Construction of Implicit Surfaces from Point Clouds Using a Feature-based Approach

Patric Keller[1], Oliver Kreylos[3], Eric S. Cowgill[3], Louise H. Kellogg[3], Martin Hering-Bertram[4], Bernd Hamann[2], and Hans Hagen[1]

1 **Department of Computer Science,**
University of Kaiserslautern, Germany.
2 **Institute for Data Analysis and Visualization (IDAV),**
Department of Computer Science, UC Davis, CA, USA.
3 **Department of Geology, UC Davis, CA, USA.**
4 **Fraunhofer ITWM in Kaiserslautern, Germany.**

Abstract

We present a novel feature-based approach to surface generation from point clouds in three-dimensional space obtained by terrestrial and airborne laser scanning. In a first step, we apply a multiscale clustering and classification of local point set neighborhoods by considering their geometric shape. Corresponding feature values quantify the similarity to curve-like, surface-like, and solid-like shapes. For selecting and extracting surface features, we build a hierarchical trivariate B-spline representation of this surface feature function. Surfaces are extracted with a variant of marching cubes (MC), providing an inner and outer shell that are merged into a single non-manifold surface component at the field's ridges. By adapting the iso-value of the feature function the user may control surface topology and thus adapt the extracted features to the noise level of the underlying point cloud. User control and adaptive approximation make our method robust for noisy and complex point data.

1998 ACM Subject Classification I.3 Computer graphics, I.3.5 Computational Geometry and Object Modeling

Keywords and phrases 3D Point Clouds, Surface Reconstruction, Implicit Surfaces;

Digital Object Identifier 10.4230/DFU.Vol2.SciViz.2011.129

1 Introduction

The exploration of point data sets, like environmental LiDaR (Light Detection and Ranging) data, is a challenging problem of current interest. Besides the vast amount of points obtained from terrestrial and airborne scanning, data complexity is increased by e.g., noise, occlusion, alternating sample density and overlapping samples. Despite of the high scanning resolution, undersampling occurs at small and fractured components like fences and leaves of trees.

In this research we introduce a new surface reconstruction approach that allows to recover surface meshes from relative complex point data sets. Since most reconstruction methods do not distinguish between surface and non-surface related point structures, they rather assume the input points always originate from surface structures, applying them to noisy and complex point data would not lead to reliable results. To perform a reliable surface reconstruction a feature-based reconstruction approach taking into account the structural composition of the acquired point cloud is proposed. It adapts the mesh reconstruction in a way that only surface-related point structures are extracted. This is accomplished by

© P. Keller, O. Kreylos, E.S. Cowgill, L.H. Kellogg, M. Hering-Bertram, B. Hamann, and H. Hagen;
licensed under Creative Commons License NC-ND

Scientific Visualization: Interactions, Features, Metaphors. *Dagstuhl Follow-Ups, Vol. 2.*
Editor: Hans Hagen; pp. 129–143
DAGSTUHL FOLLOW-UPS Dagstuhl Publishing
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Germany

first performing a structural analysis of the point cloud that filters the point data based on a classification of local point set neighborhoods. The feature values assigned to the individual points are used to define whether a point corresponds to a surface or not. The proposed approach constructs a *feature field* function approximating these feature values from which an initial surface representation is extracted. The locus of points associated with the local extrema of this implicit “feature function“ most probably represents the underlying surface structures. Thus, moving the initial surface toward these local extrema, incorporating operations adapting the topology of the surface meshes, yields the final surface approximation. Since the proposed approach is meant to complement the visualization of raw laser-scanned data, it provides a mesh-representation exhibiting a user specified topology. The topology can be controlled, similar to choosing a radius for alpha shapes in 3D space [11], in a way that the extracted surfaces adapt to the point cloud specific properties. A further advantage is that the extracted surface representation is continuous and thus can be used for further post-processing.

The remainder of this paper is structured as follows: Section 3 provides an overview about the steps of the proposed method. In Section 4, Section 5, Section 6 and Section 7 we discuss the individual steps of the reconstruction approach in detail. The results are presented in Section 8. The conclusions are provided in Section 9.

2 Related Work

Many methods exist for the construction of surfaces from scattered point data. The existing methods can be divided into explicit and volume-based reconstruction methods.

2.1 Volume-based (implicit) surface reconstruction

These reconstruction techniques transfer the input point cloud into volumetric representations implicitly defining the underlying surfaces. Using methods like the marching cubes algorithm (MC) [21] enables us to extract explicit surfaces representations in form of polygonal meshes. One of the first reconstruction methods within this field was introduced by Hoppe et al. [16]. The basic idea is to combine local signed distance functions (SDF) into a global representation from which the surface can be extracted by contouring. Following this idea the methods discussed in [33], [5] and [24] are based on defining SDFs whose zero level sets represent the desired surfaces. More recently, scattered data approximation with radial basis functions (RBF) is described in [30] [7] [6] to construct SDFs. Othake et al. [26] extended the original RBF construction to multi-scale RBFs increasing approximation quality. Shen et al. [29] improved the original method by incorporating additional normal information in the RBFs allowing a more stable reconstruction. Another possible technique for surface reconstruction concerns the usage of level set methods [31] [33]. This technique defines a function attracting the level set surface toward the data points. However, the application of these techniques is limited when working with noisy unfiltered point clouds like environmental LiDaR data.

2.2 Explicit surface reconstruction

Another possible way to obtain surfaces from unorganized point clouds is to fit parametric surface patches to the input points. The approach of Eck and Hoppe [10] fits a network of B-Spline patches in a way allowing them to reconstruct surfaces of arbitrary topology. Alexa et al. [2] introduced an approach based on moving least squares (MLS) also used in [1] [25] [12] [9] which fits polynomial surface patches to local point neighborhoods. In [22] the authors propose an MLS reconstruction scheme which takes into account local curvature approximations to obtain meshes of higher quality. One of the main problems associated with the MLS-based techniques is that they have to adapt in order to capture the correct underlying topology. Depending on the structure of the input data this can be hard to achieve. Another drawback is that the MLS-technique in general is not capable of constructing surfaces having sharp features. One approach for solving this problem was published by Fleishman et al. [12]. Alternative methods like the power crust (PC) algorithm introduced by Amenta et al. [3] the 3D Alpha Shapes method from [11], the Cocone algorithm [8] as well as some other approaches [23] [17] also produce good results concerning surface reconstruction from point data.

Most of the existing techniques make the assumption that the point cloud exclusively contains points corresponding to surface-like structures. In cases the point cloud includes objects having non-surface character, like trees or bushes as is the case in environmental point clouds featuring vegetation, most of the existing methods would fail. We propose an approach able to extract surfaces even from these types of point clouds. We do not make any assumptions on the structural composition of the point clouds in advance. This allows us to apply our method to nearly any type of point data. The provided results suggest that our method holds great promise in the field of surface reconstruction from unorganized environmental point cloud data.

3 Overview

The proposed method aims at generating an implicit representation of the underlying surface(s) in form of a 3D scalar field, named feature field. From this feature field the final surface representation is extracted using a MC-based approach. The proposed feature-based surface reconstruction consists of the following steps:

1. Pre-processing. This step includes the organization of the input points into an octree-based voxel structure as well as the determination and assignment of feature values to the individual points. These feature values in a certain sense express the likelihood that a point is part of a surface structure by measuring the ”planarity” of a point’s local neighborhood.
2. The approximation/interpolation of the computed feature values using 3D (trivariate) hierarchical B-splines provides the scalar-valued feature field function.
3. The extraction of an initial surface mesh, associated with an user-specified iso-value by using the MC-algorithm.
4. The adaption of the initial surface representation to approximate the meta-surface which represents the local maxima of the corresponding feature field. This includes additional refinement strategies increasing the quality of the final surface representation.

4 Pre-Processing

4.1 Hierarchical Point Cloud Decomposition

For purposes of efficient data processing the proposed reconstruction method applies the octree-based data organization scheme. Considering the applied surface reconstruction algorithm, this form of data management has several advantages: Besides the efficient handling and visualization of point clouds as well as the acceleration of certain point operations like k-nearest neighbor searches, it facilitates the construction of the feature field function since it makes possible applying the hierarchical B-spline approximation scheme. It also supports a straightforward MC-based iso-surface extraction by incorporating the edges of the multi-level voxel grid;

4.2 Determining Feature Values

The determination of the feature values expressing the correspondence of individual points to surfaces is based on a former approach described in [18]. The feature values are defined in a way that they measure the "planarity" of the local neighborhood $N(p)$ of a point $p \in P$, whereas $N(p)$ either can be defined by the set of k-nearest neighbor points of p, or the set of points lying within a specified radius r of p. In principle, we want the feature value of p to be high, if $N(p)$ corresponds to a smooth surface and low otherwise. This is achieved by analyzing the geometric shape of $N(p)$ by relating the eigenvalues $\lambda_0 \leq \lambda_1 \leq \lambda_2$ of the covariance matrix $\mathbf{C}$ of $N(p)$ amongst each others. It holds that $N(p)$ has a spherical shape if $\lambda_0 \approx \lambda_1 \approx \lambda_2$, a planar shape if $\lambda_0 \ll \lambda_1 \wedge \lambda_1 \approx \lambda_2$ and a cylindrical shape if $\lambda_0 \approx \lambda_1 \wedge \lambda_1 \ll \lambda_2$. We combine this information and define the surface feature value f of a point p to be

$$f(p) = \frac{\lambda_1 - \lambda_0}{\lambda_2}. \tag{1}$$

The size of $N(p)$ has major impact on the obtained value f and is an important influence factor controlling the quality of the final surface representation. Choosing $N(p)$ to be small compared to the noise ratio or the point density results in strongly varying feature field interfering with the reconstruction process. Forcing $N(p)$ to be too large will degrade the quality of the extracted surface. This effect is discussed more detailed in Section 8.

5 Scattered Data Approximation

We consider the approximation of the introduced feature values $f_i \in \mathbb{R}$ assigned to the input points $p_i \in P \subset \mathbb{R}^3$. Obtaining a smooth global feature function F approximating the given feature values f_i becomes a scattered data problem. This problem can be formalized as follows: Given a finite set of scattered data points $\{(p_i, f_i) | p \in \mathbb{R}^3, f_i \in \mathbb{R}\}$, construct a function $F : \mathbb{R}^3 \to \mathbb{R}$ that minimizes the residuals $\|F(p_i) - f_i\|$. There are several possible ways for accomplishing this task like using approximation schemes based on, e.g., RBFs [13] [27] or spline fitting techniques [19] [15]. Good overviews of existing scattered data interpolation/approximation techniques are provided in [14] [20] [27] [28]. The introduced method follows the ideas presented in [4] [32] using hierarchical B-splines defined over the rectangular regions of a corresponding quadtree. This approach is able to provide a smooth and adaptive approximation of the feature field by a set of hierarchical trivariate B-spline functions. Extended to the 3D case, starting from the original bounding voxel enclosing the entire point cloud, the approximation scheme consists of the following steps:

1. The initial approximation of the determined feature values associated with the individual points by triquadratic Bernstein-Bézier polynomials.
2. Joining the individual Bernstein-Bézier polynomials into one continuous B-spline representation by applying the knot removal scheme proposed by Bertram et al. [4].
3. The refinement of the approximation by performing above steps to the voxel grid at the next finer octree level.

Recursively applying these steps provides a hierarchical approximation of the feature values.

5.1 Initial Bézier Fitting

Let $P_i = \{(p_j, f_j)\}$ be a set of n scattered data points associated with the voxel V_i. P_i is approximated by a trivariate Bernstein-Bézier polynomials of the form

$$F_i(p_j) = \sum_{i=0}^{l} \sum_{j=0}^{l} \sum_{k=0}^{l} c_{i,j,k} \phi^l_{i,j,k}(p_j), \tag{2}$$

where $\phi^l_{j,j,k}(p_j)$ are the corresponding trivariate basis functions of degree l. The solution for F_i is found by solving Equation (5.1) according the basis coefficients $c_{i,j,k}$. In the case that an overdetermined system is given, F_i is determined by minimizing the sum of the squared residuals φ with respect to the basis coefficients $c_{i,j,k}$

$$\varphi = \sum_{j=0}^{n} \|F_i(p_j) - f_j\|^2. \tag{3}$$

To deal with an under-determined system we add additional constraints forcing F_i to become zero in regions where no scattered data samples are given. The additional term has the form

$$\psi = \sum_{i=0}^{l} \sum_{j=0}^{l} \sum_{k=0}^{l} (c_{i,j,k})^2 \ . \tag{4}$$

Minimizing

$$\varepsilon = \varphi + \psi \longrightarrow min \tag{5}$$

requires solving a relatively small linear system of equations yielding the desired function F_i. Since we are interested in obtaining a continuous representation we apply further processing.

5.2 Global B-spline Approximation

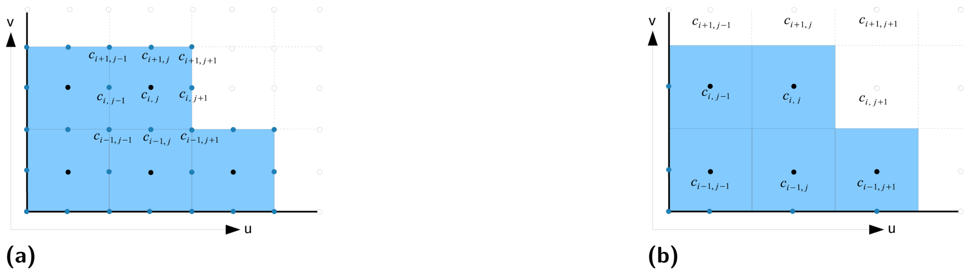

Figure 1 (a) Initial distribution of the basis coefficients defining corresponding biquadratic B-spline functions within an adaptive quad-grid consisting of empty and non-empty quadrilateral. Blue points represent zero-valued basis coefficients located at the domain boundaries of the data set having knot multiplicity three, black and white points are basis coefficients having knot multiplicity one. The white points are zero-valued basis coefficients of empty quadrilaterals. (b) quad-grid after knot removal.

To obtain a continuous approximation, we merge the individual Bernstein-Bézier polynomials F_i into a global B-spline representation F. The Bernstein-Bézier polynomials of degree l associated with the individual voxels can be represented as one global piecewise continuous B-spline function in which the control points at the voxel boundaries (faces, edges, vertices) have multiplicity l. The multiplicity of the knots associated with control points at the inside of a voxel is one. Figure 1a shows a 2D example that illustrates the distribution of the control points corresponding to a biquadratic B-spline over a quadrangular domain. The grid consists of empty (white) and non-empty (blue) quads. If a basis coefficient is located on the domain boundary of the point data set, its value is forced to be zero. The domain boundary of the data set in the 3D case is defined by the vertices, the edges and the faces of the smallest voxel enclosing the entire point set. In regions where voxels are classified as empty the control points are set to zero (depicted white). Removing knots at the voxel boundaries that are not located on the domain boundaries, yields the desired B-spline representation (see Figure 1b). In the triquadratic case the resulting B-spline is C^1 continuous. The knot removal may affect the quality of the approximation and thus demands re-fitting the remaining coefficients $c_{i,j,k}$. This is accomplished by sequentially readjusting each coefficient

$$c_{i,j,k} = \frac{\sum_{p \in V_i} \left(f_p - \sum_{x=0, x \neq i}^{l} \sum_{y=0, y \neq j}^{l} \sum_{z=0, z \neq k}^{l} c_{x,y,z} \phi_{x,y,z}(\vec{p}) \right)}{\sum_{p \in V_i} \phi_{i,j,k}(\vec{p})}, \tag{6}$$

where $\vec{p} \in \mathbb{R}^3$ denotes the coordinates of the points $p \in V_i$ in parameter space, and $f_p \in \mathbb{R}$ terms the feature value associated with p. Each adjustment of $c_{i,j,k}$ again has impact on the quality of the approximation in the neighbor voxel. Hence, the re-fitting becomes an iterative process. The process is terminated if the local approximation error $|\sum_{p \in V_i} F(p) - f_p|$ falls below a given error bound or a maximum number of iterations has been performed. The resulting function F represents a continuous approximation of the feature field function getting zero at the point clouds domain boundaries.

Problems arise in cases, where the least squares residual is small, but the slopes or curvatures of the trivariate B-spline functions are extremely high. This, especially, occurs in regions where the distribution of the points or the function values is one-sided. For example, in ”empty” voxel regions the function values may take arbitrarily large values which results in a poor representation. To encounter this problem, additional constraints are incorporated forcing the final representation to be smooth. The enhanced approximation scheme incorporates the minimization of the first principal derivatives of the B-spline representation F at the positions of the individual points.

5.3 Recursive Refinement

Let $L_j = \{(P_i^j, F_i^j), \cdots\}$ be a set where P_i^j are the scattered point sets associated with the individual voxels at octree level j and F_i^j be the corresponding approximating B-spline functions. Starting at octree level j we determine F_i^j for each $P_i \neq \emptyset$. The subsequent merging of adjacent functions F_i^j is performed by applying the above introduced knot removal. This leads to a globally C^1 continuous representation F^j at level j. In order to obtain L_{j+1} we recompute the feature values

$$\Delta f^{j+1}(p) = \Delta f^j(p) - F^j(p) \tag{7}$$

for each point, where $\Delta f_p^0 = f_p$ are the initial feature values assigned to p. This yields the basis for the subsequent approximation step performed at level $j+1$. Again, applying the least squares fitting procedure and merging the individual Bernstein-Bézier polynomials at level $j+1$ results in the continuous representations ΔF_i^{j+1} of the underlying residuals Δf_i^{j+1}. The final approximation is defined by

$$F^{j+1} = F^j + \Delta F^{j+1}. \tag{8}$$

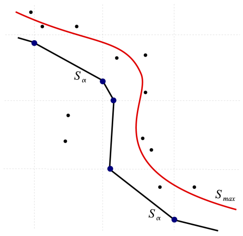

Figure 2 Two-dimensional example illustrating the extraction of the base mesh S_α via the MC algorithm regarding a specified iso-value α. The locus of the local maxima of F to be approximated by S_α is represented by the curve S_{max}.

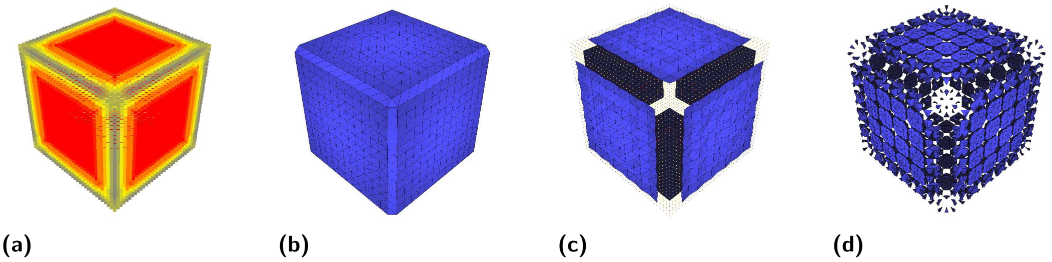

Figure 3 Example data set illustrating the effects of varying the iso-value at the initial extraction procedure. (a) Input point cloud of a simple test cube data set with points colored from blue to red according the correspondence of points to surface-related structures. (b)-(d) extracted base meshes obtained for iso-values (b) 0.4, (c) 0.6 and (d) 0.9.

The choice of the level at which the approximation is started and terminated has major impact on the performance and the quality of the final approximation.

For example, starting the computation at a low-resolution level e.g., on the root of the octree, and performing a large number of refinement steps would be computationally expensive. Otherwise, performing too few refinement steps may result in low-quality approximations. The B-spline refinement can be stopped, if a specific maximal refinement level is reached or the approximation error $|\sum_{p \in V_i} F^j(p) - f_p|$ approaches a predefined limit.

6 Initial Surface Extraction

Given a continuous function F approximating or interpolating the feature values assigned to the individual points, the first step aims at the extraction of a base mesh from F using the MC-algorithm [21]. The quality of the extracted mesh depends on the resolution of the voxel as well as on the quality of F. In general, the MC algorithm generates two-manifold meshes reflecting the topology of the iso-surface at the given extraction level (see Figure 2). Varying the iso-value results in base meshes having different topological characteristics. At this point the user has to control the surface topology by choosing the iso-value, such that the extracted mesh reflects the desired topology. This approach is inspired by the alpha-shape reconstruction method by Edelsbrunner and Mücke [11], where a Delaunay-complex is constructed for a point set, erasing all simplices whose circumscribed hypersphere exceeds a predefined radius alpha. Figure 3 shows some examples regarding a simple test data set. It illustrates the variation in topology, for extracting an initial surface from a precomputed feature function F, by varying the iso-value from low to high.

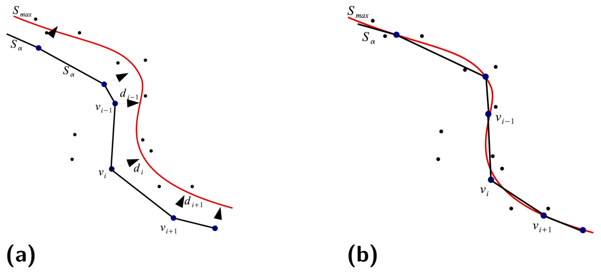

Figure 4 Examples illustrating the principles of approximating the meta-surface $S_{\max}$ representing the locus of local maxima of F. (a) Initial approximation S_α with vertices v_i and corresponding gradient vectors d_i, before the vertex displacement is applied. (b) Resulting mesh after the displacement of the vertices v_i along the vectors d_i toward S_{max}.

7 Approximation of Local Maxima

Next, the generated base is adapted to approximate the meta-surface representing the locus of local maxima of F. The set of points associated with this meta-surface is defined by $S_{\max}$. In a sense, $S_{\max}$ represents "ridge-lines" within the feature field.

The locus of the local maxima of F is defined by

$$S_{max} = \{p \in \Omega | \nabla F(p) = 0 \wedge \nabla^2 F(p) < 0 \wedge F(p) > \alpha\}, \tag{9}$$

where $\Omega \subseteq \mathbb{R}^3$ defines the support domain of F defined by the bounding voxel enclosing the point cloud and α denotes the iso-value chosen for the the initial base mesh extraction.

Depending on the topological characteristics of F the subsets of S_{max} can be associated with surfaces, lines or isolated points. As mentioned above, the function F approximates feature values assigned to the points P within the interval $[0, 1]$ where the value 1 (0) is associated with points assumed to be represent surface (non-surface) related structures. Based on this, the surface related subsets of S_{max} most probably represent the point clouds inherent surfaces. The following procedure is intended to approximate these structures. Using the extracted base mesh, the meta-surface $S_{\max}$ is obtained by iteratively applying the following steps:

1. We move the mesh toward the local maxima of the feature field F by displacing the mesh vertices (cf. Figure 4).
2. In addition, an adaptive strategy for updating the mesh connectivity, to improve the quality of the final representation, is applied.

7.1 Vertex Displacement

Moving the mesh toward S_{max} is achieved by displacing the vertices along the vector field ∇F. Let $\vec{d}$ be the corresponding displacement vector of vertex v, then v is relocated by iteratively moving v along $\vec{d}$ toward S_{max}. Since F approximates the local trend of the feature values assigned to points distributed in 3D space, the maximum values of F most probably occur in regions the corresponding point cloud has high feature values. Hence, projecting the vertices along $\vec{d}$ into these regions is obvious. The new position $\vec{v}_{new}$ of v is defined by

$$\vec{v}_{new} = \vec{v} - \vec{d} \cdot (\vec{d} \cdot (\vec{v} - \vec{c})), \tag{10}$$

where $\vec{v} \in \mathbb{R}^3$ denotes the position of vertex v, and $\vec{c}$ represents the weighted centroid of the proximate point neighborhood $N(v)$ of v determined by

$$\vec{c} = \sum_{p \in N(v)} \frac{f_p}{\sum_{q \in N(v)} f_q} \vec{p}, \tag{11}$$

where f_p denotes the feature value assigned to the point p. The size of the neighborhood $N(v)$ is determined by the radius of the one-star neighborhood of vertex v. The one-star neighborhood of v is given by all vertices that are directly connected to v. Let r be the range to the most distant vertex corresponding to the one-star neighborhood of v, then $N(v)$ is defined to be $N(v) = \{p \in \mathbb{R}^3 | \|v - p\| < r\}$.

7.2 Adapting Mesh Connectivity

This step becomes necessary when triangles corresponding to different mesh regions converge or intersect each other. To obtain the final surface representation we collapse vertices of affected triangles. The collapsing criterion depends on the size of the involved triangles. Collapsing two vertices v and w, not connected, is allowed, if the following condition holds:

$$\|\vec{v} - \vec{w}\| < r_{\min}, \tag{12}$$

where $r = min\{\cdots, r_i, \cdots\}$ with r_i representing the radius of the smallest sphere enclosing the triangle $t_i \in T_v \cup T_w$. The sets T_v and T_w consist of the triangles connected to v and w. If condition (12) is satisfied, the vertices v and w are collapsed and substituted by a new vertex v_{new}. The position of v_{new} is computed by $\vec{v}_{new} = \frac{1}{2}(\vec{v} + \vec{w})$. The collapsing operation takes into account additional rules preventing the mesh to break or to collapse. Additional operations are incorporated maintaining the mesh consistence e.g., by removing duplicated triangles. As a consequence of the vertex collapse, the resulting mesh is no longer guaranteed to be two-manifold.

After repositioning the vertices the resulting mesh might not approximate S_{max} in an optimal way. One possible reason is that the resolution of the mesh in some regions is too coarse. Subdividing selected triangles introduces new vertices at the centers of the triangles. Again, forcing these vertices toward $S_{\max}$ further increases the quality of the surface mesh. The decision whether a triangle is subdivided or not depends on the error measure

$$\varepsilon = \left(\frac{\overline{f} - f_c}{\|\nabla F_c\|} \right)^2, \tag{13}$$

where $\overline{f}$ denotes the average feature value of the vertices of the considered triangle, f_c and ∇F_c represent the feature value and the gradient at the triangles center. We allow subdividing a triangle if $\varepsilon > r^2$, with r being the radius of the smallest sphere enclosing the corresponding triangle. Subdividing triangles introduces new vertices w at the center of the triangles. Those vertices, in a subsequent step, are moved toward S_{max} following the rules of vertex displacement (cf. Section 7.1). On the other hand, triangles are collapsed if they have a bad edge-to-edge aspect ratio.

Since the subsets of S_{max} in some cases represent points or curves, it may happen that the vertex displacement forces the mesh to collapse into point- or a curve-like structures. In this case the corresponding mesh structures are deleted.

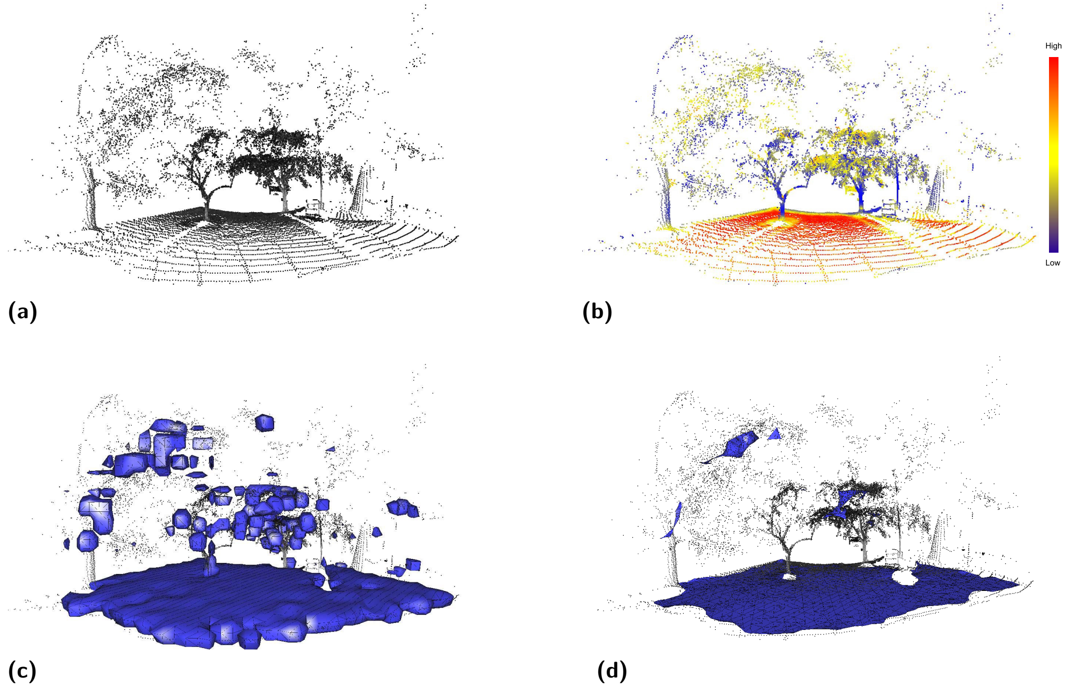

Figure 5 (a) original point cloud showing an environmental scene. (b) feature values, computed by the applied point classification scheme, which are mapped in form of color values to the corresponding points (red indicates surface-related, blue non-surface related points). (c) reconstructed initial mesh, (d) final surface representation.

8 Results

This section provides some results obtained by applying the proposed method to different point data sets. The used data sets are well-suited to demonstrate the ability of the method to reconstruct surfaces from complex and noisy point data sets.

Figure 5a presents the raw input point cloud showing an environmental scene (having about 50,000 points) consisting of quite complex structures. Applying the proposed point classification scheme delivers feature values for each point. The classification was performed using the local point neighborhood $N(p)$ determined by all points lying within a given range r of the considered point p. In the following, r is specified in terms of percentage of the diagonal of the smallest sphere enclosing the entire point cloud. Figure 5b visualizes these values by mapping colors ranging from blue to red to the points. Red colors indicate points in near-planar regions, whereas blue colors indicate points that are not assumed to be part of surfaces. In this example r was chosen to be 0.02. As the image shows, mainly the points at the ground are identified as surface-related structures, but also some isolated point structures at the crowns. This is due to the fact that the local neighborhoods of the isolated points are rather planar and thus the points misleadingly are assumed to lie on a surface. Figure 5c shows the base mesh extracted for the iso-value 0.6. Besides capturing the ground structure the initial mesh also captures the identified structures at the crowns. Figure 5c shows the final mesh after the application of the vertex displacement and the adaption of the mesh connectivity. The ground surface is almost completely recovered, except for a small part located at the tree trunk in the middle. Since the point cloud does not exhibit surface-like

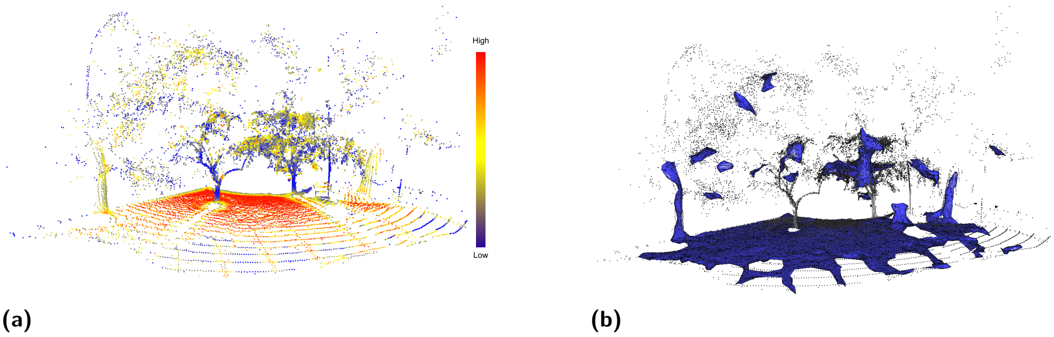

Figure 6 (a) Visualization of the feature values for a chosen neighborhood size $r = 0.016$. (b) the final surface representation.

character at the area at which the trunk merges with the ground, the local point environment rather has cylindrical or spherical character, the affected region is not captured by the final surface.

The reconstruction process strongly depends on the choice of the neighborhood size considered for performing the point classification. Figure 6a demonstrates the effect of choosing the neighborhood size to be to small with respect to the density of the points located at the ground. In this example the radius determining the neighborhood $N(p)$ was chosen to be $r = 0.016$. The generated surface mesh, depicted by Figure 6b, does not capture the entire ground. The points in the front at which the feature values are low (blue colors) are not approximated by the surface. Due to the small neighborhood size chosen the points are not assumed to be part of the ground surface and thus not part of the final surface mesh.

In both examples the chosen parameters are equal, except for r. The B-spline based approximation of the feature values was started at octree level five and terminated at level ten. The Cartesian grid used for the MC-based extraction was determined by partitioning the original bounding voxel into $40 \times 40 \times 40$ cubes.

The next example demonstrates the stability of the reconstruction approach by applying it to a modified point set. Figure 7a shows the point cloud of the Stanford bunny data set where noise is added in form of randomly distributed point samples. In this case, the boundary voxel, considered for adding the additional samples, is centered at the centroid of the point cloud and has the diagonal length of two times the length of the most distant point to the centroid. The number of additional noise samples in this example equals $80,000$ points, the original number of points of the bunny data set is approximately $35,000$. Figure 7b shows the color-coded point cloud indicating the computed feature values. Again, blue represents non-surface like, red represents surface like point structures. The chosen neighborhood size equals $r = 0.01$. It is observable that the classification process identifies most points corresponding to the original point cloud. Applying the approximation scheme followed by the MC mesh extraction delivers the initial base mesh depicted in Figure 7c. The approximation was started at level six and was terminated at octree level ten. Here, the cube grid used for the MC-based mesh extraction was determined by partitioning the original bounding voxel into $80 \times 80 \times 80$ cubes. The final mesh after performing several repositioning and smoothing steps is depicted in Figure 7d. This example shows that the method provides reliable results in cases the point cloud becomes extremely noisy. Several experiments have been performed using an increased number of additional noise samples. The reconstruction provides acceptable results up to $100,000$ noise samples. Theoretically, the reconstruction is

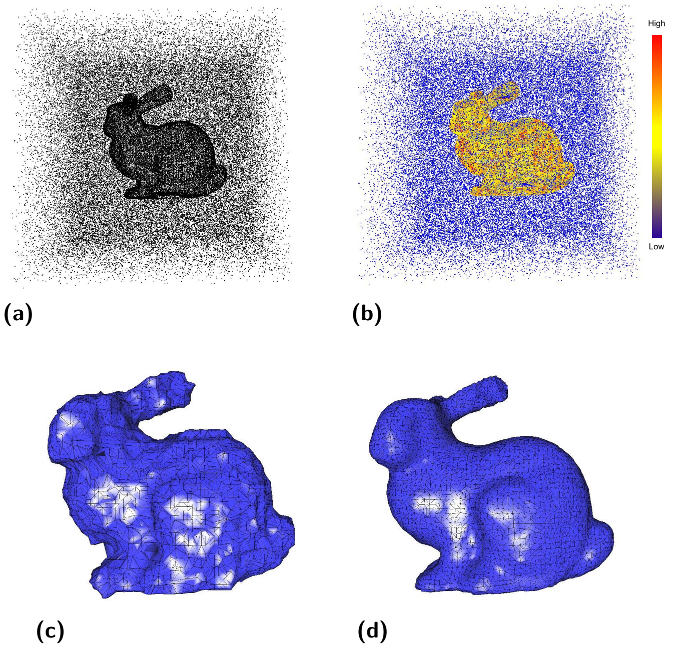

Figure 7 (a)-(d) examples illustrating the surface reconstruction from a point cloud enriched with $80,000$ noise samples. (a) original point cloud containing the Stanford bunny data set consisting of approx. $35,000$ points, (b) point cloud featuring color values indicating the estimated strength of the correspondence of points to surface-related structures (blue indicates low, red indicates high correspondences). (c) the extracted base mesh, (d) the final surface mesh.

possible as long as the density of the points, corresponding to underlying surfaces, locally is higher than the density of the proximate noise samples. However, if the noise samples form surface-related point structures close to the original point samples the reconstruction might fail at these regions.

The average computational complexity of the proposed algorithm is estimated to be $O(k^2 \cdot n \cdot log(n))$, where k is the average number of points within $N(p)$ and n is the number of input data points. In case that $k << n$ the complexity $O(n \cdot log(n))$ rather holds. The complexity of the mesh extraction is estimated to be $O(m \cdot log(n))$ and primarily depends on the number of extracted mesh vertices m and the number of input points. Computationally most expensive are the approximation of the feature values by F and the vertex displacement. Hence, it was decided to perform the determination of feature values and their approximation in a pre-processing step. This allows the user to perform the actual surface reconstruction more efficiently. Table 1 presents the computation times for the results depicted by Figure 5 and Figure 7. The processing was performed on a desktop computer equipped with an Intel Core2 Duo Processor and 4GB main memory.

9 Conclusions

We have presented a novel approach to feature classification and selective surface reconstruction from large scale point clouds. Our method is used to complement visualization of raw data sets obtained from terrestrial and airborne laser scanning. The main problem when visualizing such vast amounts of points is that the user may not recognize structures of interest

Table 1 Computation times for each step of the algorithm concerning the two example data sets. The approximation was performed using four iterations at each processed octree level.

Computation step	Time [sec.] environmental data set (50,000 points)	Time [sec.] modified bunny data set (115,000) points
Computation of feature values	46.95	72.34
Performing approximation	145.11	1107.84
Extracting base mesh	13.2	141.39
Vertex displacement (1st iteration)	7.65 (3,900 vertices)	415.12 (26,000 vertices)

without additional visual cues. Therefore, we classify point neighborhoods into different feature groups by assigning corresponding feature values. These values assigned to surface features are approximated by a hierarchical trivariate B-Spline function which is the basis of our surface reconstruction approach. Topological and geometric detail can be controlled by a user-defined parameter. Future work will be directed at choosing these parameters locally and accelerating the approach further by more sophisticated data structures for streaming and view-dependent querying.

Acknowledgements

This work was supported by the German Research Foundation (DFG) through the International Research Training Group (IRTG) 1131, the University of Kaiserslautern and the Center for Mathematical and Computational Modeling (CM)^2. It was also supported in part by the W.M. Keck Foundation that provided support for the UC Davis Center for Active Visualization in the Earth Sciences (KeckCAVES), Department of Geology. We thank the members of the Visualization and Computer Graphics Research Group at the Institute for Data Analysis and Visualization (IDAV) at UC Davis.

References

1 M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):3—15, 2003.

2 Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Point set surfaces. In *Conference on Visualization*, pages 21–28. IEEE Computer Society, 2001.

3 Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust. In *6th ACM symposium on Solid Modeling and Applications*, pages 249–266. ACM Press, 2001.

4 M. Bertram, X. Tricoche, and H. Hagen. Adaptive smooth scattered-data approximation for large-scale terrain visualization. In *Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization*, pages 177–184, 2003.

5 J. Bloomenthal, C. Bajaj, J. Blinn, M.-P Cani-Gascuel, A. Rockwood, B. Wyvill, and G. Wyvill. *Introduction to implicit surfaces.* Morgan KAufman, Inc., San Francisco, 1997.

6 J.C. Carr, R.K. Beatson, J.B. Cherry, T.J. Mitchel, W.R. Fright, B.C. McCallum, and T.R. Evans. Smooth surface reconstruction from noisy range data. In *Proc. SIGGRAPH Computer Graphics*, 2001.

7 J.C. Carr, R.K. Beatson, B.C. McCallum, W.R. Fright, T.J. McLennan, and T.J. Mitchel. Smooth surface reconstruction from noisy range data. In *Proc. ACM GRAPHITE*, pages 119–126, 2003.

8 T. K. Dey and S. Goswami. Tight cocone: A water tight surface reconstructor. In *Proc. 8th ACM Sympos. Solid Modeling Appl.*, pages 127–134, 2003.

9 Tamal K. Dey and Jian Sun. An adaptive mls surface for reconstruction with guarantees. In *3rd Eurographics Symposium on Geometry Processing*, pages 43–52, 2005.

10 M. Eck and H. Hoppe. Automatic reconstruction of b-spline surfaces of arbitrary topological type. In *ACM Siggraph*, pages 325–334, 1996.

11 Herbert Edelsbrunner and Ernst P. Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, 13(1):43–72, 1994.

12 S. Fleishman, D. Cohen-Or, and C.T. Silva. Robust moving least-squares fitting with sharp features. In *ACM Siggraph*, pages 544–552, 2005.

13 T.A. Foley. Scattered data interpolation and approximation with error bounds. In *In Computer Aided Geometric Design*, pages 163–177, 1986.

14 R. Franke. Scattered data interpolation: Test of some methods. *In Mathematics of Computation*, 38(157):181–200, 1982.

15 B.F. Gregorsky, B. Hamann, and K.I. Joy. Reconstruction of b-spline surfaces from scattered data points. In *Computer Graphics International 2000*, pages 163–170, 2000.

16 H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *ACM SIGGRAPH '92 Conference Proceedings*, pages 71–78, 1992.

17 P. Keller, M. Bertram, and H. Hagen. Reverse engineering with subdivision surfaces. In *Computing 2007*, pages 127–134, 2007.

18 Patric Keller, Oliver Kreylos, Marek Vanco, Martin Hering Bertram, Eric S. Cowgill, Louise H. Kellogg, Bernd Hamann, and Hans Hagen. *Extracting and Visualizing Structural Features in Environmental Point Cloud LiDaR Data Sets (to be published).*

19 S. Lee, G. Wolberg, and S.Y. Shin. Scattered data interpolation with multilevel b-splines. *IEEE - Transactions on Visualization and Computer Graphics*, 3(3):228–244, 1997.

20 S.K. Lodha and R. Franke. Scattered data techniques for surfaces. In *Proceedings Dagstuhl Conf. Scientific Visualization*, pages 182–222, 1999.

21 William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics*, 21(4):163–169, 1987.

22 B. Mederos, L. Velho, and L. H. de Figueiredo. Moving least squares multiresolution surface approximation. In *Sibgraphi*, 2003.

23 L. V. Boris Mederos, N. Amenta, L. Velho, and L.H. de Figueiredo. Surface reconstruction from noisy point clouds. In *Eurographics Symposium on Geometry Processing*, 2005.

24 S. Muraki. Volumetric shape description of range data using "blobby model". In *ACM SIGGRAPH Proceedings in Computer Graphics*, pages 227–235. ACM Press, 1991.

25 A. Nealen. An as-short-as-possible introduction to the least squares, weighted least squares and moving least squares methods for scattered data approximation and interpolation. In *Technical Report, TU Darmstadt*, 2004.

26 Y. Othake, A. Belyaev, and H.-P. Seidel. A multi-scale approach to 3d scattered data interpolation with compactly supported basis functions. In *Shape Modeling Interantional 2003*, pages 153–161, 2003.

27 M.J.D. Powell. Radial basis functions for multivariate interpolation. In *Algorithms for Approximation of Functions and Data*, pages 143–168, 1987.

28 L.L. Schumaker. Fitting surfaces to scattered data. In *Approximation Theory II*, pages 203–268, 1976.

29 C. Shen, J.F. O'Brian, and J.R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. In *In Proc. of ACM SIGGRAPH 2004*, pages 896–904, 2004.

30 Greg Turk and James F. O'Brien. Shape transformation using variational implicit functions. In *The Proceedings of ACM SIGGRAPH 99*, pages 335–342, 1999.

31 R. Whitaker. A level-set approach to 3d reconstruction from range data. *International Journal of Computer Vision*, 29(3):203–231, 1998.

32 W. Zhang, Z. Tang, and J. Li. Adaptive hierarchical b-spline surface approximation of large-scale scattered data. In *In Proc. Pacific Graphics '98*, pages 8–16, 1998.

33 Hong-Kai Zhao, S. Osher, and R. Fedkiw. Fast surface reconstruction using the level set method. In *IEEE Proceedings in Variational and Level Set Methods in Computer Vision*, pages 194–201. IEEE Computer Society, 2001.