

On-line Reconstruction of CAD Geometry

Klaus Denker, Daniel Hagel, Jakob Raible, Georg Umlauf

Computer Graphics Lab

Dept. of Computer Science, HTWG Konstanz

78462 Konstanz, Germany

kdenker — hageldan — jraible — umlauf@htwg-konstanz.de

Bernd Hamann

Institute for Data Analysis and Visualization

Dept. of Computer Science, University of California

Davis, CA 95616, U.S.A.

hamann@cs.ucdavis.edu

Abstract—In reverse engineering and computer-aided design (CAD) applications point cloud data is usually manually scanned, reconstructed, and post-processed in separated steps. When point cloud data resulting from a scanning process do not satisfy certain necessary reconstruction requirements, one must perform scanning again to enable proper reconstruction. On-line reconstruction of 3d geometry allows one to generate and update a CAD reconstruction on-line during the scanning process with an hand-held laser scanner. Thus, regions where the scanned data is insufficient for the reconstruction are detected on the fly to allow an immediate correction and improvement of the scanned data. This enables the operator to focus on critical regions in the scanned data to improve the reconstruction quality.

We present an on-line segmentation and on-line reconstruction of basic geometric primitives. The presented methods allow for a real-time processing of a point stream. They utilize data structures that can be updated at any time when additional data from the stream has to be processed. This data is used to complete and improve the segmentation and reconstruction during the scanning process.

Keywords—on-line reconstruction; hand-held laser scanner; computer-aided design

I. INTRODUCTION

Hand-held laser scanners are used in reverse engineering to sample the geometry of physical objects. Commercial software in this area only allows a point based preview of the scanned data. The geometry is reconstructed in a separate, post-processing step. Parameters of the geometry are manually extracted from the point or mesh data. If there is data missing for a successful reconstruction, the whole process will fail. The complete scanning and reconstruction process has to be repeated.

With an on-line reconstruction algorithm, the reconstruction is computed concurrently with the acquisition of the point data during the scanning process. Regions where data is missing for the reconstruction can be detected on the fly. The human operator of the laser scanner can immediately re-scan this region to improve the reconstruction. Additional data is added to all existing data structures to improve reconstruction constantly.

In this paper the focus is on on-line segmentation and on-line reconstruction of basic geometric primitives. The current on-line reconstruction process is fully automated

without user interaction except for the handling of the hand-held laser scanner. Specifically, the contribution is the development and implementation of a geometry reconstruction method that automatically generates a surface representation in real time from a stream of scanned point data.

Figure 1 shows an overview of the different steps of the reconstruction process: the ball tree in Section III, the segmentation in Section IV, the boundary detection in Section V, and the primitive reconstruction in Section VI.

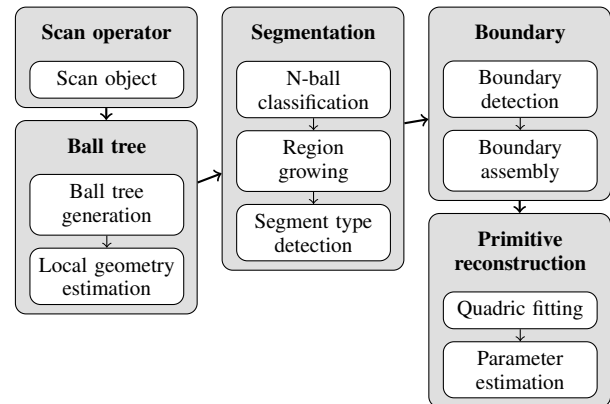


Figure 1: Overview of the on-line reconstruction process.

II. RELATED WORK

Geometric reconstruction from laser scan data is an active field of research since the development of triangulation laser scanners. One of the first approaches was developed in [1], where generalized cylinders are fitted to a point cloud from a laser scanner.

The reconstruction of the geometry of complex objects requires a segmentation of the surface. In [2] range and intensity images are segmented using a classification of surface types. Polynomials of variable degree are fitted to the segments and region growing is used to extend the segments. An application of this method to photogrammetry is presented in [3]. Here, 3d shapes of buildings are extracted from multiple types of aerial images.

For segmenting surfaces a large variety of mesh segmentation algorithms is used. The survey [4] provides an extensive

overview. For CAD reconstruction a segmentation by surface type is feasible. In [5] clustering based on the Gaussian sphere is used to filter surfaces by dimensionality. Most segmentation algorithms use meshes, because they allow a fast local analysis of the surface. An approach for unmeshed point sampled surfaces is presented in [6]. A weighted k -nearest neighbor graph is used to connect and analyze the point data.

An overview of early reconstruction methods for CAD geometry is presented in [7]. The segmentation and fitting of simple surfaces and free form geometry and the creation of polygonal boundary models are discussed. The reconstruction of rotational and translational surfaces and blends is presented in [8]. Even more flexible variational surfaces are used in [9] to segment a mesh. These segments are then reconstructed with geometric primitives using an implicit representation.

Fast stochastic methods like Random Sample Consensus (RANSAC) fitting can be used for iterative reconstruction and approximation with geometric primitives [10] or super-quadratics [11].

Reconstruction of quadrics is often used, because they allow the representation of most geometric primitives. A detailed survey of quadric reconstruction methods from triangle meshes is provided by [12]. In [13] an iterative segmentation and quadric fitting on an unorganized point cloud is presented. The fitted quadrics are used to iteratively improve the segmentation.

Stream processing of large data sets of points is done by [14], [15]. A set of geometric operators is processed sequentially on a large point set using a sweep line approach. Thus the points need to be sorted along one spacial axis.

Hand-held laser scanner devices produce unsorted streams of point data. An on-line triangulation of such data streams is done at [16]. After reducing the data to a set of equally distributed vertices, a surface mesh is generated on-line. An extension of this approach to a multi-level data structure is provided by [17], [18]. This allows to adapt the local density of the mesh to the density of the scanned points in the same area.

III. BALL TREE DATA STRUCTURE

The basic data structure for the on-line reconstruction is a ball tree as described in [18]. This data structure stores and processes the data stream from the laser scanner and prepares it for all subsequent reconstruction steps.

A. Ball tree generation

The data stream from the laser scanner consists of 3d raw points and orientation data of the laser probe. The raw points are assigned to so-called *neighborhood balls* or *n-balls* β which are defined as

$$\beta(c, r) = \{x \in \mathbb{R}^3 : \|x - c\| < r\}$$

with center c and radius r . These n-balls might intersect. As the raw points are streamed in, each raw point p is associated to the n-ball $\beta(c, r)$ that contains p with minimal distance $\|p - c\|$. If no such ball exists a new n-ball $\beta(p, r)$ is generated with center p and maximal radius r such that $\beta(p, r)$ does not contain any other n-ball center. For details see [18].

N-balls are organized in an octree data structure, the so-called *ball tree*. An n-ball is associated to the octree region that contains its center. The edge length of this region equals the radius of its associated n-balls. So, all n-ball radii are dyadic fractions of the edge length e_O of the initial scan area, i.e., $r = e_O/2^n$ for $n \in \mathbb{N}$.

B. Local geometry estimation

For each n-ball β estimates for the local normal and principal curvatures are computed. The normal n_β is computed by principal component analysis (PCA) of the raw points of a neighborhood of n-balls in the ball tree. To compute the principal curvatures κ_1 and κ_2 and principal curvature directions u_1 and u_2 a cubic polynomial P_β is fitted to these raw points based on the local tangent plane defined by n_β . The Weingarten map of P_β yields κ_1 , κ_2 , u_1 , and u_2 . Note, that these estimates are only used if the ratio of the smaller eigenvalues of the PCA is less than $1/2$.

In [18] every n-ball β corresponds to one vertex v_β of a triangle mesh. Its position is computed as the arithmetic mean of raw points of β projected to P_β . However, for the on-line reconstruction no triangle mesh is constructed. Still, v_β is used to represent the geometry of β in subsequent reconstruction steps. Similar to [6], each n-ball holds a list of 20 nearest vertices in the ball tree.

IV. SEGMENTATION

For the segmentation of the scan data the n-balls are classified by their local geometry. Based on this classification, an on-line region growing algorithm is applied to obtain segments of the same surface type (Section IV-C). This region growing is guided by segmentation criteria ensuring geometric consistency within segments (Section IV-B). Based on the information about global surface segments the geometric classification is improved (Section IV-D).

A. N-ball classification

In [2] a classification of a surface based on the Gaussian and mean curvature is introduced. The resulting eight surface types are shown in Table I. This classification is applied to each n-ball using the Gaussian $K = \kappa_1 \kappa_2$ and the mean curvature $H = (\kappa_1 + \kappa_2)/2$. For numerical reasons all moduli of K and H below a threshold ε_K and ε_H are considered to be zero.


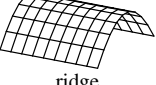

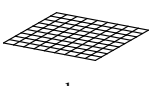
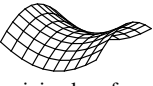
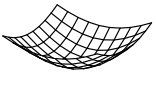
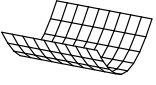
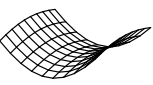
	$K > \varepsilon_K$	$ K \leq \varepsilon_K$	$K < -\varepsilon_K$
$H < -\varepsilon_H$	 peak	 ridge	 saddle ridge
$ H \leq \varepsilon_H$	—	 plane	 minimal surface
$H > \varepsilon_H$	 pit	 valley	 saddle valley

Table I: Eight surface types determined by the Gaussian and mean curvature [2].

B. Segmentation criteria

N-balls of the same surface type are clustered to surface segments. Depending on the surface type, different segmentation criteria are used in this clustering step. These different criteria are explained in the sequel.

The ratio of the two smallest eigenvalues resulting from PCA allows us to assess the quality of estimated normals during the estimation process itself. Whenever a certain quality threshold ε_q is not satisfied, we do not use the n-ball for segmentation.

1) *Criterion for a planar segment*: To cluster n-balls to a planar segment the n-ball normals n_β are used. Two n-balls are clustered to the same planar segment, if the angle between their normals is less than a threshold ε_n .

Two planar segments with the same normal do not necessarily belong to the same plane. If such planar segments are close to each other, the normal angle will not be sufficient to discriminate these segments. If β_1 and β_2 are two n-balls with parallel normals, the perpendicular distance

$$d_\perp(\beta_1, \beta_2) = (n_{\beta_1} + n_{\beta_2}) \cdot (v_{\beta_1} - v_{\beta_2}) / \|n_{\beta_1} + n_{\beta_2}\|$$

is used. Thus, β_1 and β_2 are clustered to the same segment, if $d_\perp(\beta_1, \beta_2)$ is less than ε_{d_\perp} .

2) *Principal curvature criterion*: For spherical and cylindrical segments we use principal curvature.

Spheres: All points on a sphere have the same maximum and minimum curvatures. Due to measurement noise of the laser scanner, a threshold ε_s for the curvature of n-balls on spheres is used. To enhance the robustness of this criterion, the average maximum $\bar{\kappa}_1$ and minimum curvature $\bar{\kappa}_2$ of all n-balls belonging to one segment is compared to the principal curvature κ_1 and κ_2 of a candidate n-ball

$$|\kappa_i - \bar{\kappa}_i| < \varepsilon_s \quad \text{for } i = 1, 2.$$

Cylinders: All points on a cylinder have the same maximum principal curvature κ_1 . Again, a candidate n-ball's maximum principal curvature κ_1 is compared to the

segment's average maximum curvature $\bar{\kappa}_1$

$$|\kappa_1 - \bar{\kappa}_1| < \varepsilon_c.$$

3) *On-line computation of average curvatures*: N-balls might be added and deleted from segments in an arbitrary order. Simultaneously the average principal curvature of segments must be updated. If $\bar{\kappa}_i^n$ denotes the average principal curvature of a segment s with n n-balls and a new n-ball β with principal curvature κ_i is added, the new average principal curvature $\bar{\kappa}_i^{n+1}$ of s is computed as

$$\bar{\kappa}_i^{n+1} = (\kappa_i + n\bar{\kappa}_i^n) / (n + 1) \quad \text{for } i = 1, 2.$$

If instead β is deleted, the new average principal curvature $\bar{\kappa}_i^{n-1}$ of s is computed as

$$\bar{\kappa}_i^{n-1} = (n\bar{\kappa}_i^n - \kappa_i) / (n - 1) \quad \text{for } i = 1, 2.$$

When the local geometry estimate of an n-ball is updated, the average principal curvatures of the according segments are updated by deleting the n-ball with the old geometry and adding the n-ball with the new geometry.

4) *Maximum curvature direction criterion*: Two cylindrical segments with the same average principal curvatures do not necessarily belong to the same cylinder. If such cylindrical segments are close to each other, the principal curvature criterion is not sufficient to discriminate these segments.

The principal curvature direction u_1 of the maximum principal curvature κ_1 of a cylinder is perpendicular to the axis of the cylinder. Thus, the angle between the maximum principal curvature directions is used to discriminate close cylindrical segments. Two n-balls β_1 and β_2 are clustered to the same cylindrical segment, if the angle between the projection of their maximum principal curvature directions to the plane $n_{\beta_1} + n_{\beta_2}$ is less than a threshold ε_{cd} .

C. On-line region growing

N-balls are clustered into segments based on their surface type and the segmentation criteria using region growing. This region growing processes the data stream on-line. This implies two requirements for region growing:

- The n-balls are received by the region growing in an arbitrary order, because the raw points are streamed in an arbitrary order, the implementation is parallelized, and the human operator might pause the scan process and continue at an arbitrary new region of the object.
- The local geometry estimates of an n-ball can change, because of multiple scans of the same object region. Thus, segments can merge or split at any time and all affected n-balls must be re-assigned.

When an n-ball is received by the region growing, it is either added, deleted or updated.

Adding n-balls: Every new n-ball is initially assigned to a new segment. Then, the region growing starts from this n-ball analyzing its immediate neighbors. These are clustered into a common segment depending on the surface type and segmentation criteria as shown in Table II. N-balls with surface types saddle ridge, saddle valley, or minimal surface are clustered into free-form segments. The analysis of the

Surface type	Threshold	Minimized quantity
flat	$\varepsilon_{d_{\perp}}$, perpendicular distance	normal angle
ridge, valley	ε_{cd} , curvature direction angle	principal curvature difference
peak, pit		mean curvature difference
other		normal angle

Table II: Segmentation according to segmentation criteria.

neighboring n-balls either yields a most similar n-ball or no similar n-ball. In the latter case, the new n-ball keeps its new segment. In the former case, the segments of the new and its most similar n-ball are merged, where the n-balls of the segment with less n-balls are assigned to the other segment and the smaller segment is deleted.

Deleting n-balls: To delete an n-ball, it is deleted from its segment. If it was the last n-ball of this segment, the segment is also deleted.

Updating n-balls: To update the local geometry of an n-ball, it is deleted from its actual segment and a new temporary segment is generated. For this new temporary segment the region growing is triggered. Thus, the changed n-ball is assigned to the segment with the most similar geometry.

D. Segment type detection

Initially, the surface type of a segment is determined by the surface type of its first n-ball. When the segment contains at least ten n-balls, the surface type is determined based on the average principal curvature of all its n-balls. This yields a more robust classification of the segment, especially for spherical or cylindrical segments with large radii. Table III shows the criteria for segment type classification based on the average principal curvatures $\bar{\kappa}_1$ and $\bar{\kappa}_2$. The threshold ε_{PC} is used for numerical reasons. Because the average curvatures are more stable than n-ball curvatures, ε_{PC} is smaller than ε_K and ε_H .

Average curvature criterion	Segment type
$ \bar{\kappa}_1 , \bar{\kappa}_2 \leq \varepsilon_{PC}$	planar
$ \bar{\kappa}_1 \leq \varepsilon_{PC} \oplus \bar{\kappa}_2 \leq \varepsilon_{PC}$	cylindrical
$ \bar{\kappa}_1 , \bar{\kappa}_2 > \varepsilon_{PC} \wedge \bar{\kappa}_1 - \bar{\kappa}_2 \leq \varepsilon_{PC}$	spherical
$ \bar{\kappa}_1 , \bar{\kappa}_2 > \varepsilon_{PC} \wedge \bar{\kappa}_1 - \bar{\kappa}_2 > \varepsilon_{PC}$	ellipsoidal
none of the above	free-form

Table III: Segment types determined by average principal curvature, using the *exclusive or* operator \oplus .

V. SEGMENT BOUNDARIES

Geometric primitives are reconstructed using quadrics. This requires a definition of the boundary of the segments to trim the resulting geometric primitives.

Alpha shapes provide a description of the shape of a set of points. Originally developed for two dimensions [19], alpha shapes can be extended to 3d point clouds [20]. For the reconstruction, alpha shapes are used to determine n-balls that lie on the boundary of a segment. However, for the reconstruction of the geometric primitives with quadric surfaces, a spacial boundary polygon for each segment is required. Therefore, the alpha shapes are restricted to an approximate tangent plane.

A. Planar alpha shapes

The alpha shapes algorithm is used to classify points of a point cloud as boundary or inner points. The essential steps of this algorithm are ([19]):

- 1) For each points p collect all neighboring points within a radius 2α .
- 2) Search for an empty circle with radius α that touches p and one of its neighbors q .
- 3) For each such pair p, q store the line segment between them.

B. Alpha shape test for 3d surface boundaries

The 2d alpha shapes approach can be generalized to boundaries of surfaces in \mathbb{R}^3 . Instead of circles, spheres with radius α are used whose center lies in the approximate boundary tangent plane. The centers of the spheres touching boundary points are obtained as follows, see Figure 2:

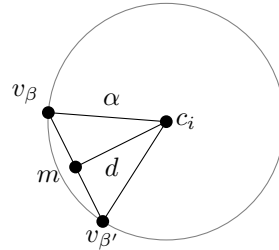


Figure 2: Cross-section in the plane spanned by u, v of a sphere with radius α for an n-ball pair β, β' .

- 1) For an n-ball β with 2α -neighbor β' compute the normal u of the plane spanned by n_β and $v = v_{\beta'} - v_\beta$ as $u = n \times v$. Note, that an approximate tangent plane along the edge u is spanned by u and v .
- 2) Compute midpoint m of v_β and $v_{\beta'}$ as $m = v_\beta + v/2$.
- 3) Compute distance d of m to the circle center as $d = \sqrt{\alpha^2 - \|v\|^2/4}$.
- 4) Compute sphere centers $c_i = m \pm d \cdot u/\|u\|, i = 1, 2$.
- 5) If one of these spheres is empty, β is at the boundary and v is a boundary edge.

Whenever an n-ball is added or updated, the alpha shape test is applied to classify the n-ball as boundary or inner ball. Note, that only neighboring n-balls belonging to the n-ball's segment are considered.

C. Boundary assembly

After the boundary n-balls are detected, a boundary polygon is assembled for each segment. Thus, the boundary n-balls and boundary edges need to be sorted.

For the construction of the boundary polygon a back-tracking strategy is used. The algorithm starts at a boundary n-ball and proceeds with one of its boundary neighbors. Already visited edges are not visited again. When a dead end is reached, a back-tracking step is done and the algorithm proceeds with a different boundary neighbor.

To avoid the detection of small boundary loops that do not enclose most of the segment, we accept a boundary polygon only when it contains at least 70% of the boundary n-balls. The algorithm terminates when it reaches the first n-ball again and when the 70% condition is met.

VI. PRIMITIVE RECONSTRUCTION

After the n-balls are segmented and a boundary polygon is extracted for each segment, geometric primitives are reconstructed for each segment.

In order to reduce the memory consumption and the complexity of the representation implicit quadrics are used, similar to [13]. Using this representation the set of n-balls for a segment can be reduced to a set of ten coefficients defining a quadric

$$0 = q(x) = x^T Q x + 2P^T x + R \quad \text{for } x \in \mathbb{R}^3$$

where

$$Q = \begin{bmatrix} A & D & E \\ D & B & F \\ E & F & C \end{bmatrix} \quad \text{and} \quad P = \begin{bmatrix} G \\ H \\ I \end{bmatrix}.$$

After the fitting the quadric surface is trimmed at the boundary polygon to yield a standard b-rep of the scanned points.

A. Quadric fitting

In order to fit a quadric surface to a segment an algebraic approach is used. Due to the classification of the segments this algebraic fit converges robustly.

A segment consists of n-balls β_1, \dots, β_n . To fit a quadric surface to this segment the least square solution of the over-determined system of equations

$$q(v_{\beta_i}) = 0, \quad \text{for } i = 1, \dots, n \quad (1)$$

is computed. To improve the robustness of this fit the additional conditions $\nabla_n q = 1$ and $\nabla_n^2 q = 0$ are used,

where ∇_n is the directional derivative with respect to the normal direction n . This leads to the additional equations

$$\nabla_{n_{\beta_i}} q(v_{\beta_i}) = 1, \quad \text{for } i = 1, \dots, n \quad \text{and} \quad (2)$$

$$\nabla_{n_{\beta_i}}^2 q(v_{\beta_i}) = 0, \quad \text{for } i = 1, \dots, n. \quad (3)$$

These two conditions control the size of the coefficients of q and ensure that q is positive in direction of the normal. For the fit a linear combination of (1), (2), and (3) is solved using singular value decomposition (SVD).

Plane fitting: Using the above quadric fitting for planes might result in a double plane due to noise in the data. However, for a plane not the complete set of quadric parameters is necessary. Using the segment type and setting $Q = 0$ in (1) and (2) simplifies the fitting for planes.

B. Parameter estimation

Quadrics are usually represented in normal form. For this the principal axes of the quadric and characteristic parameters of the surface (e.g., radii, etc.) are computed.

The canonical system of a quadric is given by the eigenvectors e_1, e_2, e_3 of Q . To get the correct translation of the quadric surface, the equation $Qx + P = 0$ is solved. The solution of this equation yields either a center point, a point on a centerline, or a point on a center plane. This yields a complete representation of the spatial position of the quadric surface.

For some degenerated quadrics (e.g., planes) this approach fails. Furthermore, to compute the characteristic parameters of a quadric surface special rules apply. Which rules apply is determined by the segment type. These rules are different for planes, spheres, cylinders, and cones.

Plane: Since for a plane Q is singular the computation of the spatial position is different. The translational component is determined by the position of a center point, which is the mean of all boundary points projected onto the surface. A possible choice for the canonical system is the normal of the segment and any suitable 2d system in the plane.

Sphere: Since the quadric representation of a sphere has a single center the translation of the sphere can be computed directly from its center point. The radius of the sphere is computed by ray casting the axes of the canonical system from the center point. To compute the intersections the ray equation is substituted into q .

Cylinder and cone: The solution of $Qx + P = 0$ for the cylinder yields an arbitrary point on the center line of the cylinder. This center line has direction e_3 . Projecting all vertices of boundary n-balls onto e_3 yields two extremal projected points. The planes through these points normal to e_3 are the upper and lower cap of the cylinder. For the translational component, the intersection of the bottom cap with the center line gives a center point c of the cylinder. The distance of the caps is the height h of the cylinder. To estimate the radius of the cylinder the distance of $c +$

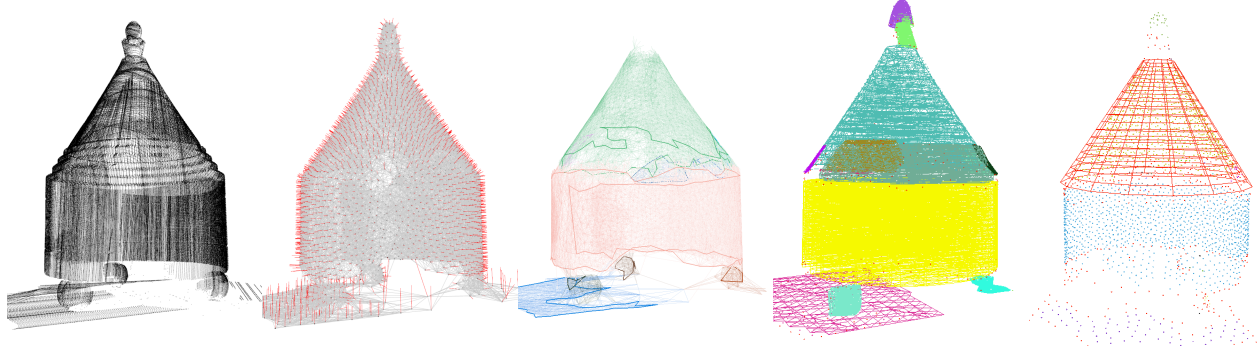


Figure 3: Scan of a casket. This example shows results of our segmentation and reconstruction methods using high quality scan data. Processing steps: raw points, n-balls with normals, segment boundaries, quadric surfaces, reconstructed primitives.

he_3 to the cylinder surface is computed using ray casting in direction e_1 and e_2 .

For a cone the computations are the same except that the radii are computed in the top and bottom cap.

VII. RESULTS AND DISCUSSION

We provide experimental results of the on-line segmentation and reconstruction for different physical models. The experimental results show the strengths and weaknesses of the presented on-line approach.

For the experiments recorded scans from two different hand-held laser scanners were used. The casket in Figure 3 was scanned with a measurement arm based scanner with high precision and low noise. For all other models we used a scanner with an optical tracking device, which causes additional noise and reduces the repetition accuracy.

A. Thresholds

During the experiments different values for the thresholds described in Section IV-B were tested. Because the algorithms shall be used in automated reverse engineering processes, an extensive calibration phase before each scan is not desirable. Thus, for a comparison the thresholds in Table IV were used for all experiments. The first three thresholds

Symbol	Threshold	Value
ε_K	Gaussian curvature	0.03
ε_H	Mean curvature	0.07
ε_{PC}	Principal curvature	0.015
ε_n	Plane normal angle	5°
ε_q	Plane normal quality	0.5
$\varepsilon_{d\perp}$	Perpendicular distance	0.5
ε_s	Principal curvature of spheres	0.2
ε_c	Principal curvature of cylinders	0.2
ε_{cd}	Maximum curvature direction	8°

Table IV: The thresholds used for the presented results.

in Table IV are most critical, because they directly influence the surface type classification of the n-balls and the segment type classification based on principal curvature.

B. N-ball classification

Different objects were scanned to test the on-line segmentation and reconstruction. The following discussion and the screen shots illustrate the performance of these algorithms while focusing on critical aspects.

The surface types of the n-balls determined by the Gaussian and mean curvature are shown in Figure 4. The raw points are also shown, colored according to their n-ball's type. In order to cope with the scan noise, relatively large values are chosen for ε_K and ε_H . This is observable at places with high noise. This leads to a false classification in Figure 4 of the cylinders as planes.

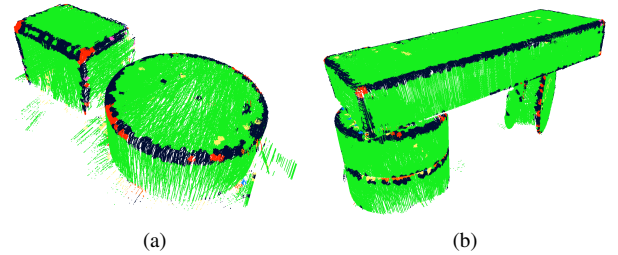


Figure 4: Surface types of the n-balls according to Section IV-A. Raw points are displayed and colored according to parent n-ball. Colors: plane, ridge, valley, peak, pit, saddle ridge, saddle valley, minimal surface.

C. Segment type detection

Figure 5 shows the surface type classification of the segments. All cylinders are correctly classified.

Figure 5b shows that the classification of elliptical parts might be unstable. The left part of the cylinder's top blend is classified as cylindrical (blue) and not as ellipsoidal (pink) as on the front part of the cylinder.

The individual segments are shown in Figure 6. Narrow areas, i.e., the cylindrical blends in Figure 6a, are subdivided

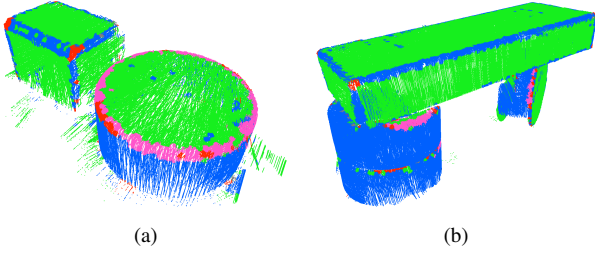


Figure 5: Segment type according to Section IV-D. Colors: planar, cylindrical, spherical, ellipsoidal, free-form.

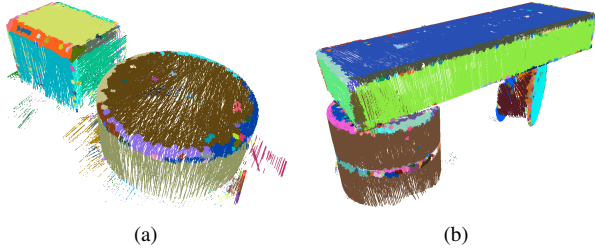


Figure 6: Resulting segments colored individually.

into multiple segments. They are so narrow, that misclassification of single n-balls can disconnect the segments. On the other hand, the two cylinders in Figure 6b are represented by a single segment. N-balls of both object regions are connected by neighborhood links. This is a typical example where the operator sees that the reconstruction is unsatisfactory and additional scan passes are required.

D. Boundary

Figure 7 shows the boundary polygons of the segments. It can be observed that narrow segments are problematic for the boundary detection and assembly. Because many possible routes through the boundary edges exist, it is not guaranteed that a surrounding boundary is found.

E. Primitive reconstruction

The plane fitting yields the optimal plane representation, due to its simplified fitting method. The quadric fitting is not

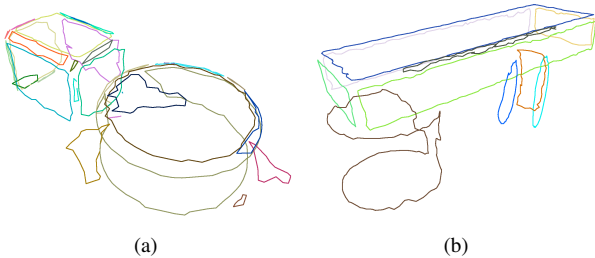


Figure 7: Boundaries of the segments.

limited to a single geometric primitive. Thus, for a segment with noise the quadric surface with the smallest error is computed. This can have the effect that for segments with little noise the fitted quadric surface has a smaller error than the perfect quadratic surface. This is demonstrated in Table V where the fit of a synthetic cylinder with and without noise is compared with the theoretical optimal cylinder fit.

	with noise	without noise
error of optimal cylinder	2.959	0.322
standard deviation	0.072	0.124
error of fitted cylinder	2.664	0.348
standard deviation	0.193	0.223
difference	0.295	0.026

Table V: Mean error and deviation of n-balls to a fitted surface and an optimal surface in millimeters. A synthetic cylinder with and without uniformly distributed noise of ten percent of the radius of the cylinder are used. The values are the averages for ten runs of the reconstruction algorithm.

The methods proposed for the primitive reconstruction are able to reconstruct all described geometric primitives as shown in Figures 3 and 8. The accuracy of the geometric model depends on the accuracy of the implicit quadric surface and on the boundary description.

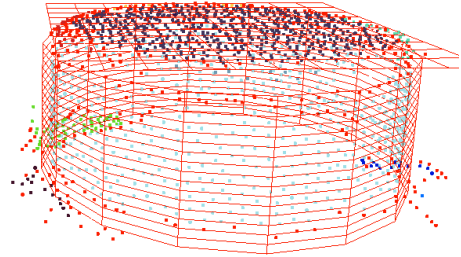


Figure 8: Geometric primitives reconstructed from a scan of a wooden cylinder with noise and low repetition accuracy.

VIII. CONCLUSION AND POSSIBLE FUTURE WORK

In this paper an on-line segmentation and reconstruction method is proposed. This method effectively handles points streams and processes this point data on-line on all levels of the reconstruction process. This on-line approach has the significant advantage that the reconstruction process and quality can be evaluated immediately by the human operator. To this end, the presented method handles the points stream efficiently for a real time on-line reconstruction.

For the future there are some aspects to improve, that do not affect essentially the effectiveness or efficiency of the presented method. Some minor aspects for improvement of the on-line reconstruction process concern multiple boundaries, trimming, and SVD computation. Firstly, the segmentation in Section IV so far supports single segment

boundaries. A typical example of a geometric primitive with two boundaries is the lateral surface of a cylinder or cone. The back-tracking algorithms needs to be extended to support this. Secondly, primitives are currently not trimmed. The boundary polygons of the segments will be projected to the geometric primitives and used as trim curves. Thirdly, for the quadric fitting in Section VI-A a SVD is used, where an on-line SVD method as [21] could improve the performance.

Furthermore, the presented method is limited to surfaces that can be reconstructed by quadrics. Blends in CAD models may contain surface types that can not be reconstructed by quadrics, e.g., a monkey saddle. For these surfaces a more general surface representation is required. Instead of reconstructing geometric primitives, a reconstruction of rolling ball blends would be useful for many CAD objects.

ACKNOWLEDGMENTS

This work was supported by DFG grants UM 26/5-1 and IRTG 1131.

REFERENCES

- [1] G. J. Agin and T. O. Binford, "Computer description of curved objects," *IEEE Transactions on Computers*, vol. C-25, no. 4, pp. 439–449, 1976.
- [2] P. J. Besl and R. C. Jain, "Segmentation through variable-order surface fitting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 2, pp. 167–192, 1988.
- [3] N. Haala and K.-H. Anders, "Acquisition of 3d urban models by analysis of aerial images, digital surface models and existing 2d building information," in *SPIE Conference on Integrating Photogrammetric Techniques with Scene Analysis and Machine Vision III*, 1997, pp. 212–222.
- [4] A. Shamir, "A survey on mesh segmentation techniques," *Computer Graphics Forum*, vol. 27, no. 6, pp. 1539–1556, 2008.
- [5] P. Benkő and T. Várady, "Direct segmentation of smooth, multiple point regions," in *Geometric Modeling and Processing*. IEEE, 2002, pp. 169–178.
- [6] I. Yamazaki, V. Natarajan, Z. Bai, and B. Hamann, "Segmenting point-sampled surfaces," *The Visual Computer*, vol. 26, no. 12, pp. 1421–1433, 2010.
- [7] T. Várady, "Reverse engineering of geometric models – an introduction," *Computer-Aided Design*, vol. 29, no. 4, pp. 255–268, 1997.
- [8] P. Benkő, R. R. Martin, and T. Várady, "Algorithms for reverse engineering boundary representation models," *Computer-Aided Design*, vol. 33, no. 11, pp. 839–851, 2001.
- [9] J. Wu and L. Kobbelt, "Structure recovery via hybrid variational surface approximation," *Computer Graphics Forum*, vol. 24, no. 3, pp. 277–284, 2005.
- [10] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for point-cloud shape detection," *Computer Graphics Forum*, vol. 26, no. 2, pp. 214–226, 2007.
- [11] G. Biegelbauer and M. Vincze, "Efficient 3D object detection by fitting superquadrics to range image data for robot's object manipulation," in *International Conference on Robotics and Automation*. IEEE, 2007, pp. 1086–1091.
- [12] S. Petitjean, "A survey of methods for recovering quadrics in triangle meshes," *ACM Computing Surveys*, vol. 2, no. 34, pp. 1–61, 2002.
- [13] M. Vančo, B. Hamann, and G. Brunnett, "Surface reconstruction from unorganized point data with quadrics," *Computer Graphics Forum*, vol. 27, no. 6, pp. 1593–1606, 2008.
- [14] R. Pajarola, "Stream-processing points," in *IEEE Visualization*, 2005, pp. 239–246.
- [15] J. Boesch and R. Pajarola, "Flexible configurable stream processing of point data," in *WSCG'2009 Full Papers*, 2009, pp. 49–56.
- [16] T. Bodenmüller and G. Hirzinger, "Online surface reconstruction from unorganized 3d-points for the DLR hand-guided scanner system," in *2nd Symp. on 3D Data Processing, Visualization and Transmission*, 2004, pp. 285–292.
- [17] K. Denker, B. Lehner, and G. Umlauf, "Online triangulation of laser-scan data," in *17th International Meshing Roundtable*, R. Garimella, Ed., 2008, pp. 415–432.
- [18] —, "Real-time triangulation of point streams," *Engineering with Computers*, vol. 27, no. 1, pp. 67–80, 2011.
- [19] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, "On the shape of a set of points in the plane," *IEEE Transactions on Information Theory*, vol. 29, no. 4, pp. 551–559, 1983.
- [20] H. Edelsbrunner and E. P. Mücke, "Three-dimensional alpha shapes," *ACM Trans. Graph.*, vol. 13, no. 1, pp. 43–72, 1994.
- [21] M. Brand, "Incremental singular value decomposition of uncertain data with missing values," in *Computer Vision – ECCV 2002*, ser. Lecture Notes in Computer Science, A. Heyden et al., Eds. Springer, 2002, vol. 2350, pp. 707–720.