FROM POINT SAMPLES TO SURFACES - ON MESHING AND ALTERNATIVES

Jan Böhm, Maria Pateraki

Institut für Photogrammetrie, Universität Stuttgart, Germany, jan.boehm@ifp.uni-stuttgart.de
Department of Cadastre, Photogrammetry and Cartography, The Aristotle University of Thessaloniki, Greece, mariapat@topo.auth.gr

KEY WORDS: Laser scanning, Point Cloud, Surface, Modelling, Visualization

ABSTRACT

Terrestrial laser scanners deliver a dense point-wise sampling of an object's surface. For many applications a surface-like reconstruction is required. The most typical example is the visualization of the scanned data. Traditional approaches use meshing algorithms to reconstruct and triangulate the surface represented by the points. Especially in cultural heritage, where complex objects with delicate structures are recorded in highly detailed scans, this process is not without problems. Often long and tedious manual clean-up procedures are required to achieve satisfactory results. After summarizing our experience with current meshing technology we therefore explore alternative approaches for surface reconstruction. An alternative approach presented within this paper is point splatting. We have developed an algorithm to compute a suitable surfel representation directly from the raw laser scanner data. This results in a speedy and fully automated procedure for surface reconstruction. The properties of the different approaches for surface reconstruction are discussed considering a practical example from the field of cultural heritage. The Panagia Kera in Kritsa near Agios Nikolaos on the island of Crete was chosen as a suitable example.

1 INTRODUCTION

Terrestrial laser scanning has become a popular tool for the documentation of architectural and cultural heritage sites. No other measurement system can parallel the speed, range and accuracy of three-dimensional data acquisition. When we refer to laser scanning we refer to time-of-flight measurement systems also known as terrestrial LIDAR. The scanner measures an object's surface by acquiring a dense point-wise sampling of the surface, typically by deflecting a laser-beam in two dimensions across the surface and measuring the distance along with it. This results in a three-dimensional point cloud representing the object. This raw product can either be used for basic measurements or as the source of many successive products, e.g. a three-dimensional mesh.

Automatic mesh generation is a wide subject stretched across many scientific fields including computer graphics, CAD, material engineering, geodesy mathematics and others (Owen, 1998). Meshing can be performed in arbitrary dimensions. The simplest case is two-dimensional meshing, where triangulation and quad mesh generation are common. These can be generalized to three-dimensional space, where the corresponding primitives are tetrahedrons and hexahedrons. A three-dimensional triangulation therefore is a tetrahedronalization. The meshing process which is usually desired in terrestrial laserscanning is surface meshing. Surface meshing is somewhere in-between two- and three-dimensional meshing. While the data, typically point samples, is in three-dimensions, we seek the meshing on a surface, a two-dimensional entity embedded in three-dimensional space, or in other words a function which maps from $\mathbb{R}^2$ to $\mathbb{R}^3$. In order to compute the surface meshing from a number of point samples we therefore have to solve two problems. The first problem is that we have to find the surface described by the points. The second problem is that we have to find a suitable meshing on that surface.

For the type of input data from range sensors we are dealing with, it is reasonable to restrict the topic to triangular meshes, which have become a standard product in laser scanning. Triangular meshes are nowadays also often referred to as models. While the term model used to be restricted to CAD like descriptions of objects this broadening of the term has become very common both in the commercial domain and in scientific literature. This makes sense especially in the domain of cultural heritage, where we deal with the delicate and complex surfaces and structures of ornaments and statues and other highly individual objects. These can not be represented with CAD primitives or library objects, a common approach in industrial applications. A triangular mesh is often the most adequate representation of cultural heritage objects and thus the term model is justified.

Within this paper we summarize in section 3 our experience with current state-of-the art meshing software for the generation of high quality surface meshes. We also want to point out some alternatives to these meshing technologies. In section 4 we introduce an alternative approach called point splatting, a speedy and easy method for surface reconstruction. In order to demonstrate the capabilities and limitations of all techniques we have chosen an exemplary dataset from the field of cultural heritage documentation. Details of the data acquisition and the object description are given in the following section.

2 THE TEST CASE OF "PANAGIA KERA"

A Byzantine Basilica on the island of Crete was chosen as a test case in our investigations. The triple-nave Byzantine church, known as Panagia Kera, is one of the best known Byzantine monuments on Crete, dates to approximately 13th century A.D. and is dedicated to the Assumption of the Virgin Mary, to Saint Anthony and Saint Anna. The importance of the monument is attributed to the frescoes of the interior that are characteristic examples of different painting styles which date to the end of the middle Byzantine period and the beginning of the late Byzantine period. The wall paintings have suffered serious damage, such as condensation of moisture on the frescoes' surfaces and salt-like efflorescence partly veiling the paintings, whereas in other areas the paint layers are either only flaking or sagging or lie as a decomposing substance on the wall. Other parts of the interior walls have been covered by a layer of whitewash. A view from the South of Panagia Kera is shown in Figure 1, whereas Figure 2 shows a part of the wall in the central nave, where the type and size of damages of the frescoes are evident.

During a fieldwork campaign in September 2005, high-resolution digital terrestrial images and laserscanner data of the interior and exterior of the Church were collected. The data was and will be further used for different research purposes and tasks, among others to assist in future restoration works undertaken by the 13th Ephorate of Byzantine Antiquities of the Hellenic Ministry of Culture.

Figure 1: Left: view on the church from the South with targets attached. Right: detail of the frescoes on the interior documenting the degree of damage.

The laser scans were acquired with a Leica HDS 3000. The scanner records range data with a single point accuracy under 6 mm together with intensity and optionally collects RGB data from a built-in camera. However we find the quality of the RGB data insufficient even for texturing and therefore acquire color images separately with a high resolution digital SLR camera, a NIKON D2X. The scanner is able to acquire data in a horizontal field-of-view of 360° and 270° vertically. For the data acquisition on the outside of the church the field-of-view rarely exceeds 40°, but on the inside we could take full advantage of the field-of-view offered by the scanner. For georeferencing control points were measured with a total station, including targets and natural points.

Eight stations were used to acquire data of the exterior of the church within one day. Two additional scans were made from an elevated station to cover parts of the roof. The interior of the church was acquired from seven stations again within a day. The stations and sampling distance were carefully planned beforehand to have a constant sampling of the surface no less than 20 mm. This could be achieved for most parts of the church, except for some small regions were occlusions and other constraints prevented scanning at high density or made it impossible at all. The data of the exterior was registered using 30 paper targets placed on the church's facade and four retro-reflective targets placed on tripods. The computations were performed with the Cyclone software. The data of the interior was purely registered using ICP registration, which was performed using the Cyclone software and checked against Polyworks.

Following the registration of the overlapping partial point clouds in Cyclone, points that were visually identified as blunders and other points in the surrounding area of the object of interest that were captured by the laser scanner (e.g. points captured on trees, moving objects and persons) were manually removed. This is a crucial step before any further processing is done (Weyrich et al., 2004). Figure 2 shows the extent of manual point removal on the dataset for the exterior of the church. The original point cloud contained 2.8 million points. After editing the cloud only contains 1.5 million points. Nearly half of the points needed to be removed. For the interior less editing was necessary and the point cloud remained nearly unchanged at 2.2 million points. The registered point cloud, containing the 3D coordinates, intensity and color of the scanner were then exported from Cyclone to an appropriate format for subsequent meshing.

3 MESHING - CURRENT STATE-OF-THE-ART

In order to derive a triangle mesh from point samples two general strategies can be identified. The first strategy is to compute the triangle mesh directly from the full set of unorganized points using
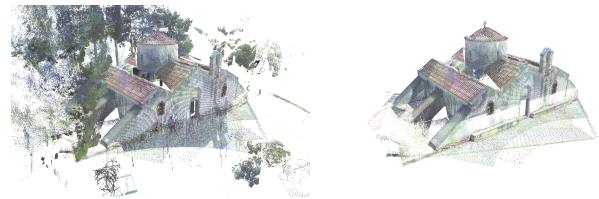

Figure 2: After the data from several stations is registered, outliers have to be removed manually.

only the three-dimensional coordinates of the points as input. The second strategy is to exploit the properties of the scanning device used, which can give hints to the typology of a single scan. An example for this is the connectivity within a range image. Using this information per scan one then has to integrate the different scans into one common mesh. From the many commercial applications available today for surface meshing Geomagic Wrap by Raindrop Geomagic (Edelsbrunner, 2003) is a good example for the first strategy, while Polyworks by Inovmetric (Soucy, 1997) is an example for the second strategy. Our interest is not in comparing or even benchmarking commercial products. Rather we want to share our experience with two popular products, which can be seen as exemplary for state-of-the-art meshing approaches and draw some general conclusions.

3.1 Geomagic

While registration of partially overlapping scans is also supported in Geomagic, we restrict our description to the meshing process. As indicated above Geomagic processes an unordered point cloud without prior knowledge of topology. It provides a large set of interactive modeling tools to process the data both before and after meshing. The user is able to perform the meshing of the imported point cloud, fill holes and use different features (planes, cylinders, lines) to improve the surface description. These tasks require manual interaction and are quite time consuming. The meshes are shown in figure 3. As the dataset becomes large the interactive handling of the data becomes difficult. After automatic generation of the mesh from the imported point cloud we have observed that several gaps existed in the mesh even though the number of points in those areas is sufficient to describe the surface. This situation can be seen in figure 4.

As could be expected from an approach that attempts to find topology without an initial approximation, Geomagic requires a very dense and almost uniform sampling in the point cloud in order to generate correct meshes. Otherwise gaps will appear in areas of lesser density. Initially hole filling, uniform sampling and refinement of single points (i.e. noise removal, fit points to planar surfaces, adjustment of boundaries in the windows and along arcs of the interior surfaces, etc.), were not performed prior to mesh
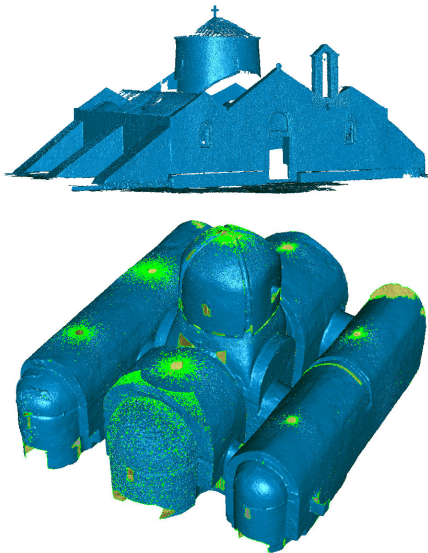
Figure 3: The full dataset of the exterior and the interior of the church meshed with Geomagic. In the bottom figure the holes in the surface are indicated with the green boundaries. Note the large number of extremely small holes.
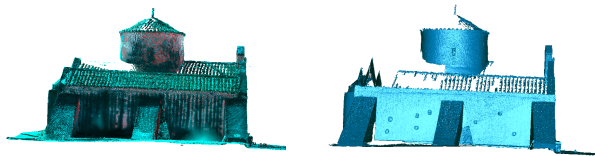

Figure 4: Problems encountered during meshing with Geomagic. Left: the point cloud captured with the laserscanner with color values. Right: the mesh derived with Geomagic. Although there are enough points measured on the roof, the meshing is incomplete over large areas.

generation. However, these steps are crucial for a more complete surface description, especially filling of holes and uniform sampling on point level. It is important to note that hole filling depends upon the selected points in the neighborhood and in some areas points have to be added manually before additional points with a uniform sampling are automatically interpolated. In order to generate an optimal mesh in Geomagic for our test case the following scheme has been followed. The point clouds from each nave have been processed individually, points were added manually in locally planar areas and a uniform sampling of points has been computed. Blunder detection and refinement of boundaries has been further applied. After these steps the mesh is created and hole filling (manual and automatic for small holes) is used.

3.2 Polyworks

While Polyworks, as well as Geomagic, is a commercial product and thus the exact algorithms involved are unknown, we can make some assumptions based on the experience as a user of the software and from early publications of the company's co-founder (Soucy and Laurendeau, 1995). As indicated above Polyworks creates a triangular mesh of the object by integrating a set of aligned 3D images. A 3D image is a $2\frac{1}{2}$-dimensional representation of the input data. A planar equidistant grid is used as a base parametrization, where each grid cell holds an associated height value. This structure is very familiar to geodesists from digital elevation models. With respect to the input data this means, that all input data has to be converted to a set of 3D images. Within Polyworks this can be done in three ways: 1) the scanning device naturally delivers point data based on a planar grid and thus

the data only needs to be re-interpolated to a regular grid, which is the case for most stripe projection systems; 2) the scanning device delivers data in another $2\frac{1}{2}$-dimensional representation, which then needs to be converted, which is the case for polar scanners; 3) the user manually subdivides the input data into a set of 3D images, for example, when the data is given in separate triangle meshes.

As do most terrestrial laserscanners, the HDS 3000 captures the data by oscillating a mirror and rotating the scanner head. This polar measurement principle creates polar coordinates. However the data is stored and exported in more intuitive Cartesian coordinates which are arranged in a two-dimensional grid of spherical topology. When importing the data it has to be converted to a grid of planar topology, as mentioned above. This obviously is not possible for the complete dataset of a $360°$ by $270°$ scan. Therefore the scan has to be subdivided and be partially projected to a planar grid. To do so without any loss of data, the sphere had to be subdivided to an infinite number of planar patches. This is computationally impossible. The user has to set the number of subdivisions by choosing an angle. To keep the connectivity within the scan, a sufficient overlap between neighboring patches from one scan is introduced. Unfortunately this creates some redundancy within the data and puts a further burden on the memory use.

As with the subdivision of each scan, the user has to choose some additional parameters upon data import. An important parameter is the sampling distance for the re-interpolation. The value should correspond to the sampling distance that was chosen for scanning the object. For a well planned and documented project the appropriate value can be chosen from the field book, but it can also be chosen interactively. In our case we targeted at a sampling distance better than 20 mm for the scanning, so half of that value seemed to be a good choice for re-interpolation. For the integration step parameters for smoothing and reduction need to be chosen, all of which can be directly derived either from the project's or the scanner's characteristics. The import of the data can be a lengthy process, when parameters have to be adjusted frequently. The integration step takes about 2-3 hours on a standard PC. Over all the meshing process including data preparation and output conversion requires a full working day. The result of the automated processing is shown in figure 5. In the case of the interior of the church the input data consisted of just over 2 million points. The resulting mesh consists of almost 7 million triangles. While in both cases the results from automated processing show smooth and consistent surfaces, we can also identify some typical artifacts.

While for the case of the exterior most of the gaps in the mesh are due to insufficient object coverage by the scans, the problems related to meshing itself become more apparent in the interior dataset. One of the most frequent artifacts appears as seam lines. They are due to a lack of information exchange from the scanning process to the meshing software. During scanning from a single station the scan is often subdivided into partial scans. This can be caused by the scanning hardware, which tries to keep the number of points for each scan below a certain threshold, or because it separates top and bottom view. Or it can be caused by the operator, who decides to split scans to adjust the sampling distance in order to keep a constant sampling for example across a long stretched wall. The adjacency of such neighboring scans is not explicitly stored and thus can not be accounted for while meshing. The situation is illustrated in the bottom of figure 6. Another problem is caused by the capability of the scanner to cover the full inside surface of a cylinder in a single scan of $360°$. The fact that the first and the last column of such a scan are adjacent is not adequately represented and thus is also not available while meshing. The situation is illustrated in the top of figure 6. Another typical problem is caused by the re-interpolation to a planar grid.
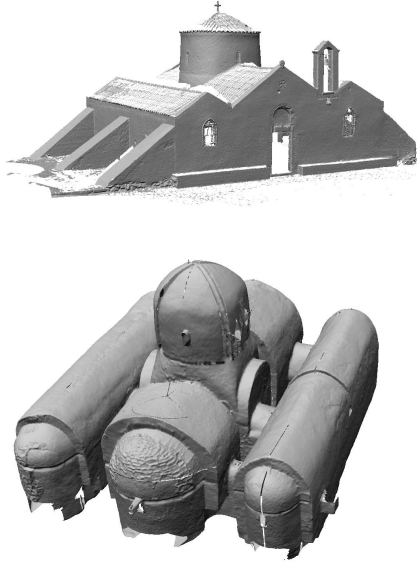
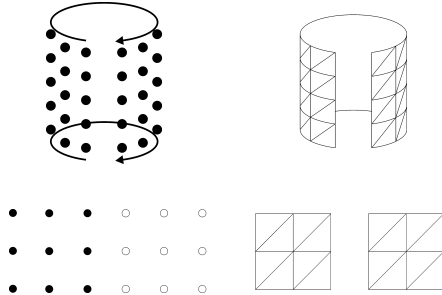Figure 5: The full dataset of the exterior and the interior of the church meshed with Polyworks.

Figure 6: Two cases where adjacency information is not correctly transferred from the scanning system to the 3rd party software. Top: the case of a 360° cylindrical scan. Bottom: the case of two adjacent but separate scans, which are distinguished by filled and non-filled dots.

As mentioned above during this process data points might be lost. This situation is illustrated in figure 7. The consequence of these situations on real data is shown in figure 8. It shall be noted again that the results presented in figure 5 and 8 were obtained with minimal interactivity, i.e. selection of software parameters. Obviously the problems discussed above can be corrected manually by inserting triangles using the interactive mesh refinement tools, which Polyworks provides as well.

4 ALTERNATIVE APPROACHES

Surface meshing currently is the de-facto standard for surface reconstruction from laser data. The available implementations are based on long years of experience and include a variety of optimizations and heuristics. However as we have demonstrated above the process has some inherent problems. We therefore want to investigate some alternatives to surface meshing. An alternative approach to derive surfaces from point data in the case of architectural models based on displacement mapping was previously published by Böhm (2005). Here we present a newly developed method derived from point-based computer graphics.

4.1 Point Splatting

In their survey on point-based techniques in computer graphics Kobbelt and Botsch (2004) raise the argument that while trian-
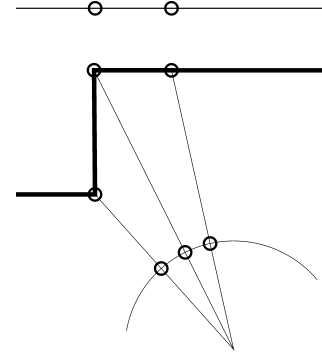

Figure 7: A piecewise planar projection of a polar scan can create problems when two separate points fall into the same grid raster.
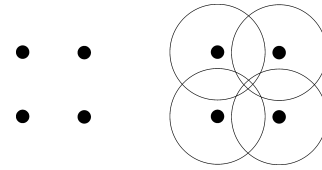

Figure 9: Individual points are converted to disks, large enough to cover the space in-between points.

gular meshes have proved to be a flexible and effective surface representation they might become inefficient when the number of vertices becomes very large. When the number of triangles exceeds the number of pixels on the screen, most triangles cover less than a pixel and the rasterization of the triangles becomes extremely expensive. We can easily follow this argument when we observe that currently a typical computer screen has just over a million pixels and even a simple dataset as the one used in our experiment exceeds this number. Laser-scanning projects of over 100 million points are not unusual.

To overcome this limitation of triangular meshes point-based methods have been proposed, which represent the surface by a point-wise sampling, where each point also stores the normal vector of the surface at this point. Point-based geometry provides all typical processing needs, such as editing, filtering and texturing. It becomes evident that this form of representation is ideally suited for datasets which were acquired with a sensor in a point-wise fashion such as terrestrial laserscanners. For the rendering of a point-based representation Pfister et al. (2000) proposed surfels as rendering primitives also referred to as point splats. Each point is associated with a disk perpendicular to the normal vector and with a radius just large enough to cover the space to the neighboring points. This idea is illustrated in figure 9.

A point-based representation does not explicitly store the neighborhood relation of the points, but attempts to dynamically compute the neighborhood typically using a k-nearest neighbor algorithm. This neighborhood relation is necessary for example to compute the surface normal in case it is not given and to compute the radius of each disk. In the context of surface meshing Soucy and Laurendeau (1995) have already noted that k-nearest neighbor algorithms tend to fail in unstructured point-clouds when the features become very small or parts of the object become very thin. We therefore propose a new method to use the topology given by the physical scanning process as the initial neighborhood information to compute both the normal vector and the radius for each point splat.

4.2 Direct Computation of Point Splats from Laserscans

For the remainder of the paper we assume that the topology of the measured points is given in form of a matrix. This is a logical assumption considering the column and row-wise fashion in which

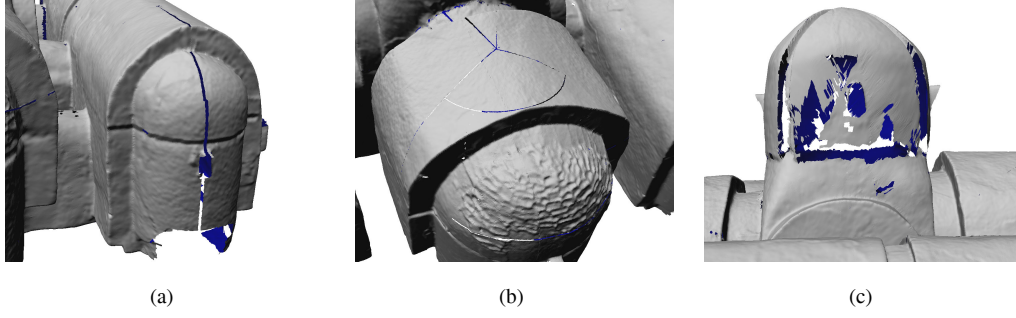|		
(a)	(b)	(c)

Figure 8: Typical problems encountered during meshing with Polyworks corresponding to the schematic situations introduced above. (a) A gap in-between the beginning and the end of a $360°$ scan. (b) Multiple gaps in-between separate scans from the same station. (c) Missing surface parts due to low angel of observation.

most laserscanners acquire the data. Since we use the scanning topology we do not have to dynamically compute neighborhood information and thus we do not need to build expensive search structures such as a kd-tree. All operations become linear with respect to the number of input pixels. As we only use a small number of points for each operation the method can also be implemented very memory efficient.

In order to compute the normal vector $\mathbf{n_i}$ for each point $\mathbf{p_i}$ we can use the direction of the laser beam from the center $\mathbf{c}$ of the scanner to the point $\mathbf{p_i}$ as a first approximation. This relation is expressed as $\mathbf{n_i} = \|\mathbf{p_i} - \mathbf{c}\|$. While this approximation is very coarse it is always guaranteed to give the correct orientation of the surface and thus there is no problem with flipped normal vectors. We also use this approximation method as a fall-back strategy in case other methods fail. A better approximation of the normal vector can be computed from the 4-connected neighbors $\mathbf{q_j}$. The normal vector is computed as the normalized sum of the cross product of the vector-pairs to its neighbors written as

$$\mathbf{n_i} = \left\| \sum_{\mathbf{j,k}} ((\mathbf{q_j} - \mathbf{p_i}) \times (\mathbf{q_k} - \mathbf{p_i})) \right\|$$

This is equivalent to the computation of the normal vector of a point in a triangular mesh as the area-weighted average of the normals of the triangles adjacent to the point often found in meshing algorithms. If we wish to further extend the neighborhood considered for the normal computation we can use a square mask centered at $\mathbf{p_i}$ and estimate a plane from all points inside the mask. This is the most costly approximation method. The latter two methods can fail if not enough neighbors to point $\mathbf{p_i}$ are available because there are too much invalid measurements surrounding the point. The results are improved at jump edges when a clipping threshold is used to suppress neighbors $\mathbf{q_j}$ which are much nearer or much further away to the center $\mathbf{c}$ compared to point $\mathbf{p_i}$.

In order to compute the radius of the splat at $\mathbf{p_i}$ we have implemented three different methods. The simplest method converts the angle in-between the directions $\mathbf{d_i}$ and $\mathbf{d_j}$ of two adjacent measurements to a distance at a virtual plane through the point inspected orthogonal to the direction of observation. This is computed as $r = \|\mathbf{d_i}\| \tan(\arccos(\mathbf{d_i} \cdot \mathbf{d_j}))$, where $\cdot$ denotes the dot product of two vectors. The second method corrects this radius by a factor accounting for the slope of the surface leading to $r^{'} = r/(\mathbf{d_i} \cdot \mathbf{d_j})$. The third method simply uses the true distance in-between the point $\mathbf{p_i}$ and the neighbor $\mathbf{q_j}$ which can be written as $r = \|\mathbf{q_j} - \mathbf{p_i}\|$. As discussed above any point $\mathbf{p_i}$ can have several neighbors $\mathbf{q_j}$. If we use the 4-connected neighbors this gives four radii. We use the maximum of the four values as
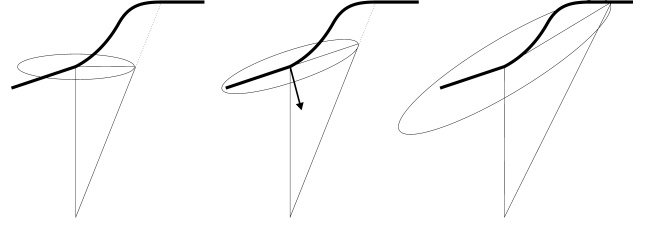

Figure 10: Three strategies to determine the radius of a splat from the measured distance and the angle of two adjacent beams. The object's surface is depicted in a bold line and the laser beams in thin lines.

the final result for each method. The three methods are illustrated in figure 10. The difference of orthogonal versus true radius is shown on a detail of the dataset in figure 11.

When the example dataset of the complete interior of the church is processed the total time of execution is under four minutes on a standard PC. This includes the reading and writing of the data with over two million points. The pure time for processing our algorithm is about 100 seconds. This results in a processing capacity of 20,000 points per second. Due to the linear nature of the algorithm we can use this number to predict the behavior on larger datasets. For the visualization of point splats several options exists. Typically they are freely available software packages developed at University institutions. PointShop3D is a rendering and editing software package developed at the Computer Graphics Lab at ETH Zurich (Zwicker et al., 2002). It provides several tools for further refinement of the models. QSplat is a viewer for large collections of point splats (Rusinkiewicz and Levoy, 2000). It uses a multiresolution hierarchy for progressive level of detail and is ideally suited for displaying large datasets at high frame rates. Figure 12 shows screen captures from both software packages of the data produced by our algorithm.

It is interesting to note that the results derived from point splatting do not suffer from the problems we found with traditional meshing. Neither non-uniform sampling nor missing adjacency information caused artifacts. Especially regions of low angle of observation were reconstructed much better. A clear disadvantage of the proposed method, which treats each scan separately, is that it can not make use of or combine information from separate scans. When two scans cover the same part of a surface each with a low resolution the point splats become large. Considering both point samples together the resolution is higher and thus the disks could be smaller and the normal estimates would adapt better to local features. This lack of combined information causes areas of rough surfaces and some loss of detail.

Figure 12: The full dataset of the interior converted to point splats and visualized using QSplat (left) and using PointShop3D in high-quality software-rendering mode (right) .
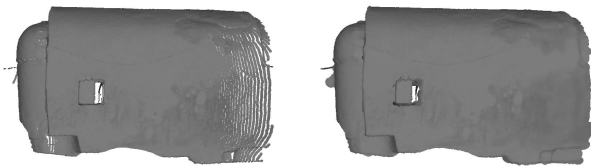

Figure 11: Difference of the first (left) and the third method (right) of splat radius computation shown on a detail of the dataset. By choosing larger splat sizes areas to the right with lesser densitycan still be adequately represented.

5   SUMMARY

Sophisticated software packages exist for the reconstruction of surfaces from dense point samples. However these software packages seem not to be tailored specifically to point samples acquired by terrestrial laser scanning. The inhomogeneous sampling often encountered in scan data and the polar nature of the instruments both can cause problems in the meshing process. With moderate manual interaction smooth and mostly consistent surfaces can be created, but some degree of error and artifacts have to be accepted. Only with intense manual labor can these meshes be transformed to high-quality models.

We have demonstrated an alternative approach to surface reconstruction based on point splatting. Our approach mainly targets visualization, but point-based computer graphics is a much broader field which offers many possibilities for post processing. The results obtained from our approach can not provide the same level of quality in the reconstructed surfaces with respect to smoothness and detail. However, the results are obtained almost instantaneous and without manual interaction. The results are ideally suited for quick visualization in the field, e.g. to decide on further data acquisition. The visualization can be extremely efficient even on simpler hardware when optimized rendering tools such as QSplat are used. The results obtained can also serve as a starting point for manual refinement, e.g. with the tools offered in Pointshop3D, which leads to results comparable to those obtained from traditional meshing. In order to enable the readers to reproduce our results and to experiment in the field of point splatting, we provide a download of our software and test data at http://www.ifp.uni-stuttgart.de.

5.1   Acknowledgments

References

Böhm, J., 2005. Terrestrial laser scanning - a supplementary approach for 3d documentation and animation. In: Fritsch (ed.), Photogrammetric Week '05, Wichmann, Heidelberg, pp. 263–271.

Edelsbrunner, H., 2003. Surface reconstruction by wrapping finite point sets in space. In: J. P. B. Aronov, S. Basu and M. Sharir (eds), Discrete and Computational Geometry - The Goodman-Pollack Festschrift, Springer-Verlag, pp. 379–404.

Kobbelt, L. and Botsch, M., 2004. A survey of point-based techniques in computer graphics. Computers and Graphics 28(6), pp. 801–814.

Owen, S. J., 1998. A survey of unstructured mesh generation technology. In: Proceedings 7th International Meshing Roundtable, Dearborn, MI.

Pfister, H., Zwicker, M., van Baar, J. and Gross, M., 2000. Surfels: Surface elements as rendering primitives. In: K. Akeley (ed.), Siggraph 2000, Computer Graphics Proceedings, ACM Press / ACM SIGGRAPH / Addison Wesley Longman, pp. 335–342.

Rusinkiewicz, S. and Levoy, M., 2000. QSplat: A multiresolution point rendering system for large meshes. In: K. Akeley (ed.), Siggraph 2000, Computer Graphics Proceedings, pp. 343–352.

Soucy, M., 1997. Inovmetric's multiresolution modeling algorithms. In: SIGGRAPH '97 Course Notes Multiresolution Surface Modeling, Los Angeles.

Soucy, M. and Laurendeau, D., 1995. A general surface approach to the integration of a set of range views. IEEE Transactions on Pattern Analysis and Machine Intelligence 17(4), pp. 344–358.

Weyrich, T., Pauly, M., Keiser, R., Heinzle, S., Scandella, S. and Gross, M., 2004. Post-processing of scanned 3d surface data. In: S. R. M. Alexa (ed.), Eurographics Symposium on Point-Based Graphics.

Zwicker, M., Pauly, M., Knoll, O. and Gross, M., 2002. Pointshop 3d: An interactive system for point-based surface editing. In: SIGGRAPH '02, pp. 322–329.