughly aligned together to get an overview of the various scanner locations and what area each scan covers. Based on this rough alignment, the very minimum of scans covering most of the area are identified. These should be aligned as good as possible, in higher resolution, maybe with -if accessible- the help of a global registration method, if the links are equally strong between scans. The resulting model can then serve as a base model or reference model, to which all other scans are registered one-by-one. This approach was confirmed by the author on a large dataset of the Zamani Project, Songo Mnara, Tanzania, consisting of more than 600 scans (fig. 3.44).



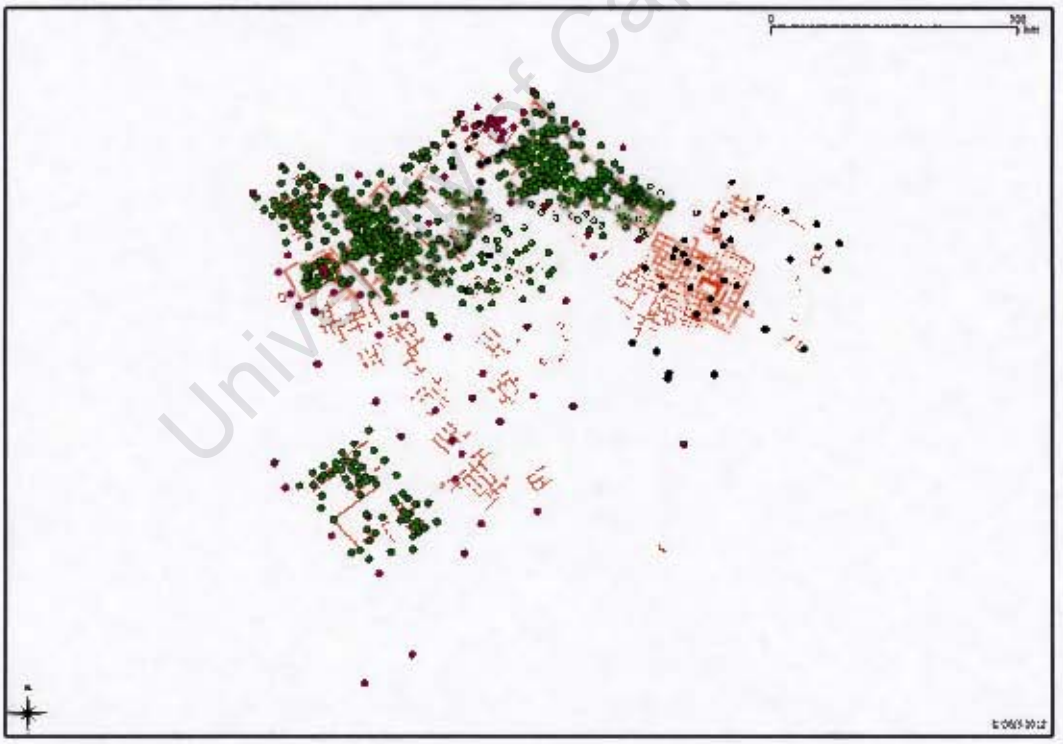

Figure 3.44: GIS of Songo Mnara showing the scan positions. The green points mark the scans which were aligned to the skeleton model (purple and black points)

### 3.6.4 Registration Software

To perform the registration, several tools are available in here: Scanalyze and Meshlab, both open source tools, MeshAlign[14] and proprietary software such as Geomagic, Polyworks and Leica Cyclone.

Despite the variety of registration software packages, the usage of the free software Scanalyze is advantageous:

- Speed - none of the other software tools allows such a fast navigation with large meshes, consisting of several millions of triangles
- Flexibility - after alignment, the 3D transformation can be saved as a XF-File containing a 4x4 transformation matrix. This matrix can be applied, with the tool ptxTransform by the Zamani-project, to different versions of the same scan, for example cleaned and un-cleaned scans allowing the registration process to be done in parallel to the cleaning
- Scalability - Alexander Agathos compiled a 64Bit version for Windows[15] allowing to load and register extremely large datasets, for example, 80 million triangles on a machine with 4GB of RAM
- Compatibility - as Scanalyze can read the PLY format, it fits very well into this workflow
- Free access - as it is open source software, no license has to be acquired.

### 3.6.5 Reducing Initial Alignment Efforts

As shown above, the registration process is still heavily based on user interaction. The establishing of an initial alignment requires a large amount of time. Even though some automatic alignment is offered in software, such as in Meshlab, based on the "4 Points Congruent Set" approach of Aiger et al. (2008), described in section 2.2.3.4, the alignment is not always successful and still quite slow. Such alignment tools might become more efficient in the future, but so far it remains mostly a manual task.

However there are some routines which can help in minimizing this effort. Firstly, the manual transformation approach seems to be much faster than the selection of three or four common points for an initial alignment. Best experiences in Scanalyze were made by viewing the scene from above, and shifting and rotating the new scan

[14] Meshlab implements parts of MeshAlign. Both Tools were developed at the VCG lab of the I.S.T.I., Pisa, however a license is required for MeshAlign.

[15] http://users.iit.demokritos.gr/ agalex/Scanalyze/ScanalyzeWindows/Scanalyze64.rar

until it approximately fits the target scans. If the scan is levelled, the scan only needs to be rotated around one axis and avoids a change of the viewer's perspective, which would enable to rotate around the other axis. Levelling a scan should thus become a habit during data acquisition. If also the orientation of the scanner during scan acquisition was more or less set into the same direction, the initial alignment would reduce to a simple shifting.

Secondly, the resolution of the scans should be reduced drastically for the initial alignment. Scans can easily contain more than 15 million triangles which are too many to just obtain a rough registration. During this initial stage, the scan needs to be rotated and shifted often and thus a smooth navigation experience will save a lot of time. Ptx2Ply can be employed to perform a simple sub-sampling of the scan. A value between four and six usually reduces a scan with 20 million triangles to about 500,000-1,000,000 triangles, which is more than enough for initial alignment purposes. Once all scans are roughly aligned the resolution can be increased again and the ICP should be run once more to refine the alignment.

### 3.6.6 Registration with Structure from Motion

Another idea, suggested here, is to recover the initial alignment of scans with the help of structure from motion tools. Similar to Spin-Images, (Johnson and Hebert, 1997), mentioned earlier, the 3D models are processed as 2D images. The idea presented in here however, is to create multiple images of each 3D scan in small, regular intervals with a virtual camera, which rotates around the scan (fig. 3.45(1)). Each scan is then represented by a group of images. The images from all scans can then be processed with SfM tools (fig. 3.45(2)).

The small steps ensure that SfM tools will correctly establish the relative camera positions for each individual image of a scan. Every oriented image can then become a potential match with images from other scans and can thus link scans together. Producing more images from various angles of each scan increases the likelihood that a match between scans is found.

Because these images are rendered images, the in- and external orientation parameters of their corresponding virtual cameras are known. Hence, the coordinates of the cameras in each group, in their local system and in the SfM-system can serve as control points for a standard 3D-transformation to establish the transformation parameters for each scan (fig. 3.45(3)). As the resolution of the virtual images will usually not match the resolution of the scan, a perfect alignment is unlikely, but at least this registration can serve as an initial alignment.

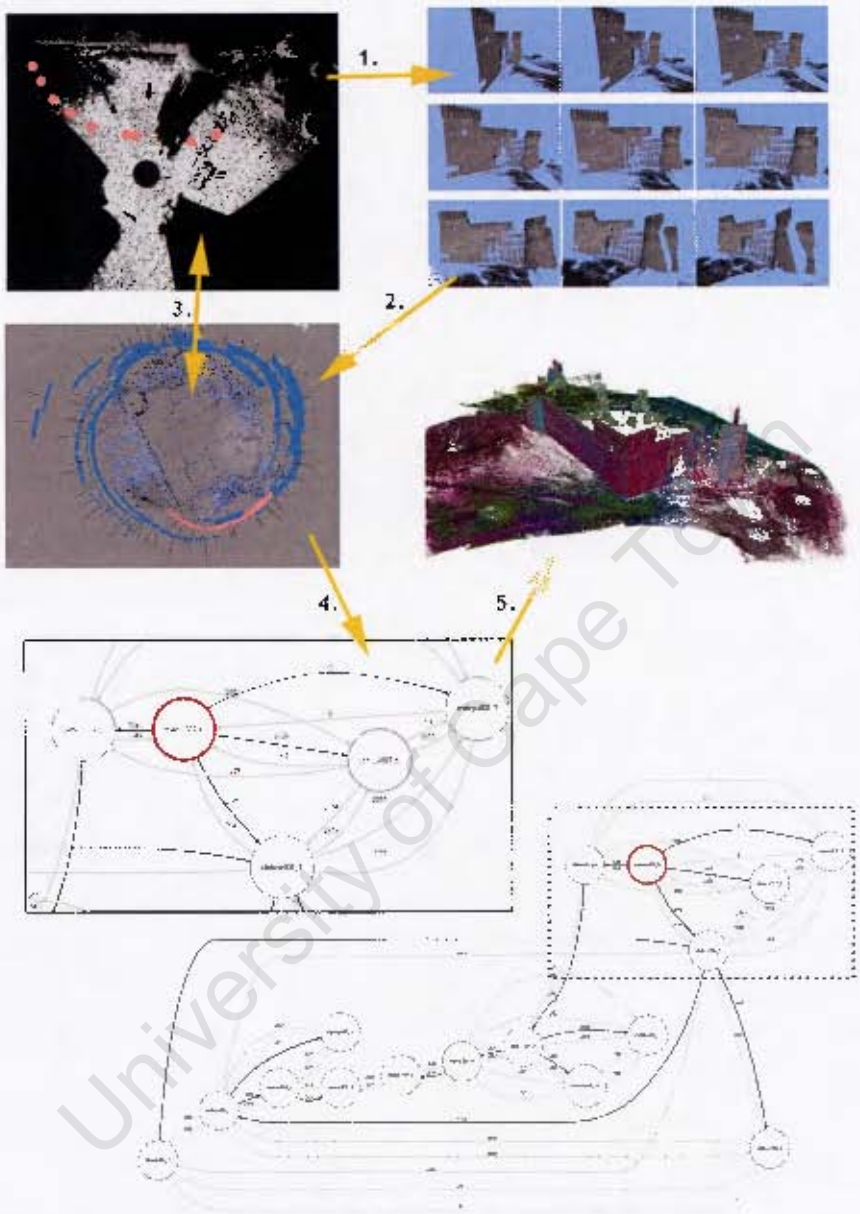




Figure 3.45: Scan-registration process with SfM: 1: several views of each scan are created. 2: these views are then registered via SfM-methods. 3: a transformation matrix for each scan can be derived by performing a 3D transformation between the positions of the virtual cameras in their local coordinate system and their position in the SfM model. 4: the point matches between scans are visualized in a graph-structure. Lower numbers refer to a "stronger" link. The "Dijkstra" method establishes the shortest path through the graph. 5. The graph can be translated into a batch-file which fine-registers all scans.

To automatically refine this "rough" registration with ICP, it is necessary to know, which scans share some overlap. This information can also be derived from the SfM process. For each found image pair, the system records how many matches were found and thus provides the information which scans are overlapping. The amount of matches also shows how strong the link between each scan pair is. More matches can be interpreted as large overlap. This information can be converted into a graph structure, which visualizes the links between scans. By finding the shortest path through the graph, for example with the Dijkstra method, (fig. 3.45(4) one can establish, which scans need to be registered together and in which order.

First experiments showed promising results (fig. 3.45(5)), but at the moment, this procedure is not reliable enough to be incorporated into a productive environment. However, in the future, by modifying the SfM-matching tools, one could align normal photographs as well in the same process for texturing purposes.

### 3.6.7 Resulting Workflow - Registration

As described above, registration remains a largely manual process. But the workload can be drastically reduced by following the approach from above: first, all scans (low-resolution) are roughly aligned together, then the most essential scans for a skeleton model are identified, and finally, all scans are fine-registered in high-resolution onto the skeleton model.

Please refer to figure 3.46 for an overview of the workflow of this section.

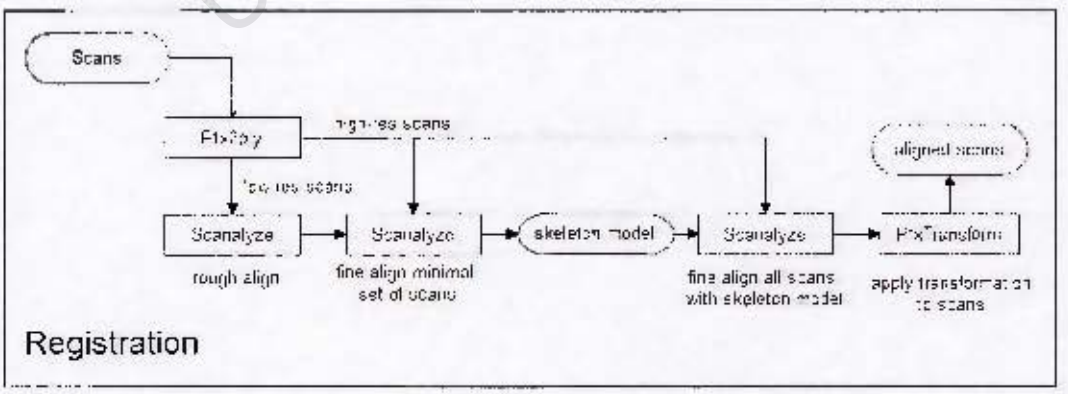


Figure 3.46: Diagram of registration workflow.

## 3.7 Cleaning

While scanning in the field, especially cultural heritage sites, which are still in use, it is usually not possible to obtain a completely "clean" scan, free of people, animals and other, undesirable objects, not belonging to the original and historical state of the site. One reason is the advancement of technology and thus the scanning speed, which does not leave the operator much time to escape the laser-beam. In addition, one simply cannot close off the whole, or even just part of the heritage site, as they are often still in use or simply became a tourist attraction. Further, over time, people might have done permanent additions to the structure, which obviously cannot be removed while scanning.

These items can be classified into three categories (see fig.3.47 for examples):

Static Items static objects which do not move during the course of the fieldwork, such as scaffolding, permanent (modern) additions to the structure, tree trunks or neighbouring structures

Movable Objects This includes all objects which move while being scanned or have moved while the instrument was moved to another position, such as people, animals, equipment parts or boxes, cars, vegetation

Technical Artefacts Laser scanners do have some technical limitations, for example, the inability to scan into water. The beam is either reflected off the water surface or at least refracted. Hence, if there are resulting data points, they might seem to be at the right position, but they are not reliable. The same effect appears with glass and mirrors. Dust and water particles in the air (fog) might result in random data points or clouds. Other artefacts are caused by the false edge problem: The laser beam, due to its not unlimitedly thin footprint, can be split on the edges of an object: the scanner will then receive two or more reflections, also from the next objects in the line of sight. Instead of being filtered, these reflections are sometimes averaged and result in points floating in space.

This list demonstrates that a cleaning step in the pipeline is unavoidable. But only the fact that an object is not part of the desired structure, does not mean that it is not of use. Some are static enough to be useful for registration by the means of surface matching, especially if there is only little overlap otherwise. Performing the cleaning after, or even in parallel to the registration is thus a viable option.

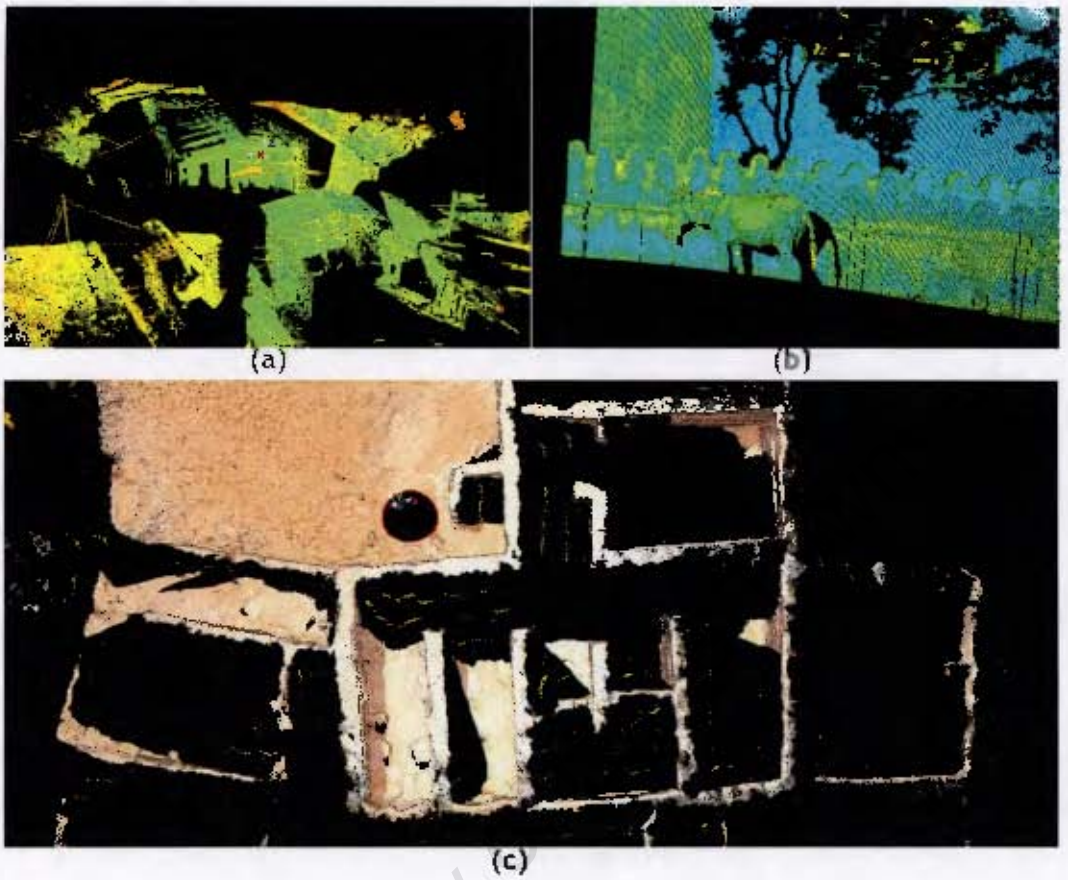


Figure 3.47: Examples of items to be cleaned: (a) scaffolding, (b) animals, (c) "false edge"-artefacts, appearing as isolated lines.

### 3.7.1 Software Tools

Leica Cyclone is capable to load very large point datasets and is thus predestined as a cleaning tool. It also allows importing and exporting the files as PTX-files, guaranteeing a smooth workflow without too many file-conversions and the preservation of the scanning grid. However, Cyclone only shows a portion of the point cloud at a time, depending on whether a user-set limit of displayable points is reached. The software continuously adapts the amount of points being displayed on screen, depending on the focused area and the zoom level, removing points from memory, which are not in view at present. This works nicely for single scans, but when looking at an entire dataset, which consists of several scans, only a reduced version of each cloud is shown. Thus, one can never be sure of selecting and removing all points unless the point cloud is "unified". Unification imports and sorts all points from all scans into a new database, rejecting the information about the scanning grid. This would complicate the whole

pipeline, as only meshing algorithms which can deal with arbitrary point clouds could be used for surface reconstruction, which basically means Delaunay, with all its pros and cons.

Pro's of cleaning individual scans

- Scanning grid can be maintained, allowing easy recovering of point normals and creating simple surfaces
- Scans can be redistributed, so other researchers can do their research on practically original data, not affected by registration errors
- Cleaning can be done in parallel to the registration process. Cleaned PTX files are interchangeable with the originals, once both processes are completed

Contra's of cleaning individual scans

- Slow process, as each scan needs to be cleaned individually. This often means removing the same objects from several scans.
- Difficult to sometimes distinguish in the point cloud between surface and unwanted objects in low resolution areas. Combining different scans into one point cloud could help in identifying parts to be removed

PointTools is another commercial software able to load large datasets and thus would allow the cleaning on the entire dataset. The performance seems to top the performance of Cyclone and it also offers more selection tools. But, at present, it is not able to export into PTX or any other "gridded" formats, which is crucial to this pipeline. However, PTX-export was announced for the future. The only useful export format at present is a X,Y,Z format with point normals, which could be used, at least as an input for the Poisson-surface reconstruction algorithm.

A compromise between cleaning individual scans and all scans at-once, would be, similar to the reconstruction process, to subdivide the datasets into regular sized subsets with ptx2ply. Cyclone would be able to show all points of a block at the same time without unification. After export, the scans, even though combined in one large PTX-file, can be separated again with ptx2ply. Tests have shown however, that the performance of the navigation within Cyclone is still quite slow. However, as it is easier to recognize individual objects and to remove all appearances of an unwanted object with one selection process might, overall, save a lot of time. The only real drawback is the necessary pre-alignment of all scans. Hence, the cleaning cannot be executed

in parallel to the registration, which does not affect the total working hours, but the production time, when working in teams.

### 3.7.2 Methods to Remove Unwanted Data

Cleaning scans is, at present, still a mostly manual process. There are however some solutions, already implemented in soft- and hardware, which could decrease the workload.

Scanners with a spinning mirror are usually scanning back and front of the scanner at the same time, which means they only need to scan 180 degrees in the horizontal plane. But if a full 360° scan is performed instead and both halves are recorded, one obtains two samples per scan point and can thus filter objects which moved during scanning. As the scanning speed increases with every generation of laser-scanners, presently completing a 180 degree scan in less than 90seconds (3cm at 25m, HDS6100), this approach could become more and more a practical solution. Laser-scanner manufacturers, such as Z+F, implemented such a function into their software packages. However the practical application could not be tested in here.

Leica Cyclone offers some semi-automatic segmentation approaches, which assist the user in the selection of artefacts. Surface detectors are looking for smooth, planar patches, selecting anything which deviates too much from it, which can then be deleted. People walking on the street or cars on the road are thus easily filtered. However, it is not a very fast approach and an experienced operator is likely to manually select these artefacts as quickly or even faster.

For the future, semi-automatic methods, operating on classification and segmentation algorithms, which would identify and separate different materials and objects, seem to be the only practical solution to the cleaning process. Vegetation for example, would produce a very irregular surface, classifiable by **normals** and **depth variations** between neighbouring points. Also **border detection** and filtering of **small components**, which consist only of a few points or the **intensity**-values of the laser-signal, can serve as another indication for vegetation or parts of the same material.

### 3.7.3 Resulting Workflow Cleaning

The only practical solution at the moment is to clean every single scan in Cyclone. Cyclone is the only software in this pipeline, offering no support for PLY-Format. But as it can read and write PTX files, there is no additional conversion necessary. Registration information in the form of XF-files can be applied with a self-developed tool, called PTX-Transform.

For the future, when other software, such as Point Tools supports the import and export of PTX files, the scans should be split into sub-blocks before cleaning. For this reason, Ptx2ply was extended to also output PTX files again and split the data into sub-directories. This would imply however, that the cleaning can only be done after the registration, not in parallel any more. First tests in Cyclone have shown that this would significantly reduce the cleaning time, but in the current version (7.0), the rendering is not fast enough, even on a high-end machine.

Please see figure 3.48 for an overview of the workflow of this section.

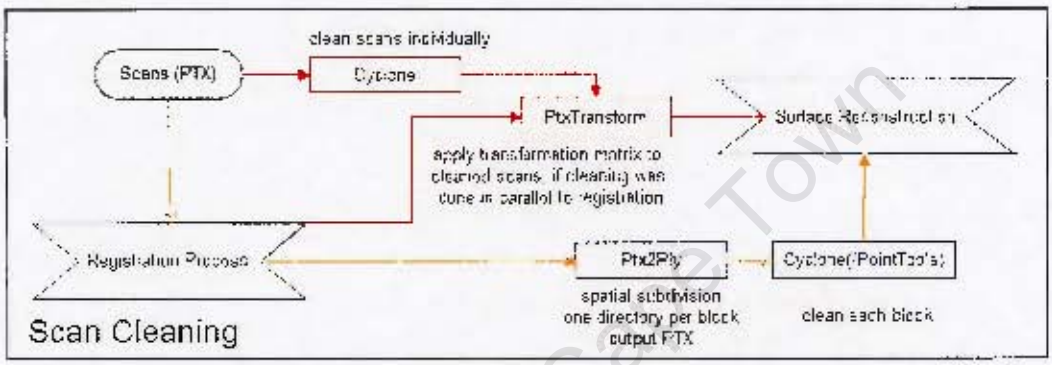


Figure 3.48: Diagram of workflow for scan cleaning.

## 3.8 Scan acquisition

The acquisition of scans became a very simple task. Common medium range systems (Leica, Z&F, Trimble, Faro) can perform spherical, full-dome scans of a scene by just pressing a single button. The scanner is moved to various locations to scan the entire site. The following section rather serves as a collection of methods and routines for scan acquisition, which would minimize efforts during post-processing.

The most time-consuming manual task is the initial alignment of the scans, which, as mentioned above, could benefit of a consistent orientation into one direction and levelling of the scanner. Many modern instruments have a tilt-sensor or even a compensator built-in, which should thus always be activated, if possible.

For internal reference, each scanner has a fixed internal horizontal zero-position. If it cannot be identified from outside any other marks on the casing should be employed to orientate the scanner consistently into one direction before each scan. This will reduce the manual effort of rotating the scan in the registration software. Many modern smartphones do come with an internal digital compass, which would be a very precise way of measuring the orientation.

Global navigation Satellite System (GNSS) sensors could also assist in determining the position of the scan, minimizing the need for manual translations. However, GNSS signal is not available everywhere, especially inside, and usually not accurate enough, unless differential GNSS is employed. It thus seems much more feasible to record scan positions roughly in a GIS and to convert these coordinates into a translation vector.

Overlap is most important for the alignment process without targets. It is thus recommended to always scan a full dome scan, as also the objects not in focus of the current survey can become useful during registration, such as parked cars or buildings nearby.

Hole filling should be avoided for the documentation of cultural heritage sites by scanning as much as possible. The philosophy should be to rather scan in lower resolution than to not scan at all. This approach might take more time for registration and cleaning, but it will take even more time to create artificial fill-in patches. In some cases, images for photogrammetric purposes, together with SfM software can replace a scan, but as it can only be confirmed in the office, if the photogrammetric documentation was successful, a quick laser scan might be safer.

### 3.8.1 Resulting Workflow - Scan Preparation for Processing

For the processing pipeline, the only data-acquisition task is to convert the laser-scan-files to make them compatible to the remaining workflow. As mentioned above, PTX was chosen as the standard scan-format, as it retains the scanning grid and is supported by all major laser-scanner manufacturers (Leica, Z&F, Trimble, Riegl, Faro, Optron).

# Chapter 4

# Summary of the Proposed Workflow



Unifying all recommendations for the individual steps, the processing pipeline is designed as follows:

- All scans are converted from scanner manufacturer specific file-formats to the PTX-format.
- **Cleaning**
  - The scans are cleaned in Cyclone and exported back into the PTX-format.
- **Registration**
  - The scans are triangulated and converted to PLY-files with ptx2ply, and roughly aligned in Scanalyze.
  - By refining the most important scans, a skeleton model is built, which is used as a reference to align the remaining bulk of scans.
  - The found transformation matrices are applied to the PTX-files with ptxTransform.
- **Surface Reconstruction with PlyMC**
  - All PTX files are then converted into PLY-files and split into sub-blocks.
  - Each sub-block is automatically meshed with PlyMC via a script.
- **Surface Reconstruction with Geomagic**
  - All PTX files are again converted with Ptx2ply, but instead of PLY, the scans are merged into an ASCII-points-file for the processing in Geomagic. The file only contains point coordinates and intensity values.

  - The point cloud is then sub-sampled, noise filtered and meshed in Geomagic.

- **Hole Filling**
  - Missing parts are modelled with SFM methods.

- **Surface Reconstruction, merging of various models**
  - SFM-Patches, and the Geomagic model, or part of it, are merged with the overlapping sub-blocks of the PlyMC process in an additional PlyMC loop.

- **Simplification**
  - The model is cleaned of artefacts in Meshlab and simplified to lower resolutions with the Quadric Error filter of Meshlab.

- **Texturing**
  - Images are matched with SfM tools, such as Bundler, SFMToolkit or Agisoft Photoscan, which produce a "bundle.out"-file including all camera positions and orientations.
  - Common points are identified on the SfM- and the laser-scan-model.
  - The list of common points and the "bundle.out" file are processed with sfm2Texture, which produces one XML-File per image.
  - These XML files can be loaded into TextAlignSuite to fine-align the image to the model. For this step the Geomagic model is used, as it carries intensity information, which can be used as a guide for the alignment.
  - If additional images are necessary for the texturing, which are not part of the SfM reconstruction, they can be aligned manually with TexAlign via the identification of common points. This alignment can then also be refined in TextAlignSuite.
  - All aligned images are projected onto the PlyMC model with TexTailor.

In figure 4.1, the proposed workflow is displayed in a unified diagram. This pipeline can be regarded as the "recommended" approach. Depending on the requirements and nature of each site it can become necessary to swap or replace some methods. For various alternatives, please refer to the individual sections.

In the following chapter, the established work-flow is verified on a large dataset.

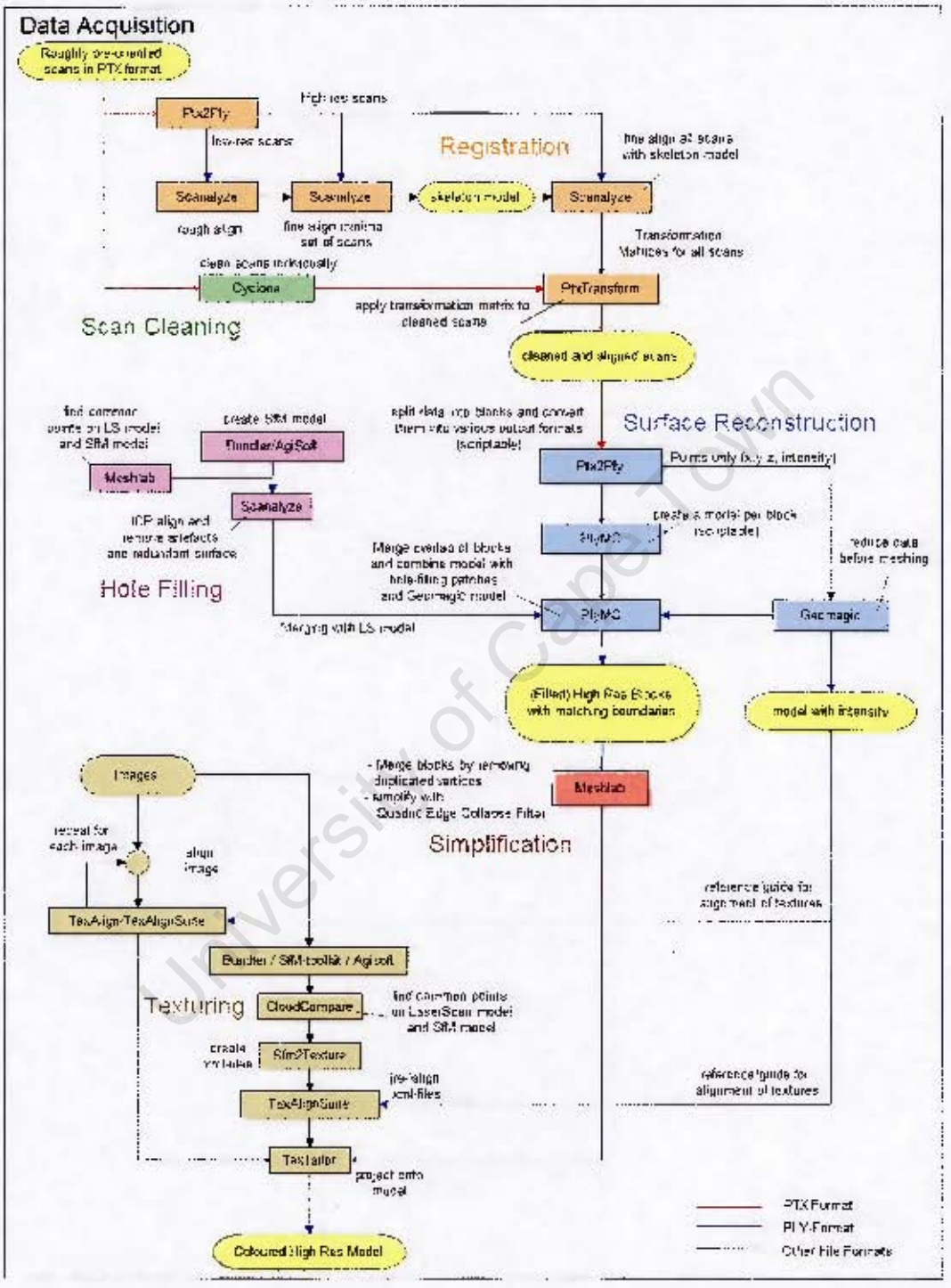


Figure 4.1: Diagram of the entire workflow. File formats are indicated by different line colours. Black lines are depicting other file formats, such as transformation files (XF), Point files (for Geomagic), Images, or XML files.

# Chapter 5

# Evaluation of the Developed Workflow

In this chapter, the previously suggested workflow is briefly demonstrated on the example of the Gereza, a Portuguese fort on the island Kilwa Kisiwani, Tanzania (fig5.1). The desired output is a coloured 3D model in a resolution of 2cm.

## 5.1 Creating a 3D model of the Gereza

### 5.1.1 Data Acquisition

The data was acquired in December 2009 with a Leica Scanstation C10 and a Leica HDS 6100. The C10 was mainly used for outside scans, because of its longer range, while the faster 6100 was used for the more complex inside. The resolution was set to 3 cm over 25m, or the next closest setting thereto. Scans of small rooms or narrow passages where recorded in low resolution, 5cm over 10m.

In 2009, restoration work on the Gereza was in progress during the field campaign and some parts were impossible to scan, due to scaffolding in the line-of-sight (fig. 5.1a). In 2011, during an additional scanning campaign, a few more scans of the East façade were taken.

In total, 125 scans were recorded: 21 with the Scanstation C10 and 99 with the HDS 6100 in 2009, plus five scans with the HDS 6100 in 2011. For most scans, the instrument was oriented into a specific direction and levelled precisely by a built-in compensator (C10) or tilt-sensor (6100). The total size of all scans in uncompressed PTX-format (ASCII) is 29.8GB, which equals 724,727,186 points.

### 5.1.2 Data Cleaning

Due to the restoration work on the Gereza, it was unavoidable to scan scaffolding and workers. Hence, the individual scans had to be cleaned in Leica Cyclone, which

Figure 5.1: The Gereza: East façade (a), South and West façade (b), inside court (c)

could read the native scanner formats of the C10 and the 6100 without prior conversion. Cyclone does mostly only offer manual selection tools, such as a polygonal fencing tool. The cleaning step was thus a straight-forward, but time-consuming task, executed by the Zamani team. During the cleaning process, 86 million points of scaffolding, people and surrounding landscape were removed (fig. 5.2). Afterwards, the scans (638 million points) were exported individually into the PTX-file format.


Figure 5.2: Before (a) and after (b) the cleaning step.

### 5.1.3 Scan Registration

The registration of the scans was done by the author in two steps, a rough- and a fine-registration in the free software Scanalyze. During rough-registration, the scans only needed to be moved into a good initial position, and thus the scans could be converted to a low resolution to allow fast rendering and navigation in the software. Scanalyze requires PLY-files, which is why the PTX-files had to be converted with the tool ptx2ply first. During the conversion process, the resolution of the scans was reduced by a 4x sub-sampling, meaning that only every 4th line and row of the scan-matrix was used. Scans in low resolution were not sub-sampled to not lose too much detail.

As many scans were already oriented approximately into the same direction during scanning, the initial alignment task was reduced to shifting the scans into position

and executing the ICP algorithm of Scanalyze several times. The closest-points search distance of the ICP was reduced from 20cm to 1cm during several iterations.

During the rough-registration step, the most important scans were identified (22 scans), which cover large portions of the object or link the inside to the outside. These few scans were then re-converted with ptx2ply, into a higher resolution (sub-sampling=2) and re-registered again. The result served as a skeleton model, to which all other scans were then fine-registered, also in high resolution (fig. 5.3). To ensure that the final model is levelled, one levelled scan should be fixed throughout the entire process and serve as a base scan, to which all other scans are registered to.

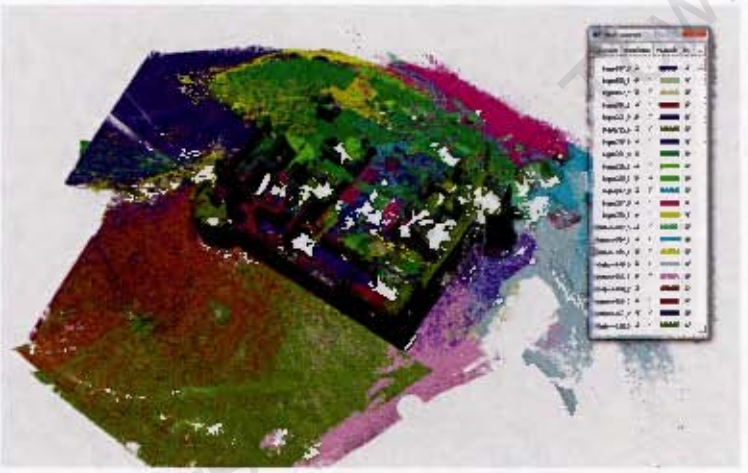

Figure 5.3: The skeleton model of the Gereza, which consists of 22 scans.

The result of a registration in Scanalyze is a XF-file, containing a 4x4 transformation matrix, which registers the scan into the final coordinate system. These matrices were applied to the individual scans with the tool PtxTransform. To view the entire, registered point set, these PTX were imported back into Cyclone (fig. 5.5a).

For easier processing during the surface reconstruction step, where the object will be sub-divided into blocks, it is recommended to, at least, roughly align the main walls of the structure with the coordinate system axis.

The additional five scans of the field campaign in 2011 were also aligned, but had to be excluded from the merging process, as too much of the surface was modified during the repairs in 2009. To fill in at least the parts, which could not be scanned in 2009, the required area was cut out of one of the scans (see fig. 5.4).

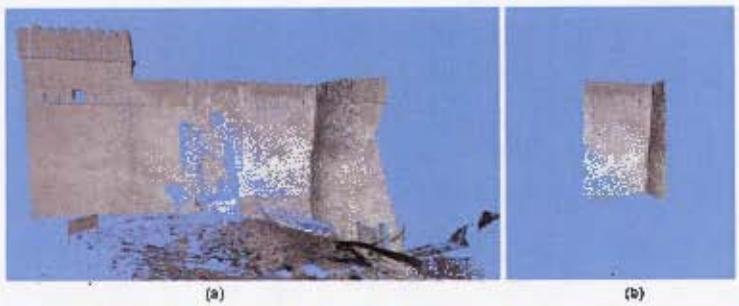


Figure 5.4: (a) Scan, which is part of the 2011 dataset and (b) the resulting patch, which was used to fill the holes of the scaffolding (comp. fig5.2).

### 5.1.4 Meshing

Following the recommendations in section 3.4.4.5, two versions of the model were produced, a detailed model with PlyMC, and a fill-in model with Geomagic, which also carries intensity information.

To minimize reconstruction efforts a bounding-box was defined in Cyclone, with the aim to exclude all non-relevant points around the building from the reconstruction process. When exported to a text file, the bounding-box, also referred to as a limit-box, can be included into the scan conversion process of ptx2ply (fig5.5).

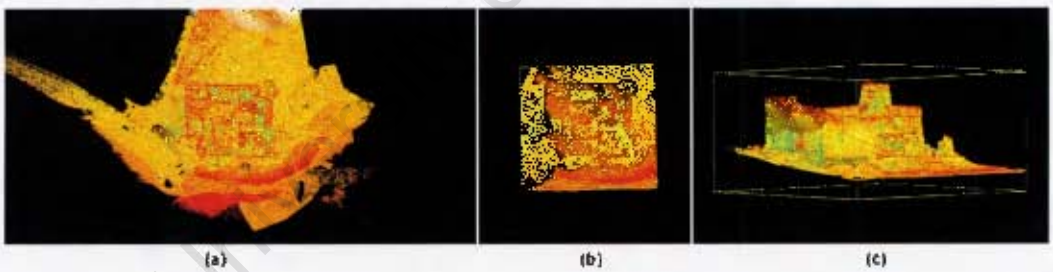


Figure 5.5: (a) all registered scans in the form of a point cloud. (b and c) Limit-box, which defines the volume to be meshed.

#### 5.1.4.1 Geomagic Model

For the Geomagic model, the scans were converted with ptx2ply into a combined vertex file, containing the points of all scans, but sub-sampled by a factor of 4. Intensity values were converted into greyscale values, using the intensity ranges specified in section 3.4.4.4.

Due to the sub-sampling, the ASCII dataset was reduced to 1.51GB which equals 38 million points. This is still too large for the surface reconstruction in Geomagic. The

point cloud was thus further reduced via a grid-sampling of 3cm in Geomagic Qualify v8. This reduced the dataset to 5.5 million points, which could then be converted to a surface by using the option "surface wrap" option within Geomagic Qualify. For the surface reconstruction, the software offers four noise reduction settings (min, medium, max and auto), of which "min noise" was used. The, comparably, low resolution is acceptable, as the Geomagic model is only needed to produce some patches for hole-filling purposes and to retain the intensity information for the texturing process. See figure 5.6 for the resulting model with intensity information or figure 5.7 for the model shaded with ambient occlusion.


Figure 5.6: The model produced with Geomagic Qualify, which retained the intensity information.

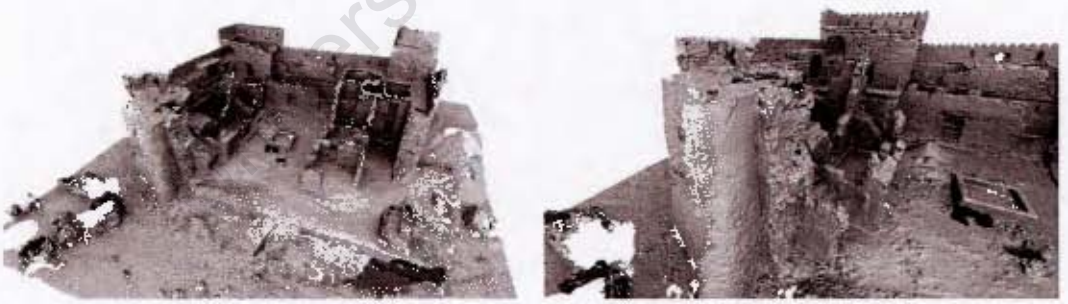


Figure 5.7: Result of the Geomagic Qualify process, shaded with ambient occlusion. Note the effect of the grid-sub-sampling on the tower (b).

#### 5.1.4.2 PlyMC Model

For the detailed PlyMC model, all scans were triangulated and converted to PLY with ptx2ply. As the aim of this model is to be as detailed as possible, the full dataset

was used without any sub-sampling. For efficient processing, the data was split into 32 sub-volumes, with an overlap of 15cm in between. The sizes of triangles were restricted by the tessellation depth threshold, limiting edges to a length of 5cm.

The scans were now ready be processed with PlyMC (32-bit), for which the following parameters were set: the Volumetric Diffusion option, to expand the surface along boundaries of the final mesh for hole-filing purposes, was set to the minimum (=1). In addition, the geodesic quality option was deactivated, which smoothes triangles of a scan, if they are close to a boundary. Instead, the quality determined by ptx2ply was used, which mainly penalizes triangles which are far away and in a flat angle towards the scanner. The surface was attempted to be extracted at a resolution of 1.5cm.

The 32-sub-volumes were processed automatically with a script by Roshan Bhurtha, which starts the PlyMC process for each volume. But even though the data was already split into sub-volumes, the individual blocks were still too large to be processed in one piece (the sizes of the volumes varied between 0.21GB-8.1GB). For this reason, PlyMC was set to process the data with a further subdivision into four blocks. Hence, the resulting surface of the process was split into 112, partially overlapping sub-blocks, not counting the "empty" blocks, containing no data (fig. 5.8b).

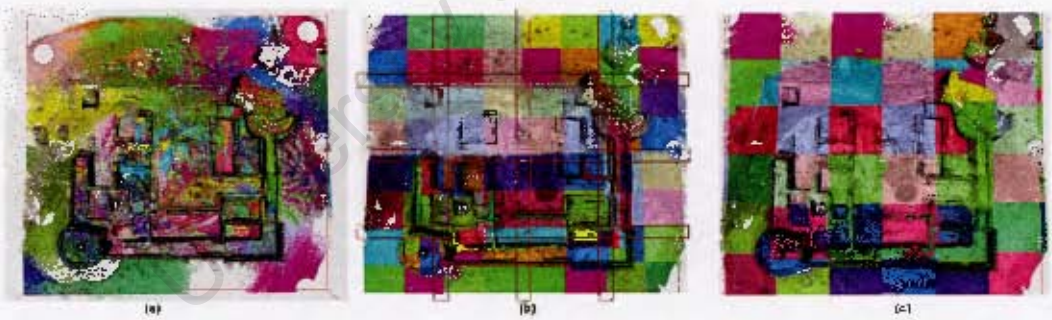

Figure 5.8: (a) all scans registered together. (b) The data was subdivided twice with ptx2ply and PlyMC into 112 blocks. Note the overlap of the ptx2ply process in between some of the blocks. (c) The second iteration of PlyMC divided the dataset into 256 blocks, of which only 96 contained data.

The processing with PlyMC is very robust, and straight-forward. No major problems could be detected. The user just has to find and specify the optimal settings for PlyMC and ptx2ply. The correct settings largely depend on the required resolution and on the amount of points, which determines the required subdivision.

#### 5.1.4.3 Hole Filling and further Merging

Examining the PlyMC model revealed only a few larger holes. These can be classified into two kinds of holes. The one is caused, if the tessellation depth threshold was set too restrictive (fig. 5.9a). This can be confirmed by comparing the PlyMC model either with the point cloud, or simply with the Geomagic model, which is not as sensitive to the TDP, as described in the previous chapter. To close a hole cause by TDP, a patch can simply be cut out from the Geomagic model, for example in Meshlab (fig. 5.9c).

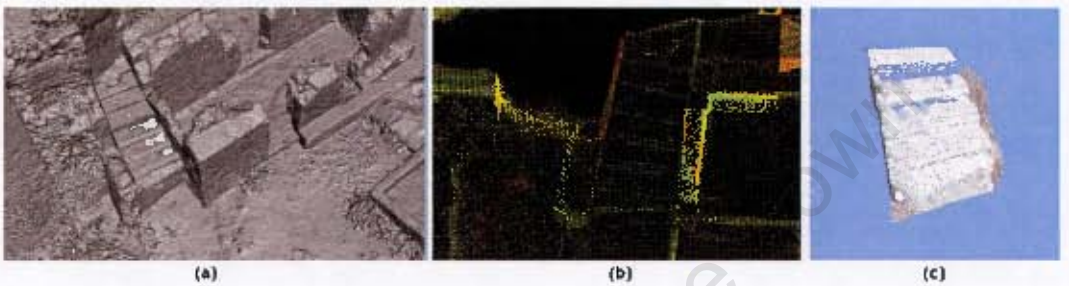


Figure 5.9: Image (a) shows a hole in the PlyMC model, which was not filled even though point data for this area exists (b). Image (c) shows the patch extracted from the Geomagic-model

The other kind of hole is a real data void, where no points were recorded (see fig. 5.10a). To close these holes accurately, other sources of data, such as Photogrammetry have to be employed. For the case of the Gereza, Agisoft Photoscan was employed to create a SfM-model of a room, where a corner was not scanned properly. The process in AgiSoft is easy and largely automated: first, the images were loaded, matched and aligned, and in a second step, a model was created. To scale and register the model with the PlyMC-Mesh, some corresponding points were selected on both models. Following this, the SfM-model was exported and fine-registered with the laser-scan model in Scanalyze via ICP (fig.5.10b). Once the patch was aligned, it was manually cleaned of most of the overlapping surface to not influence the other, scanned parts (fig.5.10c). This is especially necessary, as SfM-models of uncalibrated cameras are much less accurate than the laser scans and are often also in much lower resolution.

These hole-fill-patches were then merged with the 112 sub-volumes of the PlyMC model in an additional, second merging step with PlyMC. As both, the fill-in surface derived of the Geomagic model and the SfM-patch are in much lower resolution compared to the PlyMC model, the weight of the two patches for the new merging process was lowered to the minimum (0.01). This ensured that the detail of the first PlyMC model was retained while the patches only contributed to fill-in the holes.


Figure 5.10: (a): Another hole in the corner of a room. (b): a Patch was created from photographs with Agisoft Photoscan and aligned with ICP in Scanalyze. (c): the final patch.

The new merging process was set to a resolution of 2cm and a sub-division of the data into 256 blocks. The amount of blocks was doubled, as attempts with less sub-division were unsuccessful. This additional merging step also removed the overlap in between the sub-blocks and produced new sub-blocks with well-matching boundaries (fig. 5.8c). In the end, only 96 blocks contained data.

#### 5.1.4.4 Surface Reconstruction without Subdivision

To be able to evaluate the effectiveness of the subdivision approach, another high-resolution dataset was created with ptx2ply, but without the creation of sub-blocks. PlyMC was set to extract the surface at a resolution of 15mm and to split the data into 128 blocks, but the process failed. Also, another test with 256 blocks was unsuccessful. Only a test with 1000 sub-divisions was successful but the process took 16 hours and 40 minutes. In comparison to this result, the previous two merging steps together took only 4 hours and 55 minutes.

### 5.1.5 Simplification and Cleaning

The resulting, second PlyMC model consists of 35.5 million triangles, which could be loaded and processed with the 64-bit version of Meshlab. As PlyMC creates blocks with nicely matching boundaries, the blocks could be merged automatically, simply by removing the duplicated vertices on the boundaries with a filter of Meshlab. The model showed a lot of little floating surfaces, most likely, due to a large tessellation depth threshold. Also some irregular boundaries near vegetation areas (fig.5.11) were present. These artefacts were largely removed with filters in Meshlab, cleaning out small isolated pieces and triangles on boundaries. The cleaning process reduced the

model to 32.6 million triangles. The final, cleaned model can be seen in figure 5.12. The result of the hole-filling step is presented in 5.13.

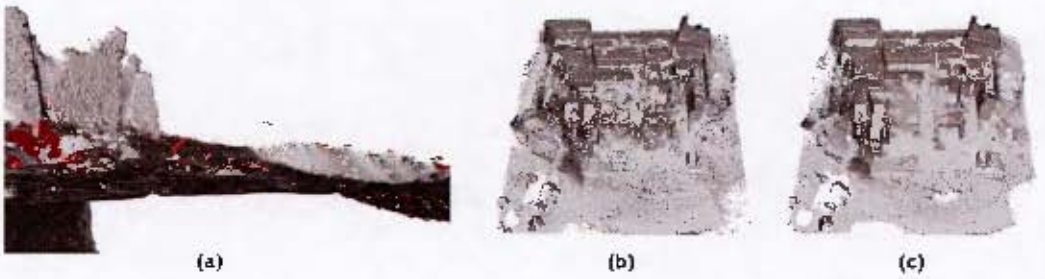


Figure 5.11: Floating artefacts produced by the merging process (a+b). These artefacts were largely removed with the help of the filters in Meshlab (c).


Figure 5.12: The final PlyMC-model in 2cm resolution.

As the model did fit into main memory, it was possible to simplify the model inside Meshlab and the Quadric Edge Collapse method without any problems. Three versions were derived, consisting of 8 million, 3 million and 100.000 triangles (see fig.5.14).

### 5.1.6 Texturing

Following the recommendation of the previous chapter, SfM-tools were employed to texture the Gereza. Agisoft Photoscan was used to analyse 753 images, taken with a Fuji Finepix S5700 and a Nikon D200, of which it managed to match and align 462 (see fig5.15a). The remaining ones, mostly of inside rooms, did not provide enough overlap to be matched with the larger set, but it is expected that it is possible to align them in additional, separate SfM-projects.

The matched features between the images can be seen as a Point Cloud, which was exported together with the estimated camera parameters as a "bundle.out" file, which is readable by the converter tool sfm2texture, developed within this research. In

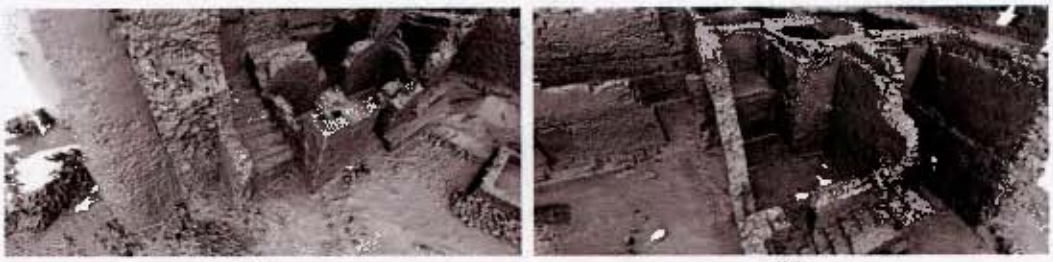


Figure 5.13: The result of the hole-filling step, after the cleaning.



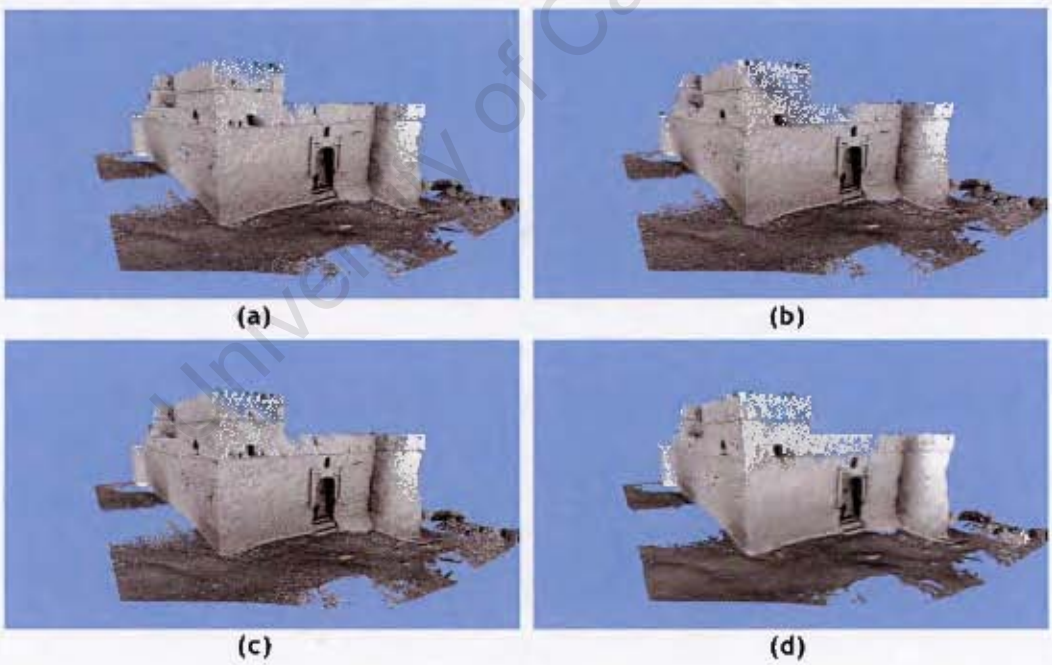


Figure 5.14: The results of the simplification process: (a) 33 million triangles, (b) 8 million triangles, (c) 3 million triangles, (d) 100,000 triangles.

addition, common points were identified on the SfM-point cloud and on the laser-scan model with the help of the open-source software CloudCompare[1](see fig5.15b+c).

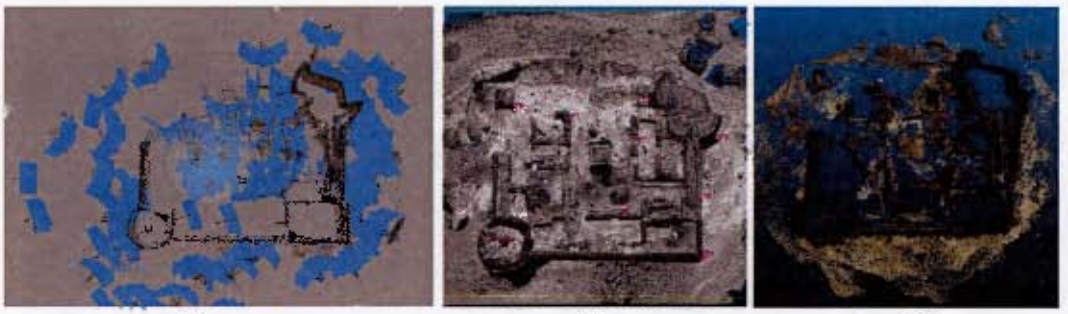


Figure 5.15: (a) The result of the alignment of the images in Agisoft Photoscan. (b) Twelve common points were selected on the PlyMC-model and the SfM point cloud(c)

Sfm2texture was used to perform a 3D transformation between the specified control points and to apply the established transformation to all external camera parameters of the "bundle.out"-file. Per camera, the tool creates a XML-file, which can be read by the software tools TexAlign, TextAlignSuite and TexTailor.

For demonstration purposes, a set of 38 images was selected, which together, cover a large part of the building. Each XML file was processed with TextAlignSuite to refine the alignment of the SfM process. The Geomagic-model of the Gereza, which carries intensity information, was employed to serve as a guide for the refinement process via the Mutual Information approach. Most images only needed two to three iterations to be very well aligned with the model (fig.5.16). In some more difficult cases, the shading of the model had to be alternated. Five shading-options were available: normal-mapping, normal-mapping plus ambient colour (in here intensity), ambient colour, specular-mapping, specular-mapping plus ambient colour. In two cases, none of these methods worked, which is why another approach, suggested before, was followed. First all other images were aligned and projected with TexTailor to the model. This model, with the new colour information attached to the vertices, could then serve as a guide for the alignment of the remaining two images (fig.5.17).

Once all images were aligned, they were projected again onto the PlyMC-model (8 million triangles) with TexTailor and the standard settings for out-of-core projection.

The output is a vertex-coloured, highly detailed mesh, which is presented in figure 5.18. The model still has many blank spots, however, the texturing was only meant as a demonstration of the process. In the same manner as described above, more and more

[1]http://www.danielgm.net/cc/


Figure 5.16: Alignment of the photograph to the model before (a) and after (b) the refinement with TextAlignSuite




Figure 5.17: (a) a set of re-aligned photographs was projected onto the model, which was then used as a guide for the fine-alignment of more difficult images. Image (b) shows the discrepancies between image and model before the refinement, while (c) shows the result.

images can be aligned, as required. It is expected that only a few images, which were not part of the SfM reconstruction process, need to be manually aligned with TexAlign via the selection of common points, prior to the refinement with TextAlignSuite.


Figure 5.18: The coloured model of the Gereza.

## 5.2 Experiences with Other Datasets

During the course of this research, the workflow was partially applied to sites of various other natures by the team of the Zamani Project with success. Songo Muara, one of the largest sites of the Zamani Project with about six billion points, was successfully modelled with the described registration and surface reconstruction methods. The texturing methods were employed on, among others, the South African rock art sites of Game Pass, Drakensberg and Diepkloof, Cederberg. Also various other projects served as testing environments for other parts of the workflow, such as DGB I+II (Cameroon), Petra (Jordan) and Namoratung'a (Tanzania).

## 5.3 Discussion of the Workflow

Reviewing the proposed workflow and the experience made on the example of the Gereza, in the following section, the individual parts of the developed workflow will be reviewed and discussed.

### 5.3.1 Registration

Registration is still a largely manual task, and thus, the required time depends to a large amount on rendering speed while moving and rotating the scans around. Scans in lower resolution would increase the rendering speed, but also provides fewer features for an accurate surface matching. Splitting the alignment process into a rough and a fine-alignment step, proved to be a very efficient solution for this problem, as experienced on the example of the Gereza. For the initial alignment, where manual intervention is large, the resolution of the scans was lowered, while for the fine-alignment, where the view of the scene is hardly changed, the high-resolution scans were loaded. As it is not possible to load all scans of very large datasets at the same time, the skeleton model approach turned out to be a very effective solution. Only the very minimum of scans needed to remain loaded during fine-registration, while all others were loaded, fine-aligned and un-loaded again from memory.

Global registration, as the automation of the fine-alignment process was often studied in literature. However, it was shown here, that these processes do not always improve the alignment. A disturbing clustering behaviour of the scans could be observed, which are linked by only little overlap. However, it is believed, that a skeleton model, which cannot be modified during the global registration, could also be a solution to this problem. The skeleton model provides enough overlap for the bulk of scans to be aligned properly and automatically. However, further tests are necessary to confirm this idea.

### 5.3.2 Cleaning

The cleaning, as experienced on the Gereza dataset, is the most time-consuming part of the pipeline and could largely be speed up by working on the entire point cloud, as opposed to single scans. As most software is not able to easily process such large datasets, a sub-division of the scans into blocks, similar to the meshing procedure, could solve this problem. For this purpose, the file converter ptx2ply was adapted to split the scans into blocks, without converting them into a different format. First brief experiments showed promising results.

It is not believed that the cleaning process can be automated entirely. However, the process could be drastically accelerated by providing intelligent classification and segmentation algorithms, which can remove entire bushes or other objects, by a simple click. As explained before, information such as intensity, colour (if available), point normals, depth variations between point neighbours could be used for segmentation of the data into individual objects.

### 5.3.3 Surface Reconstruction

The developed approach of using data-subdivision, the double meshing process with PlyMC and the creation of a Delaunay model in Geomagic worked without any complications on the example of the Gereza. With these methods, two key issues of the surface reconstruction process could be solved.

Firstly, the subdivision made the surface reconstruction independent of data size and independent of the employed surface reconstruction algorithm. The meshing experiment without subdivision, presented in section 5.1.4.4, confirmed the inefficient scan handling of PlyMC as described earlier in section 3.4.4.3. Thus, as it is expected that this behaviour becomes more and more problematic with even lager datasets, the only alternative to become independent of the input data volume is the use of sub-divisions. As an additional advantage, the first merging process with PlyMC could theoretically be swapped with any other meshing technique, even if it is not designed for out-of-core data processing. However, the sub-division requires introducing an overlap between the individual sub-blocks and thus a second merging step with PlyMC becomes necessary.

Secondly, the detailed output of PlyMC was combined with parts of the more complete Delaunay model to produce a highly decorated model with only a minimum amount of holes. This proved to be an effective work-around for the tessellation depth problem.

Despite the successful design of a flexible and scalable approach for the creation of a 3D model, the suggested approach is quite complicated and requires a considerable amount of experience to be able to estimate the appropriate subdivision settings and tessellation depth threshold. It is believed that the meshing workflow, presented here, could be simplified by using a surface reconstruction algorithm, based on Moving Least Squares, similar to the approach by Cuccuru et al. (2009), as described earlier. Results with another MLS software, OM, were reported in this research to combine detail and completeness. However, the software was found to be not reliable enough to be included in the current workflow, due to problems with vegetation.

### 5.3.4 Simplification

With the availability of larger RAM resources, simplification became a very straightforward technique. A mesh of over 32 million polygons could be simplified without any problems on a 24GB machine. However, for more demanding cases, another method was developed, which first simplifies strips of sub-blocks before all strips are decimated altogether.

As shown in the literature review of this research, the tendency is to implement the simplification as part of the viewing process, and thus with the development of new viewing tools, this step will be mostly excluded from the pipeline.

### 5.3.5 Hole Filling

In general, hole filling is a delicate matter in relation with cultural heritage. However, the proposed process of creating realistic and accurate patches with SfM-methods and their merging with PlyMC, as shown on the example of the Gereza, proved to be a fast and easy method.

### 5.3.6 Texturing

The texturing of 3D models with the help of SfM-methods, as proposed in literature (Liu et al., 2006), was found to be a very effective and fast solution to the image alignment problem. The developed software sfm2texture allows the incorporation of SfM into the workflow of aligning and projecting individual images with the VCG-labs software suite TexAlign, TextAlignSuite and TexTailor. Due to inaccuracies in the SfM-reconstruction-process, the images need to be re-fined with TextAlignSuite, which was shown to be a fast process. Also the intensity-coloured-model, created with Geomagic, was found to be of great value, resulting in faster convergence of the fine-alignment procedure, as it served as a guiding reference for the mutual information approach within TextAlignSuite. In parallel to this research, a similar approach was developed by Corsini et al.[2], which is believed to automate the SfM-texturing procedure to a large extend.

Taking images for the processing with SfM tools for all rooms of a larger building will be quite difficult, as this would require hundreds or even thousands of photographs. However, full dome panoramas, together with the developed software PanoAlign, could be a more effective approach to document rooms.

No solution could be found for automatic colour matching all images before they are projected onto the surface and further research is recommended.

comment

[2]M. Corsini, M. Dellepiane, F. Ganovelli, R. Gherardi, A. Fusiello, R. Scopigno - Private Communication with M. Dellepiane, November 2011

# Chapter 6

# Conclusions and Recommendations for Future Research



## 6.1 Conclusions

The primary objective of this research was to produce a pipeline, which is independent of the size and the nature of the dataset and also allows the creation of highly detailed 3D models. The main usage of this pipeline is meant to be within the Zamani project, a unique project, collecting data of various cultural heritage sites of different nature, which requires a flexible workflow, providing alternatives for each step of the pipeline, in case some methods fail or do not produce the required result. Besides the large variety of scanned objects, the increasing data volume per scanned site, which grew by two orders of magnitude during the past seven years, emerged as one of the key problems in laser-scanning.

Producers of laser-scanning hard- and software tend to create the impression, that producing an accurate, high resolution 3D model is an easy and almost automated task. This research showed however that spatial data processing for cultural heritage is not trivial at all and requires fundamental knowledge of the procedures to be able to set the adequate parameters. Thus, a large part of this research was dedicated to the collection and the discussion of the various available approaches to be able to design a generic workflow, producing highly detailed 3D models, as successfully demonstrated on the example of the Gereza, a dataset consisting of 720 million points.

As part of the primary objective, the increasing amount of points was also part of the research questions, asking how the pipeline can be made independent of the input data volume. This was mainly achieved by splitting the scan data automatically into manageable sub-blocks and by presenting routines for the efficient processing and

merging of these sub-blocks into a highly detailed 3D model. The subdivision process, which can be adjusted according to the specifications of the current computer system and the data volume of the current site, in combination with the use of out-of-core tools, enables the reconstruction of a surface, based on the highest available resolution. Subdivision however, can only optimize tasks, which are taking place after the registration of scans. To also allow the registration to become independent of the amount of recorded scans, it was suggested to produce a skeleton model of a few, very important scans, covering the largest part of the object's surface. As the registration is mainly a pairwise alignment process, which accumulates errors with every scan added to the group, the skeleton model can reduce the overall error, by serving as a reference for the alignment of the bulk of scans.

Another research question to be investigated was about the design of a workflow which is flexible enough to be applied to different objects of various natures. The answer to this question was found by analysing various methods and offering alternatives to be incorporated into the pipeline when necessary. While a specific workflow was proposed here, software alternatives were presented for most processing steps, which make the workflow independent of the nature of the scanned object. The user can thus adapt to specific requirements of a site, for example, exchanging the meshing algorithm, in case a watertight surface is desired, for example for landscapes or statues. Converters, such as sfm2Texture and ptx2ply, were created or modified, to ensure compatibility between the individual steps.

The third research question was about the optimization and automation of the workflow, to minimize the manual intervention. With regards to this question, ideas and methods for the optimization of most of the steps were presented, which decrease the manual processing time, such as the subdivision of scans for the cleaning of individual blocks rather than individual scans, and the use of a skeleton model during registration. In addition, an approach was proposed to register scans automatically with SfM-methods, as described in section 3.6.6, which seems to be worthwhile to be investigated further. Also, the texturing and hole filling steps were found to benefit greatly from the incorporation of Structure-from-Motion methods into the workflow.

With these and various other methods the optimisations of the individual steps of the pipeline, as well as the overall procedure was achieved and an efficient pipeline was developed. However, the process is still time consuming and requires high skill levels and there is room for further development in nearly all steps of the pipeline.

### 6.1.1 Limitations

A general limitation of this research was the limited space and time for the individual processing steps, due to their integration into a larger, very complex workflow. Ideas, which could have optimized some tasks even further, such as the registration of scans with SfM methods, could not be followed up. Each of the processing steps provides enough material to be a research topic on its own, and future work should rather focus on resolving individual issues.

Another limiting factor was the little availability of free processing tools. As the review of various approaches in literature showed, some very interesting ideas were proposed, but only very few tools are released to the public. However, as this pipeline was designed flexibly, it can be seen as a modular framework, of which its individual parts can be replaced by other, new processing tools, if they become available.

### 6.1.2 Recommendations for Future Research

Even though a pipeline was presented, which can be employed in a productive environment and all research questions were answered, there is still a lot of room for further development. In general, the key to further improvement and automation is believed to be in the combination of laser-scanning with other techniques, such as photogrammetry. SfM for example, as mentioned above, could be developed further to perform the registration of laser scans and the alignment of photographs to the laser scan data in one process.

It is not believed however, that the entire pipeline can be entirely automated to a "one-click" process, as there will always be unforeseen problems or situations where the algorithm cannot decide on how to proceed, such as during cleaning. Thus the key is aiming for semi-automation, assisting the user as much as possible.

Apart from the processing, the actual viewing and examination of these high resolution models is still quite difficult, as appropriate software-viewers are still rare and did not develop much beyond a proof of concept. Future work also needs to focus on the implementation of these highly detailed 3D models in the workflow of other professions such as architects, conservation and restoration experts. Only with the acceptance and understanding of the data by the users, laser-scanning of cultural heritage site can receive the attention it deserves.

### 6.1.3 Outlook

In future, terrestrial laser scanning is likely to move away from scanning from fixed locations to scanning from mobile platforms; an approach which is already used in the

industry for the surveys of roads, industrial plants and cityscapes. For inside real-time scanning, very promising results were reported for the Microsoft game-controller Kinect, which records real-time 3D data with a built-in depth sensor (Izadi et al., 2011). The authors presented an approach to dynamically create a 3D-model of a scene, while the controller is swept across the surface. The sensor was also attached to a small quad-copter drone which can autonomously explore and scan entire objects[1]. Some problems experienced within this research will thus be overcome, but almost certainly new issues will arise as well. Considering these developments, it is believed that in the near future, producing highly detailed 3D models will become a standard and affordable approach to the documentation of cultural heritage sites, a key aspect to our cultural identity.



[1] http://groups.csail.mit.edu/rrg/index.php?n=Main.VisualOdometryForGPS-DeniedFlight

# References


Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S. M., and Szeliski, R. (2011). Building rome in a day. *Commun. ACM*, 54:105–112.

Aiger, D., Mitra, N. J., and Cohen-Or, D. (2008). 4-points congruent sets for robust surface registration. *ACM Transactions on Graphics*, 27(3):#85, 1–10.

Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., and Silva, C. T. (2001). Point set surfaces. In *VIS '01: Proceedings of the conference on Visualization '01*, pages 21–28, Washington, DC, USA. IEEE Computer Society.

Allen, P. K., Troccoli, A., Smith, B., Murray, S., Stamos, I., and Leordeanu, M. (2003). New methods for digital modeling of historic sites. *IEEE Computer Graphics and Apllications*, 23(6):32–41.

Amenta, N., Choi, S., and Kolluri, R. K. (2001). The power crust. In *Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 249–260. ACM Press.

Amenta, N. and Kil, Y. J. (2004). Defining point-set surfaces. *ACM Trans. Graph.*, 23(3):264–270.

Barazzetti, L., Scaioni, M., and Remondino, F. (2010). Orientation and 3d modelling from markerless terrestrial images: combining accuracy with automation. *The Photogrammetric Record*, 25(132):356–381.

Bernadini, F. and Rushmeier, H. (2002). The 3d model acquisition pipeline. *Computer Graphics Forum*, Volume 21(2):149–172.

Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., and Taubin, G. (1999). The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359.

Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14:239–256.

Boehler, W. and Marbs, A. (2002). 3D scanning instruments. In *Proc. of the CIPA WG6 Int. Workshop on scanning for cultural heritage recording.*

Boehler, W., Vicent, M. B., and Marbs, A. (2003). Investigating laser scanner accuracy. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Antalya - 2003*, Vol. XXXIV, Part 5/C15:pp. 696–701.

Bolitho, M., Kazhdan, M., Burns, R., and Hoppe, H. (2007). Multilevel streaming for out-of-core surface reconstruction. In *SGP '07: Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 69–78, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.

Bolitho, M., Kazhdan, M., Burns, R., and Hoppe, H. (2009). Parallel poisson surface reconstruction. In *ISVC '09: Proceedings of the 5th International Symposium on Advances in Visual Computing*, pages 678–689, Berlin, Heidelberg. Springer-Verlag.

Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., and uno Levy, B. (2010). *Polygon Mesh Processing*. AK Peters.

Callieri, M., Cignoni, P., Corsini, M., and Scopigno, R. (2008). Masked photo blending: mapping dense photographic dataset on high-resolution 3d models. *Computer & Graphics*, 32(4):464–473. for the online version: http://dx.doi.org/10.1016/j.cag.2008.05.004.

Callieri, M., Cignoni, P., Dellepiane, M., and Scopigno, R. (2009). Pushing time-of-flight scanners to the limit. In *The 10th International Symposium on Virtual Reality, Archaeology and Cultural Heritage VAST (2009)*, pages 85–92. Eurographics.

Callieri, M., Cignoni, P., Ganovelli, F., Montani, C., Pingi, P., and Scopigno, R. (2003). Vclab's tools for 3d range data processing. *4th International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage (VAST2003) and First EUROGRAPHICS Workshop on Graphics and Cultural Heritage, Brighton (UK)*, 5-7.

Carr, J. C., Beatson, R. K., Cherrie, J. B., Mitchell, T. J., Fright, W. R., McCallum, B. C., and Evans, T. R. (2001). Reconstruction and representation of 3d objects with radial basis functions. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 67–76, New York, NY, USA. ACM.

Chen, Y. and Medioni, G. (1992). Object modelling by registration of multiple range images. *Image Vision Comput.*, 10(3):145–155.

Cignoni, P., Ganovelli, F., Gobbetti, E., Marton, F., Ponchio, F., and Scopigno, R. (2004). Adaptive tetrapuzzles: efficient out-of-core construction and visualization of gigantic multiresolution polygonal models. *ACM Trans. Graph.*, 23(3):796–803.

Cignoni, P., Ganovelli, F., Gobbetti, E., Marton, F., Ponchio, F., and Scopigno, R. (2005). Batched multi triangulation. In *Proceedings IEEE Visualization*, pages 207–214, Conference held in Minneapolis, MI, USA. IEEE Computer Society Press.

Cignoni, P., Montani, C., Rocchini, C., and Scopigno, R. (2003). External memory management and simplification of huge meshes. *IEEE Transactions on Visualization and Computer Graphics 9(4).*

Cignoni, P., Montani, C., and Scopigno, R. (1998). A comparison of mesh simplification algorithms. *Computers & Graphics, Pergamon Press*, Vol. 22(1):37–54.

Cohen, J., Varshney, A., Manocha, D., Turk, G., Weber, H., Agarwal, P., Brooks, F., and Wright, W. (1996). Simplification envelopes. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 119–128, New York, NY, USA. ACM.

Corsini, M., Dellepiane, M., Ponchio, F., and Scopigno, R. (2009). Image-to-geometry registration: a mutual information method exploiting illumination-related geometric properties. *Computer Graphics Forum*, 28(7):1755–1764.

Cuccuru, G., Gobbetti, E., Marton, F., Pajarola, R., and Pintus, R. (2009). Fast low-memory streaming mls reconstruction of point-sampled surfaces. In *GI '09: Proceedings of Graphics Interface 2009*, pages 15–22, Toronto, Ont., Canada, Canada. Canadian Information Processing Society.

Curless, B. and Levoy, M. (1996). A volumetric method for building complex models from range images. In *SIGGRAPH 96 Conference Proceedings ACM SIGGRAPH, Addison Wesley, August 1996*, pages 303–312.

Davis, J., Marschner, S. R., Garr, M., and Levoy, M. (2002). Filling holes in complex surfaces using volumetric diffusion. *First International Symposium on 3D Data Processing, Visualization, and Transmission Padua, Italy, June 19-21.*

Debevec, P., Tchou, C., Gardner, A., Hawkins, T., Wenger, A., Stumpfel, J., Jones, A., Poullis, C., Yun, N., Einarsson, P., Lundgren, T., Martinez, P., and Fajardo, M.

(2004). Estimating surface reflectance properties of a complex scene under captured natural illumination. *USC ICT Technical Report ICT-TR-06.2004.*

Dellepiane, M. (2009). *Uses of uncalibrated images to enrich 3D models information.* PhD thesis, University of Pisa.

Dellepiane, M., Benedetti, L., and Scopigno, R. (2010). Removing shadows for color projection using sun position estimation. In *The 10th International Symposium on VAST International Symposium on Virtual Reality, Archaeology and Cultural Heritage*, pages 55–62. Eurographics.

Dellepiane, M., Marroquim, R., Callieri, M., Cignoni, P., and Scopigno, R. (2011). Flow-based local optimization for image-to-geometry projection. *IEEE Transaction on Visualization and Computer Graphics*, Online first.

Dey, T. K., Giesen, J., and Hudson, J. (2001). Delaunay based shape reconstruction from large data. In *Proceedings of the IEEE 2001 symposium on parallel and large-data visualization and graphics*, PVG '01, pages 19–27, Piscataway, NJ, USA. IEEE Press.

Edelsbrunner, H. (2000). Triangulations and meshes in computational geometry. *Cambridge University Press.*

Edelsbrunner, H. and Mücke, E. P. (1992). Three-dimensional alpha shapes. In *VVS '92: Proceedings of the 1992 workshop on Volume visualization*, pages 75–82, New York, NY, USA. ACM.

Eisemann, M., De Decker, B., Magnor, M., Bekaert, P., de Aguiar, E., Ahmed, N., Theobalt, C., and Sellent, A. (2008). Floating textures. *Computer Graphics Forum (Proc. of Eurographics)*, 27(2):409–418. Received the Best Student Paper Award at Eurographics 2008.

Farenzena, M., Fusiello, A., and Gherardi, R. (2009). Structure-and-motion pipeline on a hierarchical cluster tree. In *3DIM09*, pages 1489–1496.

Fiorin, V., Cignoni, P., and Scopigno, R. (2007). Out-of-core mls reconstruction. In *Proceedings of the Ninth IASTED International Conference on Computer Graphics and Imaging*, CGIM '07, pages 27–34, Anaheim, CA, USA. ACTA Press.

Franken, T., Dellepiane, M., Ganovelli, F., Cignoni, P., Montani, C., and Scopigno, R. (2005). Minimizing user intervention in registering 2d images to 3d models. *The Visual Computer*, 21(8-10):619–628. Special Issues for Pacific Graphics 2005.

Furukawa, Y., Curless, B., Seitz, S. M., and Szeliski, R. (2010). Towards internet-scale multi-view stereo. In *CVPR*, pages 1434–1441.

Furukawa, Y. and Ponce, J. (2007). Accurate, dense, and robust multi-view stereopsis. In *CVPR*.

Gal, R., Wexler, Y., Ofek, E., Hoppe, H., and Cohen-Or, D. (2010). Seamless montage for texturing models. *Computer Graphics Forum*, 29(2):479–486.

Garland, M. and Heckbert, P. S. (1997). Surface simplification using quadric error metrics. *SIGGRAPH 97*.

Garland, M. and Heckbert, P. S. (1998). Simplifying surfaces with color and texture using quadric error metrics. *IEEE Visualization 98*.

Gelfand, N., Mitra, N. J., Guibas, L. J., and Pottmann, H. (2005). Robust global registration. In *SGP '05: Proceedings of the third Eurographics symposium on Geometry processing*, page 197, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.

Gherardi, R., Farenzena, M., and Fusiello, A. (2010). Improving the efficiency of hierarchical structure-and-motion. In *CVPR10*, pages 1594–1600.

Ghosh, S. (2005). *Fundamentals Of Computational Photogrammetry*. Concept Publishing Co.

Gruen, A. W. (1985). Adaptive least squares correlation: A powerful image matching technique. *South African Journal of Photogrammetry, Remote Sensing and Cartography*, 14:175–187.

Guennebaud, G. and Gross, M. (2007). Algebraic point set surfaces. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 23, New York, NY, USA. ACM.

Hoppe, H. (1996). Progressive meshes. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108, New York, NY, USA. ACM.

Hoppe, H. (1998). Smooth view-dependent level-of-detail contorl and its application to terrain rendering. *IEEE Visualization 98 Conf.*, pages 35–42.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W. (1993). Mesh optimization. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 19–26. ACM Press.

Hoppe, H., DeRose, T., Duchampt, T., McDonald, J., and Stuetzle, W. (1992). Surface reconstruction from unorganized points. *ACM SIGGRAPH*, pages 71–78.

Ikeuchi, K., Nakazawa, A., Hasegawa, K., and Ohishi, T. (2003). The great buddha project: Modeling cultural heritage for vr systems through observation. In *Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, ISMAR '03, pages 7–, Washington, DC, USA. IEEE Computer Society.

Isenburg, M. and Gumhold, S. (2003). Out-of-core compression for gigantic polygon meshes. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 935–942, New York, NY, USA. ACM.

Isenburg, M. and Lindstrom, P. (2005). Streaming meshes. In *Proceedings of Visualization'05*, pages 231–238.

Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., and Fitzgibbon, A. (2011). Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 559–568, New York, NY, USA. ACM.

Jamin, C., Gandoin, P.-M., and Akkouche, S. (2009). Chumi viewer: Compressive huge mesh interactive viewer. *Computers & Graphics*, 33(4):542 – 553.

Johnson, A. (1997). *Spin-Images: A Representation for 3-D Surface Matching.* PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Johnson, A. E. and Hebert, M. (1997). Surface registration by matching oriented points. In *NRC '97: Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, page 121, Washington, DC, USA. IEEE Computer Society.

Kazhdan, M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *SGP '06: Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 61–70, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.

Lensch, H. P. A., Heidrich, W., and Seidel, H.-P. (2000). Automated texture registration and stitching for real world models. *In Proceedings of Pacific Graphics '00*, pages 317–326.

Levin, D. (2003). Mesh-independent surface interpolation. In Brunnett, H. and Mueller, editors, *Geometric Modeling for Scientific Visualization*, pages 37–49. Springer-Verlag.

Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J., and Fulk, D. (2000). The digital michelangelo project: 3d scanning of large statues. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 131–144. ACM Press/Addison-Wesley Publishing Co.

Lévy, B., Petitjean, S., Ray, N., and Maillot, J. (2002). Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.*, 21:362–371.

Lindstrom, P. (2000). Out-of-core simplification of large polygonal models. *ACM SIGGRAPH 2000*, pages 259–262.

Liu, L., Stamos, I., Yu, G., Wolberg, G., and Zokai, S. (2006). Multiview geometry for texture mapping 2d images onto 3d range data. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, pages 2293–2300, Washington, DC, USA. IEEE Computer Society.

Lorensen, W. E. and Cline, H. E. (1987). Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60:91–110.

Marras, S., Ganovelli, F., Cignoni, P., Scateni, R., and Scopigno, R. (2010). Controlled and adaptive mesh zippering. In *GRAPP - International Conference in Computer Graphics Theory and Applications.*

Mikhail, E. M., Bethel, J. S., and McGlone, J. C. (2001). *Introduction to Modern Photogrammetry.* John Wiley & Sons, Inc., New York.

Newman, T. S. and Yi, H. (2006). A survey of the marching cubes algorithm. *Computers & Graphics*, 30(5):854–879.

Nishino, K. and Ikeuchi, K. (2002). Robust simultaneous registration of multiple range images. In *Proc. of Fifth Asian Conference on Computer Vision ACCV '02*, number pp454-461.

Palma, G., Corsini, M., Dellepiane, M., and Scopigno, R. (2010). Improving 2d-3d registration by mutual information using gradient maps. In *Eurographics Italian Chapter Conference 2010.*

Pulli, K. (1999). Multiview registration for large data sets. In *SECOND INTERNATIONAL CONFERENCE ON 3D DIGITAL IMAGING AND MODELING*, pages 160–168.

Pulli, K., Abi-Rached, H., Duchamp, T., Shapiro, L. G., and Stuetzle, W. (1998). Acquisition and visualization of colored 3d objects. In *Proceedings of the 14th International Conference on Pattern Recognition-Volume 1 - Volume 1*, ICPR '98, pages 11–, Washington, DC, USA. IEEE Computer Society.

Rossignac, J. and Borrel, P. (1993). Multi-resolution 3d approximations for rendering. In Falcidieno, B. and Kunii, T., editors, *Modeling in Computer Graphics*, pages 455–465. Springer-Verlag.

Rusinkiewicz, S. and Levoy, M. (2000). Qsplat: a multiresolution point rendering system for large meshes. *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 343–352.

Rusinkiewicz, S. and Levoy, M. (2001). Efficient variants of the icp algorithm. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pages 145–152, Quebec.

Rüther, H. (2002). An african heritage database, the virtual preservation of africa's past. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXIV.

Rüther, H., Chazan, M., Schroeder, R., Neeser, R., Held, C., Walker, S. J., Matmon, A., and Horwitz, L. K. (2009). Laser scanning for conservation and research of african cultural heritage sites: the case study of wonderwerk cave, south africa. *Journal of Archaeological Science*, 36(9):1847 – 1856.

Schroeder, W. J., Zarge, J. A., and Lorensen, W. E. (1992). Decimation of triangle meshes. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 65–70, New York, NY, USA. ACM.

Seitz, S. M., Curless, B., Diebel, J., Scharstein, D., and Szeliski, R. (2006). A comparison and evaluation of Multi-View stereo reconstruction algorithms. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:519–528.

Sharf, A., Blumenkrants, M., Shamir, A., and Cohen-Or, D. (2006). Snappaste: an interactive technique for easy mesh composition. *Vis. Comput.*, 22(9):835–844.

Snavely, N., Seitz, S. M., and Szeliski, R. (2006). Photo tourism: exploring photo collections in 3d. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pages 835–846, New York, NY, USA. ACM.

Sottile, M., Dellepiane, M., Cignoni, P., and Scopigno, R. (2010). Mutual correspondences: an hybrid method for image-to-geometry registration. In *Eurographics Italian Chapter Conference 2010*, pages 81–88. EG.

Thiel, K.-H. and Wehr, A. (2004). Performance capabilities of laser scanners - an overview and measurement principle analysis - performance capabilities of laser scanners - an overview and measurement principle analysis -. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXVI, Part 8/W2.

Turk, G. and Levoy, M. (1994). Zippered polygon meshes from range images. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 311–318. ACM Press.

Wu, C. (2011). Visualsfm: A visual structure from motion system. http://www.cs.washington.edu/homes/ccwu/vsfm/.

Zoller+Fröhlich-GmbH (2011). Imager 5010 accuracy. Marketing Material.

# Glossary

| | | page |
|---|---|---|
| 4PCS | 4 Point-Congruent Sets, automatic scan alignment method | 22 |
| Photoscan | Structure-from-motion software, commercial, `http://www.agisoft.ru` | 13 |
| Autodesk 123D Catch | Structure-from-motion software, commercial, `www.123dapp.com` | 13 |
| BRDF | Bidirectional reflectance distribution function | 50 |
| Bundler | Structure-from-motion software, open source, `http://phototour.cs.washington.edu/bundler/` | 13 |
| Cloud Compare | Point Cloud analysis tool, open source, `http://www.danielgm.net/cc/` | 68 |
| CMVS | Clustering Views for Multi-view Stereo, software to subdivide large image-datasets to be processed with PMVS by (Furukawa et al., 2010) | 15 |
| Cyclone | Industrial point cloud processing software, commercial, `http://www.leica-geosystems.com` | 54 |
| DLT | Direct Linear Transform | 13 |
| Geomagic | Industrial point cloud processing software, commercial, http://www.geomagic.com | 78 |
| GNSS | Global navigation Satellite System. General terms for systems such as GPS | |
| EXIF | Digital Images can accommodate also additional metadata in their header. EXIF is one of the available standards. | |

| | | |
|---|---|---|
| ICP | Iterative Closest Point Algorithm to align two objects (Besl and McKay, 1992; Chen and Medioni, 1992) | 17 |
| ImAlign | Software to align laser-scans, Part of Innovmetric Polyworks, `http://www.innovmetric.com` | 54 |
| ImEdit | Software to edit 3D models, Part of Innovmetric Polyworks, `http://www.innovmetric.com` | 54 |
| ImMerge | Software to reconstruct a surface from laser scans, Part of Innovmetric Polyworks, `http://www.innovmetric.com` | 54 |
| Intensity values | Reflectance strength of the signal which returns to the laser scanning instrument | |
| LAS | File-Format to store point data, acquired by laser-scanners | 55 |
| LOD | Level-of-Detail | 40 |
| Manifold | see definition in section 1.4.2 | 7 |
| Marching Cubes | Method to extract a triangulated surface from an implicit representation | 30 |
| Meshlab | Software to edit 3D models, incorporating a number of algorithms of various publications, open source, `http://meshlab.sf.net` | 54 |
| MLS | Moving Least Squares. Method to reconstruct a surface from a point cloud (Levin, 2003) | 33 |
| Mutual Information | Automatic method to align a photograph to a 3D model (Corsini et al., 2009) | 46 |
| My3DScanner | Structure-from-motion software, commercial, `http://www.my3dscanner.com` | 13 |
| nexus | Software developed by the VCG-lab, Pisa to display large triangulated meshes | 71 |
| OM | OctreeMerge, Software developed by the VCG-Lab, Pisa to reconstruct a surface, based on Moving Least Squares (MLS) | 33 |
| PanoAlign | Software developed by the author to assist in aligning cube-map panoramas | 61 |

| | | |
|---|---|---|
| Photosynth | Structure-from-motion software, commercial, `http://www.photosynth.net` | 13 |
| PLY | File format, developed by the Stanford University, to store polygonal 3D models. The format is often used for points only as well. | 55 |
| Plycut | Software by the author, which spatially divides a model in a PLY-file into subblocks or trims the model of excess surface outside a specified bounding-box | 95 |
| PlyMC | Software developed by the VCG-lab, Pisa to merge laser-scans. Based on the Volumetric Integration approach by (Curless and Levoy, 1996) | 31 |
| PMVS | Patch-based Multi-view Stereo Software for the reconstruction of dense point clouds from photographs (Furukawa and Ponce, 2007) | 15 |
| Poisson | Software to reconstruct a surface from point cloud data, open source, based on (Kazhdan et al., 2006) | 34 |
| PTX | File Format developed by Leica to store point cloud data | 55 |
| Ptx2Ply | Software to convert a point cloud (PTX file) into the various formats required by the surface reconstruction algorihtms. It was developed within the Zamani Project to convert PTX files into PLY, and was extended during this research to accomodate more output formats and more features, such as a spatial subdivision. Based on code from Scanalyze. | 55 |
| PTXtransform | Software to apply a 4x4 transformation matrix, specified in a XF-File to the points stored in a PTX-File | 108 |
| Scanalyze | Software developed by the Stanford University to align laser-scans with ICP, open source, `http://graphics.stanford.edu/software/scanalyze/` | |
| SIFT | Scale invariant Feature Transform, a image feature detection method (Lowe, 2004) | 13 |
| SfM/SaM | Structure-from-Motion / Structure-and-Motion | 13 |

| | | |
|---|---|---|
| Sfm2Texture | Software developed within this research to align camera positions, part of a SfM-Reconstruction with a 3D model, and to create a file per camera station which can be read by TextAlignSuite | 63 |
| SFM-toolkit | Structure-from-motion software, open source, `http://www.visual-experiments.com/demos/sfmtoolkit/` | 13 |
| Spin Images | Method to automatically align laser scans (Johnson and Hebert, 1997) | 20 |
| TDP | Tesselation Depth Problem | 73 |
| Tetrapuzzles | Adaptive Tetrapuzzles approach, developed to handle the visualization of very large triangulated meshes | 41 |
| TexAlign | Software developed by the VCG Lab to recover a camera position by selecting common points on photograph and 3D model, software available on request only | 58 |
| TexTailor | Software developed by the VCG Lab to project images onto a 3D model, software available on request only | 58 |
| TextAlignSuite | Software developed by the VCG Lab to recover a camera position by using a mutual information method, software available on request only | 58 |
| tricdecimator | Software developed by the VCG Lab to decimate 3D models, based on the Quadric Error Metric approach by (Garland and Heckbert, 1997) | 39 |
| VCG | Visual Computing Lab of the ISTI-CNR, Pisa, `http://vcg.isti.cnr.it/` | |
| VisualSFM | Structure-from-motion software, open source, `http://www.cs.washington.edu/homes/ccwu/vsfm/` | 13 |
| XF | Transformation File. Contains a 4x4 transformation matrix | |

| | | |
|---|---|---|
| Zamani | Swahili word for the past. Zamani Project is the short name for the "African Cultural Heritage Sites and Landscapes Project" `http://www.zamaniproject.org` | 2 |
| Zippering | Method to merge triangulated laser scans, based on (Turk and Levoy, 1994) | 29 |