Multigrid Narrow Band Surface Reconstruction via Level Set Functions

J. Ye[1], I. Yanovsky[1,2], B. Dong[3], R. Gandlin[4],
A. Brandt[1,5], S. Osher[1]

[1] Department of Mathematics, University of California, Los Angeles, CA, USA
[2] Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA
[3] Department of Mathematics, University of California, San Diego, CA, USA
[4] Department of Mathematics, Carnegie Mellon University, Pittsburgh, PA, USA
[5] Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel.

Email: ye@math.ucla.edu, igor.yanovsky@jpl.nasa.gov, b1dong@math.ucsd.edu, rima121212@gmail.com, abrandt@math.ucla.edu, sjo@math.ucla.edu

Abstract. In this paper we propose a new method for implicit surface reconstruction from unorganized point clouds. Our algorithm employs a multigrid solver on a narrow band, which greatly improves the computational efficiency of surface reconstruction process. The new model can reconstruct surfaces from noisy unorganized point clouds that also have missing information. Comparing to traditional methods, our method is significantly faster, generating surfaces that have detailed information and preserved sharp features.

1 Introduction

The field of surface reconstruction from scattered point clouds has been developing rapidly in the past few years. It is a challenging problem since point clouds lack ordering information and connectivity, and are often very noisy. There are two ways of representing the reconstructed surfaces: explicit and implicit. Explicit representation usually gives the exact location of a surface in a physical domain, while implicit representation defines the surface as the zero level set of some scalar function. Common explicit representations include parametric surfaces [15, 14] and triangulated surfaces [1–3, 6, 7]. Implicit surfaces are most frequently represented using level set functions, typically signed distance functions [13], and some recent ones [12, 16].

One of the traditional approaches for implicit surface reconstruction is via the use of radial basis functions representation [5]: $s(\boldsymbol{x}) = p(\boldsymbol{x}) + \sum_{i=1}^{n} \lambda_i \phi(|\boldsymbol{x} - \boldsymbol{x_i}|)$, where p is a polynomial, ϕ is a global smooth function that allows fast summation, e.g. $\phi(\boldsymbol{r}) = \boldsymbol{r}$, and $(\boldsymbol{x_1}, ..., \boldsymbol{x_n})$ is a set that includes the N given surface points and a comparable number of off-surface points at each of which $s(\boldsymbol{x_i})$ is prescribed as an estimated distance from $\boldsymbol{x_i}$ to the surface. The iterative solver for computation of the coefficients λ_i and the polynomial p requires $C_1 N \log N$ computer operations, where C_1 is a very large constant.

Another well-known method was introduced by Zhao *et al.* in [11, 10], where the authors constructed a weighted minimal surface-like model. The energy is defined as

$$E(\Gamma) = \int_{\Gamma} d(x) ds. \tag{1}$$

Here Γ is an arbitrary surface and ds is the surface area. The unsigned distance function $d(x)$ is computed by solving the Eikonal equation (see [9]):

$$\begin{aligned} |\nabla d(x)| &= 1 \quad x \in \Omega \setminus \Gamma, \\ d(\Gamma) &= 0. \end{aligned} \tag{2}$$

If we represent the surface Γ by a level set function ϕ, the gradient flow of energy (1) can be written as:

$$\phi_t = |\nabla\phi| \nabla \cdot \left[d \frac{\nabla\phi}{|\nabla\phi|} \right] = |\nabla\phi| \left[\nabla d \cdot \frac{\nabla\phi}{|\nabla\phi|} + d \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} \right]. \tag{3}$$

The above partial differential equation (PDE) based method successfully reconstructs the surface (see [11], [10] for more details). However, solving the above PDE requires small time steps and hence longer computational time. Furthermore, the energy functional (1) is not convex, which makes the result sensitive to the initialization and noise. Therefore, in order to avoid local minimizers, one should start from an initial surface which is very close to the given point cloud.

Our method, as will be described in the following section, gives a faster implicit surface reconstruction that is less sensitive to initialization and more robust in the presence of noise (possibly nonuniform noise).

2 Proposed Model

Given a data set $\{\boldsymbol{x}_l\}_{l=1,\ldots,N} \subset \mathbb{R}^{\dim}$, i.e. a set of points with dim = 2 or 3, we seek a function ϕ that is close to zero on this set and smooth elsewhere (see [8]). For this purpose, we consider the following energy functional:

$$E(\phi) = \int G(\phi(x)) dx + \sum_{l=1}^{J} \beta_l (P_l \phi)^2, \tag{4}$$

where the projection operator P_l is some local averaging defined as

$$P_l \phi = \int p_l(x) \phi(x) dx, \quad \int p_l(x) dx = 1. \tag{5}$$

The first term in (4) is the regularization term which imposes smoothness on the level set function ϕ. The second term is the fidelity term which forces the zero level set of ϕ align with the data set. If the data is uniformly distributed and the surface is well-resolved in some region, we set $G(\phi(x)) = |\nabla\phi(x)|^2$. Otherwise,

in regions that lack points, more sophisticated regularization is required, for instance, defining $G(\phi(x)) = |\nabla \cdot \nabla \phi(x)|^2$. Furthermore, special regularization is needed for the anisotropic case when the coefficient function varies in different directions. For example in $\mathbb{R}^2$, when the coefficient function $a_1(x, y)$ in the x-direction differs from the coefficient function $a_2(x, y)$ in the y-direction, we set $G(\phi(x, y)) = a_{11}|\phi_x|^2 + a_{12}|\phi_x\phi_y| + a_{22}|\phi_y|^2 = \alpha_1|\phi_\xi|^2 + \alpha_2|\phi_\eta|^2$ with some proper change of variables $(x, y) \to (\xi, \eta)$. In this paper, we will focus on the uniform regularization with $G(\phi(x)) = |\nabla\phi(x)|^2$.

The values of the weight function β_l should, in general, depend on the accuracy of data points, the density of data set, and the curvature of surface to be reconstructed. As a starting point, a fixed β is used for all data points in this paper.

We now provide the detailed formulation of the energy $E(\phi)$ and the projection operator P_l in the discrete setting. We first consider the two dimensional case. The discretization of the projection operator (5) takes the following general form:

$$\overline{P}_l\phi = \sum_{k,n} \bar{p}^l_{k,n}\phi_{k,n}, \quad \sum_{k,n} \bar{p}^l_{k,n} = 1. \tag{6}$$

The energy functional (4) is discretized as

$$\bar{E}(\phi) = \sum_i \sum_j \Big(\frac{\phi_{i+1,j} - \phi_{i,j}}{h}\Big)^2 + \Big(\frac{\phi_{i,j+1} - \phi_{i,j}}{h}\Big)^2 + \sum_l \beta_l[\bar{P}_l\phi]^2. \tag{7}$$

Then at the grid point (i, j), the Euler-Lagrange equation is given by

$$\frac{1}{2}\frac{\delta\bar{E}}{\delta\phi_{i,j}} = -\frac{\phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} - 4\phi_{i,j}}{h^2} + \sum_{l=1}^{J} \beta_l p^l_{i,j} P_l\phi = 0, \tag{8}$$

where J is the total number of neighboring points of grid point (i, j). Operator P_l is a linear interpolation operator, and $P_l\phi$ represents the interpolated value of function ϕ at a point $\boldsymbol{x}_l$. Figure 1 shows the l-th point inside a grid cell associated to (i, j), with location determined by r_1 and r_2 $(0 \le r_1, r_2 < h)$. The interpolation coefficients are:

$$\begin{aligned} &p^l_{i,j} = \frac{(h - r_1)(h - r_2)}{h^2}, \quad p^l_{i,j+1} = \frac{(h - r_1)r_2}{h^2}, \\ &p^l_{i+1,j} = \frac{r_1(h - r_2)}{h^2}, \quad p^l_{i+1,j+1} = \frac{r_1 r_2}{h^2}, \\ &P_l\phi = p_{i,j}\phi_{i,j} + p_{i,j+1}\phi_{i,j+1} + p_{i+1,j}\phi_{i+1,j} + p_{i+1,j+1}\phi_{i+1,j+1}. \end{aligned} \tag{9}$$

In three dimensions, the corresponding Euler-Lagrange equation at a point (i, j, k) is given as

$$\frac{1}{2}\frac{\delta\bar{E}}{\delta\phi_{i,j,k}} = -\frac{\phi_{i+1,j,k} + \phi_{i-1,j,k} + \phi_{i,j+1,k} + \phi_{i,j-1,k} - 6\phi_{i,j,k}}{h^3} + \sum_{l=1}^{J} \beta_l p^l_{i,j,k} P_l\phi = 0. \tag{10}$$

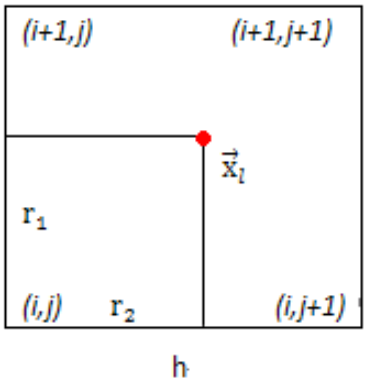

Fig. 1. The illustration of 2D interpolation operator. Here r_1 is the displacement in vertical direction and r_2 is the displacement in horizontal direction.

The natural choice for interpolation operator in three dimensions is bilinear interpolation. In the following formulation, r_1 and r_2 are defined as above, with r_3 denoting the displacement in the z dimension. The interpolation coefficients in three dimensions are given as:

$$
\begin{aligned}
p^l_{i,j,k} &= \frac{(h-r_1)(h-r_2)(h-r_3)}{h^3}, & p^l_{i,j+1,k} &= \frac{(h-r_1)r_2(h-r_3)}{h^3}, \\
p^l_{i+1,j,k} &= \frac{r_1(h-r_2)(h-r_3)}{h^3}, & p^l_{i+1,j+1,k} &= \frac{r_1r_2(h-r_3)}{h^3}, \\
p^l_{i,j,k+1} &= \frac{(h-r_1)(h-r_2)r_3}{h^3}, & p^l_{i+1,j+1,k+1} &= \frac{r_1r_2r_3}{h^3}, \\
p^l_{i+1,j,k+1} &= \frac{r_1(h-r_2)r_3}{h^3}, & p^l_{i,j+1,k+1} &= \frac{(h-r_1)r_2r_3}{h^3},
\end{aligned}
\tag{11}
$$

$$
\begin{aligned}
P_l\phi = {} & p_{i,j,k}\phi_{i,j,k} + p_{i,j+1,k}\phi_{i,j+1,k} + p_{i+1,j,k}\phi_{i+1,j,k} \\
& + p_{i+1,j+1,k}\phi_{i+1,j+1,k} + p_{i,j,k+1}\phi_{i,j,k+1} + p_{i,j+1,k+1}\phi_{i,j+1,k+1} \\
& + p_{i+1,j,k+1}\phi_{i+1,j,k+1} + p_{i+1,j+1,k+1}\phi_{i+1,j+1,k+1}.
\end{aligned}
$$

In the next section, we will describe an efficient numerical implementation for solving (8) and (10).

3 Numerical Implementation

In this section, we consider two dimensional case and describe the process of obtaining a numerical solution of equation (8). The treatments for three dimensional case (10) are similar.

3.1 Initialization

In order to solve equation (8) numerically, we define a suitable computational domain. First, given the coordinates of the data $\{\boldsymbol{x}_l\}$, we find the smallest rectangular box that contains the data. Since extra space should be allocated at the boundaries of the computational domain, we extend the rectangular box by some factor ρ. The typical choice for extension is $\rho = 1.2$. In case data set has a large hole (i.e. a large region with missing information), the value ρ is increased.

To make the algorithm efficient, we can restrict our computations within a narrow band containing the data. To construct the narrow band, we first calculate the unsigned distance function to the data set S by solving the Eikonal equation (See Equation (3)). For the boundary conditions for (3), if the data point is not on the grid, we set the unsigned distance function d to zero at the nearest neighboring grid point to the data point in consideration. We then solve Equation (3) using fast sweeping method [9, 17], an efficient algorithm with a computational cost of $O(N)$. The narrow band is obtained by thresholding unsigned distance function at value $\epsilon = \frac{1}{2} m_h h$, where h is the mesh size and m_h is the band width. Here, we denote $\Omega_1 = \{x : d(x) < \epsilon\}$ to be the set of grid points within the narrow band.

Once the narrow band is obtained, we need to find the grid points for the outer and inner boundaries in order to set up the boundary conditions for (8). Outer and inner boundary grid points are denoted as Γ_{in} and Γ_{out}, respectively, and are obtained using the Breadth-First Search (BFS) algorithm. In graph theory, breadth-first search (BFS) is a graph search algorithm that begins at the root node and explores all the neighboring nodes. Then for each of those nearest nodes, it explores their unexplored neighbors, and so on, until it finds the goal. In order to find outer and inner grid points, first, we take a thin narrow band Ω_2, which is directly connected to Ω_1, such that the band width is equal to one grid point. Specifically, $\Omega_2 = \{x : \epsilon \leq d(x) \leq \epsilon + h\}$. Second, we select an arbitrary outer boundary grid point $(i, j) \in \Omega_2$. This can be easily done by taking the first grid point in Ω_2 since the outer boundary of the narrow band is usually reached while traversing the grid points. This grid point is then assigned to Γ_{out}. Third, we use BFS algorithm to find all the connected grid points among $(i \pm 1, j), (i, j \pm 1)$ which are assigned to Γ_{out}. The algorithm stops when there are no new grid points to be added into Γ_{out}. The complementary set of Γ_{out} is the inner boundary $\Gamma_{\text{in}} = \Omega_2 / \Gamma_{\text{out}}$.

If the data has spurious noise, e.g. contains points far from the meaningful data points, the noise and the data set would be disconnected. In this case, all connected components are obtained using the BFS algorithm as described above. We first select one starting point and find the maximally connected component. Then, we select another point in the complementary set and find another maximally connected component. The process is repeated until all data points are traversed once. Since spurious noise is usually composed of relatively few data points, we let Γ_{out} and Γ_{in} be the largest and second largest connected components, respectively.

Once exterior and interior boundary grid point sets Γ_{in} and Γ_{out} are obtained, we can specify boundary conditions for $\phi(x)$. For the exterior boundary Γ_{out}, we set $\phi(x) = d(x)$; for the interior boundary Γ_{in}, we set $\phi(x) = -d(x)$; and for the points within the narrow band, we simply set $\phi(x) = 0$.

Fig.(2a) shows the initialization of the narrow band associated with the given data set. The red dots and black dots enclose the blue data points with the equal distance $\epsilon = 5h$. The unsigned distance function at the red outer boundary and the black inner boundary points is equal to $5h$ and $-5h$, respectively. The grid points inside the narrow band are all set to zero.

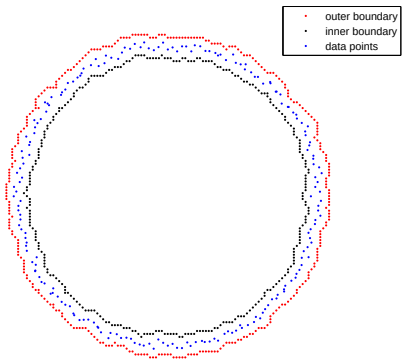

Fig. 2. The band width parameter $\epsilon = 5h$. The red dots are outer boundary points, the black dots are inner boundary points, and the blue dots are the data points.

3.2 Multigrid Narrow Band Solver

The problem defined by (8) with boundary conditions is a well-posed elliptic problem and can be efficiently solved using multigrid method. The Full Approximation Scheme (FAS) [4] solves equation (8) on multiple grids, starting with the coarsest and finishing at the finest grid. On coarser levels, a single cell may contain multiple data points. Since high resolution results are not required on coarser grids, mean coordinates of all data points in a given cell can be used to represent all such points for the purpose of calculating the interpolation operator P_l. After the equation is solved on a coarser level using a few Gauss-Seidel relaxations [4], the obtained solution is linearly interpolated onto the finer level. At the finest level, all data points within each cell are considered to be carrying equal weights. The solution obtained on the finest level is the desired final reconstruction.

We usually use three level construction method with the band width $m_h h$. Here, $m_h = 5$, and h is the mesh size for the corresponding level.

We are now ready to give an algorithm for the Multigrid Narrowband Surface Reconstruction:

Algorithm 1 Multigrid Narrow Band Surface Reconstruction

1: Given a set $\{\boldsymbol{x}_l\}_{l=1,\ldots,N}$, solve Eikonal equation (3) to find an unsigned distance function $d(x)$ for the data set.
2: Find a narrow band with band width ϵ to enclose the data set. Record all the grid points inside narrow band as Ω_1.
3: Find the outer boundary Γ_{out} and inner boundary Γ_{in} of the narrow band.
4: Assign the values of $\phi(x)$ to ϵ at the outer boundary and to $-\epsilon$ at the inner boundary.
5: Denote l to be the coarsest level.
6: Solve equation (8) on level l within narrow band. If l is the finest level, then stop.
7: Linearly interpolate the solution from level l to a finer level $l+1$.
8: Set $l := l+1$ and go to Step 6.

4 Experimental results

In this section, we show the two and three dimensional results obtained using the proposed Multigrid Narrow Band surface reconstruction model to solve (8) and (10). In our numerical examples, we found that $\beta = 0.5$ produces desirable results for most data sets. The value of β may change somewhat with changes in noise level and density for a given data set. Fig. (3) shows a two-dimensional point cloud and multigrid surface reconstruction results at four consecutive steps on different levels. The reconstruction captures additional details as the grid is refined, with the finest level having the most information.

Fig. 4 shows a two-dimensional numerical result for a man made hand with Gaussian noise. The hand shape has long and thin concave regions which are difficult to reconstruct using traditional methods. However, our method recovers the detailed information. We observe some inaccuracy at the ends of the fingers, which may be attributed to the use of uniform β for high curvature places.

Next we consider some three dimensional examples. Fig. 5 shows 3D multigrid surface reconstruction results for "bunny" point cloud on different levels. The difficulty of reconstructing this point cloud lies in the presence of several holes. We observe that finer features are recovered as the mesh size becomes smaller, and the holes are filled automatically. On $257 \times 257 \times 201$ mesh, the total computational time to obtain the final result on 2.93 GHz Intel Core 2 Duo CPU is 6.5 seconds (see Table 1).

Fig. 6 shows 3D surface reconstruction result from "dragon" point cloud. Since this data set contains numerous regions with missing information (i.e. small holes in the point cloud), it is difficult to reconstruct the corresponding surface. The proposed model is successful in capturing fine details and filling the holes in the surface.

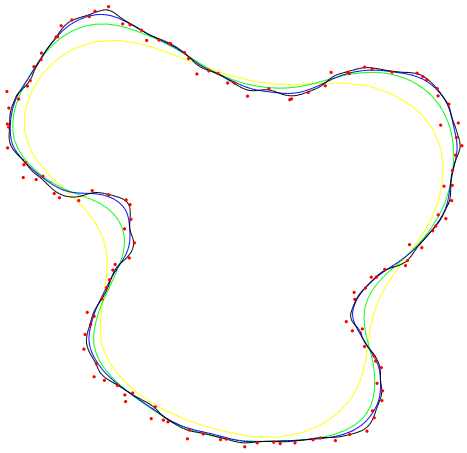

Fig. 3. Two-dimensional multigrid narrow band surface reconstruction results are shown on grids of different resolution. Contours reconstructed on different levels, from coarsest to finest, are displayed in yellow, green, blue, and black colors.

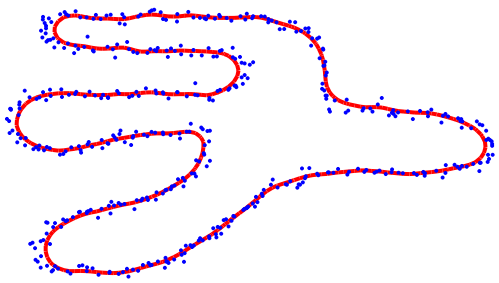

Fig. 4. Two-dimensional multigrid narrow band surface reconstruction of a man made hand with Gaussian noise is shown. A noisy point cloud (in blue) and the reconstructed shape (in red) are displayed.

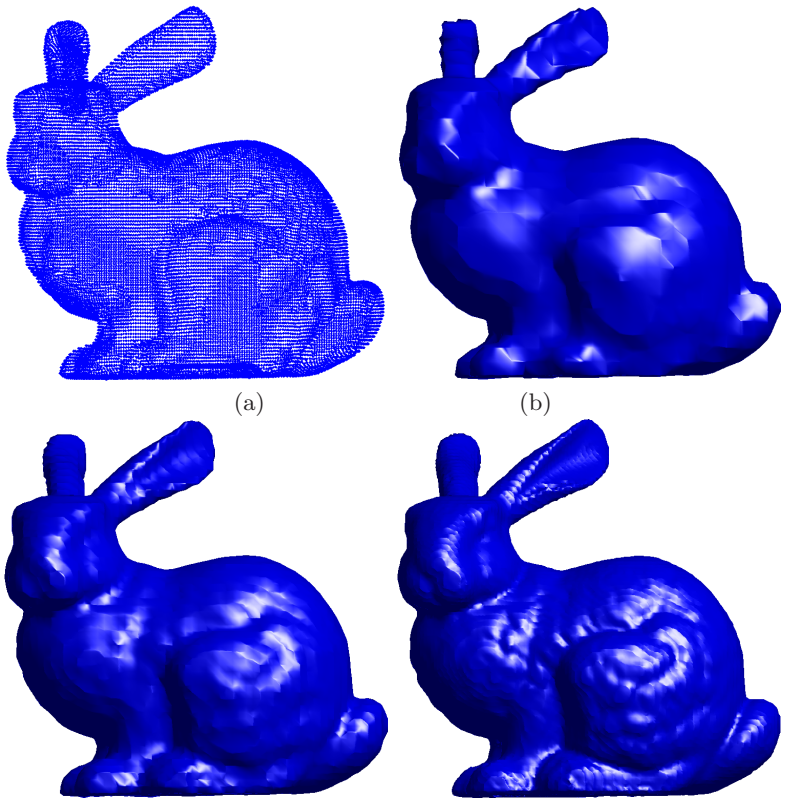

Fig. 5. Three dimensional multigrid narrow band surface reconstruction for bunny. (a) Original point cloud. The results are obtained on (b) 65x65x51, (c) 129x129x101, and (d) 257x257x201 grids.

Fig. 7 shows 3D surface reconstruction result from Buddha point cloud. Even though this data set contains no holes, it has small bridges. We observe that all fine features are captured well using the proposed surface reconstruction model.

5 Conclusion

The two and three dimensional curve/surface reconstruction results presented in this paper demonstrate that our method is among the fastest surface reconstruction methods. Furthermore, the proposed method is robust to noise and can easily recover surfaces from point clouds with missing information. The computational time is $O(N^3)$ where N is the grid size of final level reconstruction.

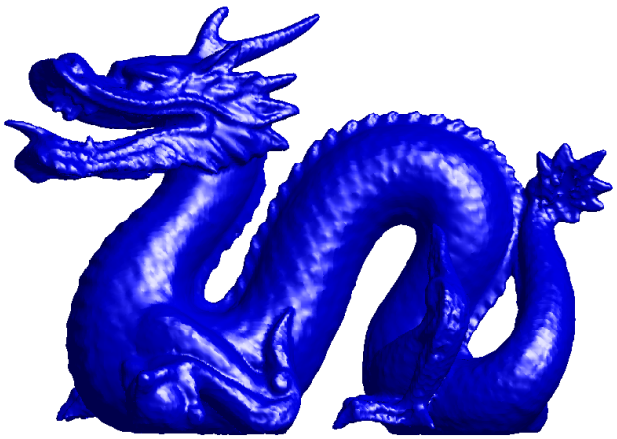

Fig. 6. Three dimensional multigrid narrow band surface reconstruction for dragon on $301 \times 213 \times 133$ grid. The point cloud contains 437645 data points.

Fig. 7. Three dimensional multigrid narrow band surface reconstruction for Buddha on $149 \times 365 \times 149$ grid. The point cloud contains 144647 points.

Table 1. Computational times for 3D data sets.

data set	size	time (in seconds)
bunny	257x257x201	6.5
dragon	301x213x133	7.2
buddha	149x365x149	6.7

Acknowledgements

This work was supported in part by NIH GRANT, P20 MH65166, NSF GRANT, DMS-0714807, and NIH GRANT, U54 RR021813. The research of Igor Yanovsky was carried out in part at the University of California, Los Angeles, and in part at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

1. N. Amenta, M. Bern, and D. Eppstein, "The crust and the β-skeleton: Combinatorial curve reconstruction," *Graphical Models and Image Processing*, vol. 60, no. 2, pp. 125–135, 1998.
2. N. Amenta, M. Bern, and M. Kamvysselis, "A new Voronoi-based surface reconstruction algorithm," pp. 415–421, 1998.
3. J. Boissonnat, "Geometric structures for three dimensional shape reconstruction," *ACM Trans. Graphics 3*, pp. 266–286, 1984.
4. A. Brandt, "Multigrid techniques: 1984 guide with applications to fluid dynamics," 1984.
5. J. Carr, W. Fright, and R. Beatson, "Surface interpolation with radial basis functions for medical imaging," *IEEE Transactions on Medical Imaging*, vol. 16, no. 1, pp. 96–107, 1997.
6. H. Edelsbrunner, "Shape reconstruction with Delaunay complex," *Lecture Notes in Computer Science*, pp. 119–132, 1998.
7. H. Edelsbrunner and E. P. Mucke, "Three dimensional α shapes," *ACM Trans. Graphics 13*, pp. 43–72, 1994.
8. R. Gandlin, "Multigrid solvers for inverse problems," 2004, ph.D. thesis, Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science.
9. H.K.Zhao, "A fast sweeping method for eikonal equations," *Mathematics of Computation*, vol. 74, no. 250, pp. 603–627, 2004.
10. H.K.Zhao, S.Osher, B.Merriaman, and M.Kang, "Implicit and nonparametric shape reconstruction from unorgazied data using a variational level set method," *Computer Vision and Image Understanding*, vol. 80, no. 3, 2000.
11. H.K.Zhao, S.Osher, and R.Fedkiw, "Fast surface reconstruction using the level set method," 2001.
12. S. Leung and H. Zhao, "A grid based particle method for moving interface problems," *Journal of Computational Physics*, vol. 228, no. 8, pp. 2993–3024, 2009.
13. S. Osher and R. Fedkiw, *Level set methods and dynamic implicit surfaces*. Springer, 2003.

14. L. Piegl and W. Tiller, *The NURBS book*. Berlin, Germany: Springer-Verlag, 1996.
15. D. Rogers, *An Introduction to NURBS*. Morgan Kaufmann, 2003.
16. S. Ruuth and B. Merriman, "A simple embedding method for solving partial differential equations on surfaces," *Journal of Computational Physics*, 2007.
17. Y. Tsai, L. Cheng, S. Osher, and H. Zhao, "Fast sweeping algorithms for a class of Hamilton-Jacobi equations," *SIAM journal on numerical analysis*, vol. 41, no. 2, pp. 673–694, 2004.