Eurographics/ IEEE-VGTC Symposium on Visualization (2006)
Thomas Ertl, Ken Joy, and Beatriz Santos (Editors)

Efficient Surface Reconstruction from Noisy Data using Regularized Membrane Potentials

A. C. Jalba and J. B. T. M. Roerdink

Institute for Mathematics and Computing Science, University of Groningen, The Netherlands

Abstract

We present a novel, physically-motivated method for surface reconstruction that can recover smooth surfaces from noisy and sparse data sets, without using orientation information. A new volumetric technique based on regularized-membrane potentials for aggregating the input sample points is introduced, which manages improved noise tolerability and outlier removal, without sacrificing much with respect to detail (feature) recovery. In this method, sample points are first aggregated on a volumetric grid. A labeling algorithm that relies on intrinsic properties of the smooth scalar field emerging after aggregation is used to classify grid points as exterior or interior to the surface. We also introduce a mesh-smoothing paradigm based on a mass-spring system, enhanced with a bending-energy minimizing term to ensure that the final triangulated surface is smoother than piecewise linear. The method compares favorably with respect to previous approaches in terms of speed and flexibility.

Categories and Subject Descriptors (according to ACM CCS): G.1.2 [Numerical Analysis]: Approximation - Approximation of surfaces and contours; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling - Physically based modeling; I.4.10 [Image Processing and Computer Vision]: Image Representation - Volumetric;

1. Introduction

The goal of surface reconstruction is to obtain a digital representation of a real, physical object or phenomenon described by a cloud of points, sampled on or near its surface. Recently there has been a growing interest in this field motivated by the increased availability of point-cloud data obtained from medical scanners, laser scanners, vision techniques (*e.g.* range images), and other modalities. Apart from being ill-posed, the problem of surface reconstruction from unorganized point clouds is challenging because the topology of the real surface can be very complex, and the acquired data may be non-uniformly sampled and contaminated by noise. Moreover, the quality and accuracy of the data sets depend upon the methodologies which have been employed for acquisition (*i.e.* laser scanners versus stereo using uncalibrated cameras). Furthermore, reconstructing surfaces from large datasets can be prohibitively expensive in terms of computations.

In this paper, we propose a novel, physically-based technique for surface reconstruction, which employs regularized-membrane potentials, evaluated on a volumetric grid, to output smooth surfaces from noisy and sparse data. The purpose of these potentials is twofold: to aggregate data points and to remove outliers due to noise. In the following we denote by *aggregation* the process in which gaps between the data points are bridged by a slowly-varying scalar field.

The contributions of this paper include:

- a new method for aggregating the input data points, based on regularized-membrane potentials, as an alternative approach to the widely employed distance transform (Section 3.1)
- a new approach to surface smoothing (Section 3.3) based on a mass-spring system enhanced with a bending-energy minimizing term.

The main advantage of our approach is that the robustness of the proposed surface-reconstruction method with respect to noise is improved. Furthermore, minimizing the bending energy of the surface ensures that the final triangulated surface is smooth, *i.e.*, smoother than piecewise linear. Our formulation handles noisy as well as non-uniform data sets, and works in any dimension. In particular, we show that the proposed method quickly reconstructs surfaces of large models and tolerates large amounts of Gaussian and shot noise.

2. Previous and related work

Most popular approaches for surface reconstruction are based on implicit surfaces or volumetric representations. The traditional approach is to compute a signed distance function and represent the reconstructed implicit surface by an iso-contour (usually at iso-value zero) of this function [BC00, HDD*92, CL96, LTGS95, BBX95]. These methods require a way to distinguish between the inside and outside of closed surfaces. For example, the method of Hoppe *et al.* [HDD*92] approximates the normal at each data point by fitting a tangent plane in its neighbourhood, using principal component analysis. Tang and Medioni [TM98] use the tensor-voting formalism to estimate the orientations of the data points. Both methods are sensitive to noise since they require accurate normal estimates. Zhao *et al.* [ZOF01] use the level-set formalism for noise-free surface reconstruction. Their method can handle complicated surface topology and deformations, and the reconstructed surface is smoother than piecewise linear. The main drawback is the sensitivity of the method to shot noise, due to its reliance on the distance transform.

More recently, modeling of surfaces with Radial Basis Functions (RBFs) has become a popular technique [CBC*01, MYR*01, KSH04, DTS02, TO99]. Turk and O'Brien [TO99] and Carr *et al.* [CBC*01] use globally supported RBFs to fit data points by solving a large and dense linear system of equations. These methods are very sensitive to noise because local changes of the positions of the input points have global effects on the reconstructed surface. Morse *et al.* [MYR*01] and Ohtake *et al.* [OBS03] use compactly-supported RBFs to achieve local control and reduce the computational cost by solving a sparse linear system. Dinh *et al.* [DTS02] use RBFs and volumetric regularization to handle noisy and sparse range data sets. Recently, Ohtake *et al.* [OBA*03] proposed a method based on the so-called "partition of the unity implicits", which can be regarded as the combination of algebraic patches and RBFs. Carr *et al.* [CBM*03] further address reconstruction from noisy range data by fitting a RBF to the data and convolving with a smoothing kernel during the evaluation of the RBF. Kojekine *et al.* [KSH04] use compactly-supported RBFs and an octree data structure such that the resulting matrix of the system is band-diagonal, thus reducing computational costs. The main advantages of implicit surface representations include topological flexibility, mesh independent representation (*i.e.* the possibility to generate a mesh on demand, *e.g.* for visualization purposes) and compact representation to within any desired precision. Moreover, efficient algorithms for polygonization of implicit surfaces are available [LC87, NB93, Blo94].

3. The proposed method

The computational flow diagram of our method is shown in Fig. 1. The method starts by assigning the input sample

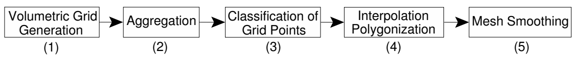
Volumetric Grid Generation
(1)
Aggregation
(2)
Classification of Grid Points
(3)
Interpolation Polygonization
(4)
Mesh Smoothing
(5)

Figure 1: *Algorithmic flow diagram of the proposed method.*

points to grid cells, using cloud-in-cell (CIC) interpolation (first step in Fig. 1). Step 2 performs aggregation of the sample points by computing regularized-membrane potentials on the grid. A labeling algorithm, which follows increasing paths of the scalar potential field is engaged to classify the grid points as exterior or interior to the surface, defining in this way an implicit (rough) surface (step 3). Prior to polygonization, we use again diffusion potentials, but this time with the purpose of producing a smooth implicit surface. Then, we employ Bloomenthal's polygonizer [Blo94] to turn the implicit surface into a triangulated one (second part of step 4), and use a mass-spring system, enhanced with a bending-energy minimizing term, to obtain a larger degree of smoothness (step 5).

3.1. Aggregation of the input data points

The first step of our method consists of assigning the input data points to cells of a three-dimensional grid, using CIC interpolation. Accordingly, a constant numerical value (we fix this value to one), representing the contribution of each data point to the initial distribution, is spread to the eight nearest cell centers. The weights are given by the overlap volumes of a box, centered around the data point under consideration, with the neighbouring voxels. If more than one point contributes to the same cell, the values are accumulated. The non-empty grid cells will serve as sources generating potentials on the grid (see below).

The non-empty grid cells, called *source points*, can be regarded as sources for the physical simulation of the flow of heat, defined by the linear-diffusion equation. However, aggregation using linear diffusion has the disadvantage that it converges to constant steady state. This issue can be addressed by supplementing the diffusion equation with a reaction term, leading to the *regularized membrane equation* (see also [SS98] and [Ter86]),

$$\frac{\partial u}{\partial t} = \nabla^2 u + \beta(f - u), \qquad (1)$$

where β is a regularization parameter, u is the concentration of diffusing material, with the original volume f as initial condition, $u(t = 0) = f$. The last term in Eq. (1) ensures that the reaction-diffusion equation reaches a steady state not far from the original values of f. However, the problem of choosing a proper stopping time for the linear diffusion is shifted to finding a suitable value for the parameter β. To alleviate this problem we have chosen the value of β equal to the absolute value of the original signal f at each voxel location x, yielding

$$\frac{\partial u}{\partial t} = \mu \nabla^2 u + |f|\,(f-u), \tag{2}$$

where μ is a regularization constant which controls the amount of smoothing. Note that in Eq. (2) we have used $\beta = |f|$ so that we can handle also negative values of f; this will be necessary when performing interpolation in Section 3.3, see Eq. (3). Also, the choice of μ is not critical, as we show in Section 4.

One can use the method of eigenfunction expansion [Hab97] to find approximate solutions to Eq. (2). The approximate steady-state solution of this equation linearly interpolates the data points, whereas transient solutions are equivalent to Gaussian interpolants in space which decay exponentially in time.

3.2. Classification of grid cells

After aggregation, source cells correspond to regional maxima of the scalar field u, provided that a transient solution of Eq. (2) has been (numerically) computed. In addition, *ridges* of this field match shortest paths connecting nearby source cells. It is this property which can be used to classify the grid points as exterior or interior to the surface by an algorithm similar to the tagging method of Zhao *et al.* [ZOF01]. Since the only modification of this tagging algorithm consists in replacing the *maximum heap* with a minimum one, we refer to [ZOF01] for further details.

3.3. Surface smoothing and polygonization

Once the classification of the grid points into interior, boundary and exterior has been completed, one can use Bloomenthal's method [Blo94] to polygonize the implicit surface given by the zero level set of the scalar field f defined by

$$f(x,y,z) = \begin{cases} -1 & \text{if (x,y,z) is labeled as } \texttt{INTERIOR} \\ 0 & \text{if (x,y,z) is labeled as } \texttt{BOUNDARY} \\ 1 & \text{if (x,y,z) is labeled as } \texttt{EXTERIOR}. \end{cases} \tag{3}$$

3.3.1. Interpolation using membrane potentials

Direct polygonization will cause "staircase" artefacts in the resulting mesh. A better approach is to interpolate the implicit surface using the reaction-diffusion process (Eq. (2)) a second time with the labeled grid points as sources. Note that, as given in Eq. (3), sources are instantiated only at the locations of the interior and exterior grid points, since the membership of boundary points is uncertain. After the potentials have been computed, a smooth scalar field emerges at these locations, and by tracing its zero iso-contour, the implicit surface is turned into a triangulated one. Since boundary voxels form thin bands along surface borders, a small number of iterations is required, resulting in fast computation. The resulting triangulated surface, which is a better approximation to the real surface than the initial one, is used as initialization for the more computationally demanding mass-spring system, described next.

3.3.2. Mesh smoothing with a mass-spring system

Since for now we assume that the correct surface topology has been inferred, and the triangulated surface possesses consistent orientation (see Section 3.4 for a justification), we propose to use a mass-spring system as a means for obtaining a larger degree of smoothness. Also, we shall integrate in our mass-spring system an extra term corresponding to the *bending energy* of the system. This has the beneficial effect, analogous to curvature flow, that the triangulated surface is smoothened by moving its vertices along their normals with a speed proportional to the (normal) curvature.

We start by defining nodes p_i, $i = 1,2,\ldots,N$, of the mass-spring network, as having mass m_i and position vector $\mathbf{x}_i(t) = [x_i(t), y_i(t), z_i(t)]$. We will denote by $\mathcal{N}_i$ the set of neighbours of p_i, *i.e.*, all particles p_j such that there exists an edge e_{ij} between p_i and p_j. Let spring s_{ij} connect nodes p_i and p_j, have rest length l_{ij} and stiffness $c_{ij} = c$, where c is a constant. Also, let $\mathbf{r}_{ij} = \mathbf{x}_j - \mathbf{x}_i$ be the vector separating the two nodes. The potential energy of a particle p_i of the mass-spring system due to its interactions with neighbouring particles p_j, $j \in \mathcal{N}_i$ is given by

$$E_i = \sum_{j \in \mathcal{N}_i} \alpha E_{s_{ij}}(\mathbf{r}_{ij}) + (1-\alpha) E_{b_{ij}}(\mathbf{r}_{ij}, \mathbf{n}_i), \tag{4}$$

where the first term represents the energy of the spring connecting the particles, $E_{s_{ij}} = \frac{c}{2}\left(|\mathbf{r}_{ij}| - l_{ij}\right)^2$, the second term is the bending energy (see Eq. (5)), and α is a scalar weight. Smoothing a mesh by minimizing a membrane energy functional [Ter86] can be seen also as the physical simulation of a mass-spring network with zero-rest length springs that will shrink to a single point. On the one hand, because such behaviour of the mass-spring system is undesirable for our purposes, the rest lengths of the springs should be chosen such that they reflect the lengths of the edges of the initial (un-deformed) mesh. On the other hand, in order to facilitate the relaxation of the mesh structure into a desirable, smooth configuration, the rest lengths of the springs should be made smaller than the initial lengths of the edges of the mesh, *i.e.*, we use a percentage of the initial edge lengths.

The bending energy of an ideal, thin flexible plate of elastic material, which is a measure of the strain energy of the deformation, is defined as the sum of squared curvatures along the surface. We modify this definition of bending energy slightly, to restrict it to the neighbourhood of a particle p_i as

$$E_{b_i} \equiv \sum_{j \in \mathcal{N}_i} E_{b_{ij}} = \frac{1}{2} \sum_{j \in \mathcal{N}_i} k_{ij}^2, \tag{5}$$

where k_{ij} is some discrete curvature measure between the particle pair (p_i, p_j). A (mean) curvature estimate, which has been previously used in the context of mesh smoothing [Tau95], is given by $k_{ij} = 2(\mathbf{n}_i \cdot \mathbf{r}_{ij})/|\mathbf{r}_{ij}|^2$, where $\mathbf{n}_i$

is the normal at p_i. Having defined the energy associated with our mass-spring system, we can derive its equations of motion. The variations of particle potentials with respect to positions yield forces acting on particles. We use the Verlet method [Ver67] to integrate the corresponding system of differential equations in time.

3.4. Error analysis of the framework

Let us assume that the grid resolution agrees with the (appropriate) sampling rate of the unknown surface to be reconstructed, *i.e.*, each point of the data set is assigned to a distinct grid cell. Note that in the small-cell-size limit, the CIC interpolation scheme becomes nearest neighbour interpolation. Also, we assume a noise-free input data set. Therefore, if $h_x = bx/gx$, $h_y = by/gy$, $h_z = bz/gz$ are the grid-cell sizes, bx, by, bz are the dimensions of the bounding box of the data points, and gx, gy, gz are the number of cells in the x, y, z directions, then a bound on the reconstruction error is given by the length of the diagonal of a grid cell, *i.e.*, $\varepsilon = \sqrt{hx^2 + hy^2 + hz^2}$. Implicit surface interpolation using Eq. (2) with initial condition given by Eq. (3) cannot increase the reconstruction error since grid cells labeled as interior/exterior maintain their labels (values) due to the similarity term. Moreover, interpolation using Eq. (2) yields a smooth field at boundary locations, which can only decrease the reconstruction error, though the error bound remains the same. After interpolation, the gradient field has correct orientation, without singular points at boundary locations, and therefore the reconstructed surface is *consistently oriented*. Also, when the grid resolution is large enough, any of the surfaces of the interior/exterior layers has the *same topology* as the unknown surface, and therefore, the reconstructed surface has the same topology.

In the presence of noise, surface features smaller than the noise amplitude in the data set cannot obviously be recovered. However, as we show in Section 4, the method is noise tolerant, albeit the error bound will also increase up to $\varepsilon_r = \eta + \sqrt{hx^2 + hy^2 + hz^2}$, where η denotes the standard deviation of the noise in the data. Note that these error bounds remain the same even if the mapping of data points to non-empty grid cells is not one-to-one.

The mass-spring system may potentially increase the overall reconstruction error at the cost of obtaining smooth surfaces. Therefore, our mass-spring system should preserve the features of the triangulated surface.

4. Results

4.1. Large data sets

In the first experiment, we present surface reconstruction results on large data sets. The parameters of the method were set as follows. For aggregation, we discretized Eq. (2) using central difference approximations and solved it using the explicit Euler method for numerical integration. To guarantee

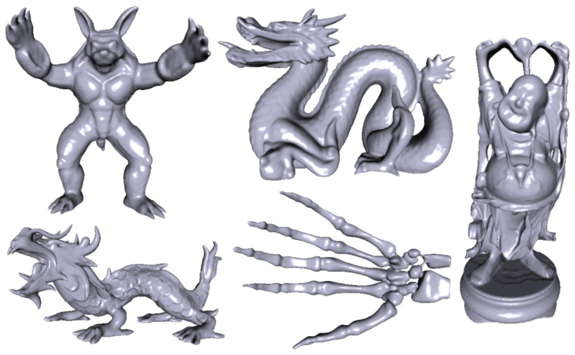

Figure 2: *Reconstruction of large models, see Table 1.*

stability, the time step parameter Δt must obey $\Delta t \leq \frac{\Delta x \Delta y \Delta z}{6\mu}$; we set $\mu = 1.0$ and $\Delta t = 0.16$; we used $N_m = 20$ iterations. The parameters of the Verlet integrator used by the mass-spring system were $\mathrm{dt} = 0.1$ and $t = 10$. The weight in Eq. (4) was set to $\alpha = 0.1$, to emphasize the bending-energy minimizing term. As discussed in section 3.3.2, to facilitate the relaxation of the mesh structure into a smooth configuration, the rest lengths of the springs were set to 90% of the initial edge lengths. Finally, the largest dimension of the computational grid was set to 400 and the remaining two dimensions were obtained by uniform scaling of the bounding box of the sample points. Below we use the same values of the parameters (unless stated otherwise).

The meshes resulting from this experiment are shown in Fig. 2. All computations were performed on a system with a Pentium IV processor at 3.0 GHz and a GeForce FX 5900 Ultra GPU. Timing (measured in seconds) of each step of the method, for the models shown in Fig. 2, is given in Table 1. The most expensive computations are the labeling algorithm and the second stage of smoothing implemented by the enhanced mass-spring system. Note, however, that the computations have the same order of magnitude, and that the computational cost generally depends exclusively on the grid size (*e.g.* compare the timing for the Asian Dragon model with that for the Armadillo model). The time taken to reconstruct nicely either of the models Happy Buddha, Dragon, Hand or Asian Dragon (see Fig. 2) is well under one minute, whereas the Armadillo model needs about one minute. Table 1 also provides some statistics associated with these models. The sixth column of Table 1 shows the approximation error – an estimate about the quality of the reconstruction. The approximation error is an upper bound for the average distance from the data points to the surface, and it is computed as the average distance from the data points to the centers of mass of the mesh triangles. The error is given in percentages of the diagonal of the bounding box of the data points.

Table 1: *Statistics, reconstruction quality and timings obtained using the membrane potential for aggregation.*

Model	No. Points	Grid	No. Vertices	No. Triangles	Error (%)	Aggregation Time (s)	Marching Time (s)	Interpolation Time (s)	Smoothing Time (s)	Total Time (s)
Buddha	543,625	170x400x170	297,058	594,176	**0.08**	3.3	6.9	3.4	13.3	**27.2**
Armadillo	172,974	337x400x307	370,718	741,432	**0.05**	8.1	29.5	7.7	15.6	**61.7**
Dragon	433,375	400x284x184	383,674	767,352	**0.08**	5.1	13.4	4.9	17.7	**42.0**
Hand	327,323	400x283x143	190,654	381,340	**0.06**	4.1	11.5	4.3	8.1	**28.2**
Asian Dragon	3,609,600	400x226x269	206,140	412,280	**0.06**	6.1	18.5	6.2	12.6	**44.5**

4.2. Coping with noise

Our second experiment concerns the behaviour of the method under noise conditions. Additionally, we study how the method copes with sparse data obtained by random sub-sampling.

4.2.1. Shot noise

We changed a certain amount of empty voxels by assigning them the value one, *i.e.*, the same numeric value used to assign the input points. The number of corrupted voxels is expressed as a percentage of the number of *source points*. We used nearest-neighbour interpolation for grid assignment, as this results in a binary volume and represents a fair experimental setting, without a-priori information.

The results are shown in Fig. 3; the number of source points was $3,337$. For the computation of the membrane potential, the number of iterations N_m was increased from 20 to 200. The reason is that a large number of iterations results in a large aggregation support covering most of the exterior volume around the object, which will be correctly labeled as exterior. Note that the method is able to reconstruct the surface of the cactus shown in Fig. 3 even when as much as 40% of the source points ($1,335$ voxels) were corrupted by noise.

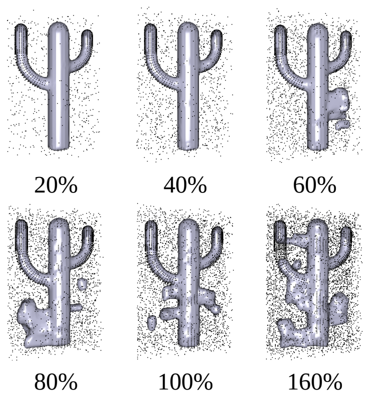

Figure 3: *Shot noise. The number of corrupted voxels is expressed as a percentage of the number of source points.*

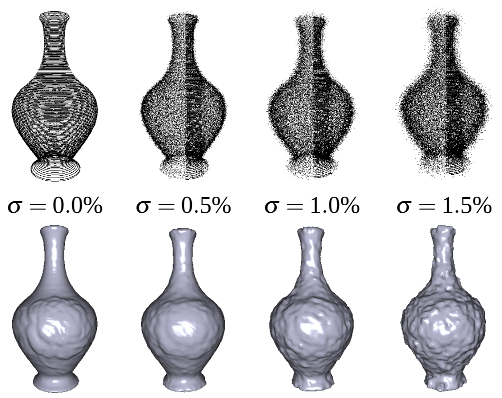

Figure 4: *Gaussian noise with zero mean and standard deviation σ (expressed as a percentage of diagonal size of the bounding box);* first row*: source points,* second row*: reconstructed surfaces.*

4.2.2. Gaussian noise

The experimental setup consisted of perturbing the input points with Gaussian noise with zero mean and standard deviations $\sigma = 0.5, 1.0, 1.5(\%)$, expressed as percentages of the length of the diagonal of the bounding box. The results are shown in Fig. 4. The grid size was $140 \times 146 \times 250$. The parameters of the method were set as in the previous section, except that the stopping time t of the mass-spring system was increased from 10 to 20. Unlike methods which rely on distance transforms, our method can cope with large amounts of Gaussian noise. In fact, in the third case ($\sigma = 1.0\%$) shown in Fig. 4, one percent of the diagonal of the bounding box means that $\sigma = 3.2$, which implies that the coordinates of most points were randomly translated in the interval $[-9.6; 9.6]$. Yet, even in these cases the method is able to output smooth surfaces, with errors bounded by ε_r (results not shown).

4.2.3. Random sampling

Our last analysis studied the behaviour of the method with sparse data sets obtained by randomly sub-sampling the source points. Here, we have used a large grid ($198 \times 143 \times 650$), such that the number of source points ($136, 138$) is comparable to the number of input points ($140, 734$). Then, keeping the grid resolution constant, we randomly sampled

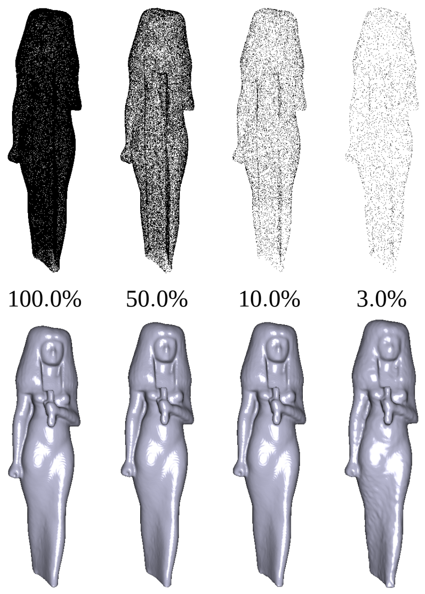

Figure 5: *Random sampling.* First row*: sampled points as a percentage of the number of source points;* second row*: reconstructed surfaces.*

subsets from the set of source points and performed reconstruction using only the sampled points; nearest-neighbour interpolation was used for grid assignment, and the parameters were set as in the previous section. Fig. 5 shows the results. Note that the method yields a very good result even with only 10% of the source points.

4.3. Comparison to other methods

4.3.1. Mesh smoothing

We compared our mesh-smoothing method based on mass-spring systems (see subsection 3.3.2) with curvature flow [MDSB03, DMSB99], and with the non-iterative, feature-preserving method of Jones *et al.* [JDD03]. The results are shown in Fig. 6. The stopping time for the iterative methods (*i.e.* curvature flow and mass-spring system methods) was set to $t = 300$, whereas the parameters of the non-iterative method were set to $\sigma_f = 2$, $\sigma_g = 10$ (smooth large features), which yielded the best result. Note that the non-iterative method preserved too many mesh details, whereas, at the other extreme, curvature flow smeared out even large mesh features. Our method seems to offer a better tradeoff between mesh smoothness and preservation of features. In addition, it can be efficiently implemented on GPU hardware, unlike the non-iterative method.

4.3.2. Surface reconstruction

We compared the proposed framework for surface reconstruction to that by Hoppe *et al.* [HDD*92] and to the Power

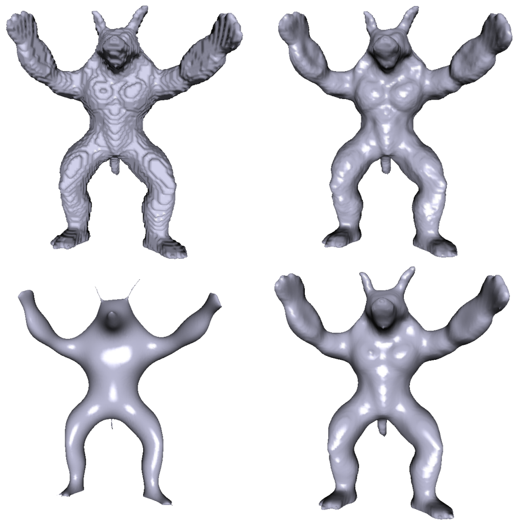

Figure 6: *Mesh smoothing comparison.* Left-to-right, top-to-bottom*: original mesh, method of Jones* et al. *[JDD03], curvature flow [MDSB03, DMSB99], our method.*

Crust algorithm by Amenta *et al.* [ABE98, ABK98], see Fig. 7. The time taken by our method to reconstruct the surface of the model shown in Fig. 7 was 56 seconds on a grid with dimensions $450 \times 320 \times 206$ (the largest which we could use on GPU hardware); the reconstruction error was 0.06. It took 4 minutes for the method by Hoppe *et al.* to reconstruct the same model. In this case some holes are visible in the triangulated surface, since we maximally increased the parameter controlling the sampling of the unknown surface, in an attempt to reconstruct fine surface details. Although the reconstructed surface is smooth, fine surface details are lost. This method can tolerate Gaussian noise provided that each sample point has on average the same distance to its neighbours. However, the method does not tolerate shot noise (results not shown).

The highest resolution of the reconstructed surfaces is obtained by methods which interpolate the data points, similar to the Power Crust algorithm, see Fig. 7. However, the time taken to reconstruct large models (see Table 1) within floating-point precision is one order of magnitude larger than that of our method. Since this method interpolates the data points, it cannot cope with either type of noise which we considered.

We did not perform a direct comparison with the volumetric method of Zhao *et al.* [ZOF01]; refer to [ZOF01] for several results on the same data sets as we used. Although they used lower grid resolutions, the timings required by their level-set method is in the order of hours.

One of the fastest techniques for surface reconstruction is that of Ohtake *et al.* [OBA*03]. Comparing the result

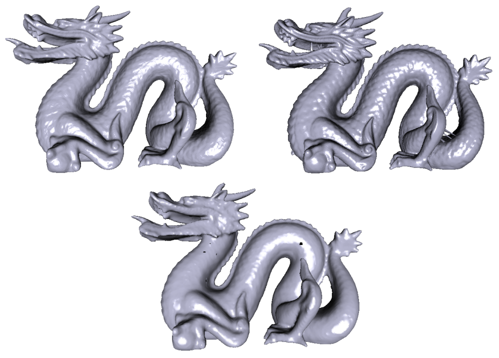

Figure 7: *Surface reconstruction comparison.* Top-left*: reconstruction by the proposed method;* Top-right*: Power Crust algorithm [ABE98, ABK98];* Bottom*: the method of Hoppe* et al. *[HDD*92].*

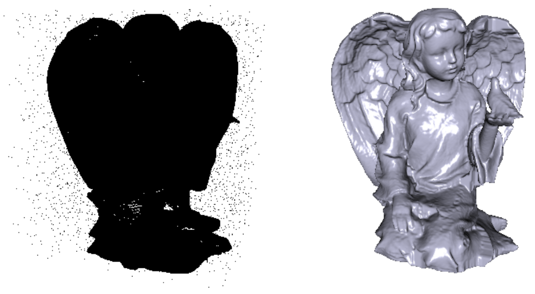

Figure 8: Left*: noisy data set with* 2,008,414 *input points and* 4000 *outliers;* right*: reconstructed surface. Computation time was* 234 *seconds.*

from Table 1 on the Dragon data set with that from Table II in [OBA*03] one can observe that our method is 2.3 times faster, at the same accuracy (8.0×10^{-4}). However, if very large accuracies are needed, their method may become more efficient. In addition, their method assumes that *accurate* normal estimates are available.

Among the few methods which can tolerate a large amount of outliers is the recent one by Kolluri *et al.* [KSO04]. The CPU time reported in [KSO04] is one order of magnitude larger than that of our method, on the same input set (compare Fig. 1 in [KSO04] to Fig. 8).

4.4. Reconstruction error

To verify the error bound of Section 3.4, we studied the behaviour of the reconstruction error (cf. Table 1) when grid resolution increases, see Table 2. As can be seen, the reconstruction error is always bounded by ε_r, even when the mass-spring system is used for mesh smoothing. Only at small grid resolutions does the reconstruction error increase when mesh smoothing is applied. This happens because our mesh smoother preserves small features of the triangulated surface (see subsection 4.3.1).

Table 2: *Grid resolution vs. reconstruction error with/without mesh smoothing; results using the Buddha model.*

Grid	Error (%) w/o Smoothing		Error bound, ε_r (%)	Total time (s) w/o Smoothing	
$67 \times 150 \times 67$	0.4	0.3	0.8	3.0	1.6
$108 \times 250 \times 108$	0.1	0.1	0.5	10.9	5.7
$190 \times 450 \times 190$	0.06	0.06	0.3	53.0	34.0
$272 \times 650 \times 273$	0.03	0.03	0.2	140.8	104.4
$354 \times 850 \times 355$	0.008	0.008	0.1	295.4	218.7

5. Limitations

Surface features smaller than the grid size are not appropriately reconstructed. A possible solution would be to increase the grid resolution at the expense of larger computational time and memory requirements. The method is not geometrically adaptive, but we are currently investigating an adaptive, multi-resolution approach based on data-structures similar to octrees, which can also be efficiently implemented on GPUs. As is usual for methods that employ implicit surface representations, we assume that the surfaces to be reconstructed are closed, though the method does intrinsically perform hole filling by minimal surfaces (results not shown).

6. Conclusions

We have introduced a novel framework for surface reconstruction starting from unorganized point clouds, and demonstrated its effectiveness in several experimental settings. The method can be used to efficiently reconstruct surfaces from clean as well as noisy data sets, and in our opinion, this represents an advantage over existing methods. The method can deliver multi-resolution representations of the reconstructed surface, and can be used to perform reconstruction starting from particle systems, contours or even grey-scale volumetric data leading to image segmentation. Most constituent parts of the method have already been implemented on GPU hardware, but due to lack of space, we will report on these topics elsewhere.

References

[ABE98] AMENTA N., BERN M., EPPSTEIN D.: The crust and the β-skeleton: Combinatorial curve reconstruction. *Graphical Models and Image Processing 60*, 2 (1998), 125–135. 6, 7

[ABK98] AMENTA N., BERN M., KAMVYSSELIS M.: A new Voronoi-based surface reconstruction algorithm. In *Proc. SIGGRAPH'98* (1998), pp. 415–421. 6, 7

[BBX95] BAJAJ C. L., BERNARDINI F., XU G.: Automatic reconstruction of surfaces and scalar fields from 3D scans. *Computer Graphics 29* (1995), 109–118. 2

[BC00] BOISSONNAT J. D., CAZALS F.: Smooth surface reconstruction via natural neighbour interpolation of distance functions. In *Proceedings of the sixteenth annual symposium on computational geometry* (2000), pp. 223–232. 2

[Blo94] BLOOMENTHAL J.: An implicit surface polygonizer. Academic Press Professional, Inc., San Diego, CA, USA, 1994, pp. 324–349. 2, 3

[CBC*01] CARR J. C., BEATSON R. K., CHERRIE J. B., MITCHELL T. J., FRIGHT W. R., MCCALLUM B. C., EVANS T. R.: Reconstruction and representation of 3D objects with radial basis functions. In *Proc. SIGGRAPH'01* (2001), pp. 67–76. 2

[CBM*03] CARR J., BEATSON R., MCCALLUM B., FRIGHT W., MCLENNAN T., MITCHELL T.: Smooth surface reconstruction from noisy range data. In *Proc. Graphite 2003* (2003), pp. 119–126. 2

[CL96] CURLESS B., LEVOY M.: A volumetric method for building complex models from range images. In *Proc. SIGGRAPH'96* (1996), pp. 303–312. 2

[DMSB99] DESBRUN M., MEYER M., SCHRÖDER P., BARR A. H.: Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proc. SIGGRAPH'99* (New York, NY, USA, 1999), ACM Press/Addison-Wesley Publishing Co., pp. 317–324. 6

[DTS02] DINH H. Q., TURK G., SLABAUGH G.: Reconstructing surfaces by volumetric regularization using radial basis functions. *IEEE Trans. Pattern Anal. Machine Intell.* (2002), 1358–1371. 2

[Hab97] HABERMAN R.: *Elementary Applied Partial Differential Equations: With Fourier Series and Boundary Value Problems*. Prentice Hall, 1997. 3

[HDD*92] HOPPE H., DEROSE T., DUCHAMP T., MCDONALD J., STUETZLE W.: Surface reconstruction from unorganized points. In *Proc. SIGGRAPH'92* (1992), pp. 71–78. 2, 6, 7

[JDD03] JONES T. R., DURAND F., DESBRUN M.: Non-iterative, feature-preserving mesh smoothing. *ACM Trans. Graph. 22*, 3 (2003), 943–949. 6

[KSH04] KOJEKINE N., SAVCHENKO V., HAGIWARA I.: Surface reconstruction based on compactly supported radial basis functions. In *Geometric modeling: techniques, applications, systems and tools*. Kluwer Academic Publishers, 2004, pp. 218–231. 2

[KSO04] KOLLURI R., SHEWCHUK J., O'BRIEN J.: Spectral surface reconstruction from noisy point clouds. In *Symposium on Geometry Processing* (July 2004), ACM Press, pp. 11–21. 7

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3D surface construction algorithm. In *Proc. SIGGRAPH'87* (1987), pp. 1631–169. 2

[LTGS95] LIM C. T., TURKIYYAH G. M., GANTER M. A., STORTI D. W.: Implicit reconstruction of solids from cloud point sets. In *Proceedings of the third ACM symposium on Solid modeling and applications* (1995), pp. 393–402. 2

[MDSB03] MEYER M., DESBRUN M., SCHRÖDER P., BARR A. H.: Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, Hege H.-C., Polthier K., (Eds.). Springer-Verlag, Heidelberg, 2003, pp. 35–57. 6

[MYR*01] MORSE B. S., YOO T. S., RHEINGANS P., CHEN D. T., SUBRAMANIAN K. R.: Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. In *Shape Modeling International* (2001), pp. 89–98. 2

[NB93] NING P., BLOOMENTHAL J.: An evaluation of implicit surface tilers. *IEEE Comp. Graphics and Appl. 13* (1993), 33–41. 2

[OBA*03] OHTAKE Y., BELYAEV A., ALEXA M., TURK G., SEIDEL H.: Multi-level partition of unity implicits. In *Proc. SIGGRAPH'03* (2003), pp. 463–470. 2, 6, 7

[OBS03] OHTAKE Y., BELYAEV A., SEIDEL H.: Multi-scale approach to 3D scattered data interpolation with compactly supported basis functions. In *Proc. of Shape Modeling International* (2003), pp. 153–164. 2

[SS98] SCHARSTEIN D., SZELISKI R.: Stereo matching with nonlinear diffusion. *International Journal of Computer Vision 28* (1998), 155–174. 2

[Tau95] TAUBIN G.: Estimating the tensor of curvature of a surface from a polyhedral approximation. In *Proc. ICCV'95* (1995), IEEE Computer Society, pp. 902–907. 3

[Ter86] TERZOPOULOS D.: Regularisation of inverse visual problems involving discontinuites. *IEEE Trans. Pattern Anal. Machine Intell. 8* (1986), 413–424. 2, 3

[TM98] TANG C. K., MEDIONI G.: Inference of integrated surface, curve, and junction descriptions from sparse 3-D data. *IEEE Trans. Pattern Anal. Machine Intell. 20* (1998), 1206–1223. 2

[TO99] TURK G., O'BRIEN J.: *Variational Implicit Surfaces*. Tech. rep., Georgia Institute of Technology, 1999. 2

[Ver67] VERLET L.: Computer experiments on classical fluids I. thermodynamical properties of Lennard-Jones molecules. *Phys. Rev. 159* (1967), 98–103. 4

[ZOF01] ZHAO H., OSHER S., FEDKIW R.: Fast surface reconstruction using the level set method. In *Proceedings of the IEEE Workshop on Variational and Level Set Methods in Computer Vision* (2001), pp. 194–202. 2, 3, 6