Least Squares and Point-based Surfaces: New Perspectives and Applications

João Paulo Gois Antonio Castelo Filho

Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo
{jpgois,castelo}@icmc.usp.br

Abstract

Surface approximation from unorganized points belongs to the state-of-art of computer graphics. In this work, we present approaches for surface reconstruction that are based on efficient numerical schemes for function approximation from scattered data and on sophisticate data structures. In addition, we develop a relevant surface reconstruction method to model moving interfaces, specifically, interfaces of numerically simulated multiphase fluid flow. Finally, from our accumulated experiences on numerical schemes and on the development of surface reconstruction methods, we propose a matrix-free approach for rendering arbitrary volumetric scattered data, which presents interesting properties to be implemented on GPU.

1. Introduction

It is not difficult to find reasons for surface reconstruction methods being a strong tendency in computer graphics. First, despite significant advances, the manipulation of complicate point clouds originated from arbitrary geometries is an important issue that still requires improvements. Second, several research areas have make use of surface reconstruction methods, for instance, reverse engineering, optimization and medicine.

In this work, we present our contributions to surface reconstruction from unorganized points, whose their developments were focused on novel moving-least-squares and partition of unity implicits methods. It will be noticed that our methods have common features of having low computational cost (mainly regarding matricial computations) and being robust. The use of efficient data structures to aid the surface reconstruction is also in the scope of our work [7, 9, 8, 10, 18, 6, 17]. In addition, we address a surface reconstruction scheme for the front-tracking problem. An important contribution of such scheme is the absence of Lagrangian meshes. In fact, it is designed on the algebraic moving-least-squares method and on a approach for Lagrangian advection of particles. Finally, from our experiences on surface reconstruction and function approximation, we also developed a method for rendering scattered volumetric data based on a recently proposed matrix-free approach.

In the following sections, we describe relevant details of our approaches. Further information and comparisons can be found in [7, 9, 8, 10, 18, 6].

2. Twofold Adaptive Partition of Unity Implicits

We propose an approach based on partition of unity implicits [8, 9] which combines an adaptive and algebraic triangulation, named J_1^a [3], and a recursively defined local approximation method based on multivariate orthogonal polynomials [2].

Similar to other implicit methods for surface reconstruction, the partition of unity implicits defines the reconstructed surface as the zero set of a function F, in a finite domain Ω, from a linear combination of local approximations and weighting functions. For that purpose, we define a set of nonnegative weighting functions with compact support $\Phi = \{\omega_1, \ldots, \omega_n\}$, satisfying $\sum_{i=0}^{n} \omega_i(\mathbf{x}) \equiv 1, \ \mathbf{x} \in \Omega$ and we also consider a set of signed local approximations $\mathbb{F} = \{f_1, \ldots, f_n\}$, which completely define the function $F : \mathbb{R}^3 \to \mathbb{R}$:

$$F(\mathbf{x}) \equiv \sum_{i=0}^{n} f_i(\mathbf{x})\omega_i(\mathbf{x}), \ \ \mathbf{x} \in \Omega. \tag{1}$$

The *partition of unity* is then defined from a set of nonnegative functions with compact support θ:

$$\omega_i(\mathbf{x}) = \frac{\theta(\|\mathbf{x} - \mathbf{c}_i\| / R_i)}{\sum_{k=1}^{n} \theta(\|\mathbf{x} - \mathbf{c}_k\| / R_k)}, \tag{2}$$

0 Ph.D. thesis

where $\mathbf{c}_i$ and R_i are the center and the support radius of ω_i, respectively, and $\theta(s) = (1-s^2)^4$ if $s < 1$; $\theta(s) = 0$ otherwise.

Thus, the implicit surface is given by $\mathscr{S} = \{\mathbf{x} \in \mathbb{R}^3 : F(\mathbf{x}) = 0\}$. Figure 1 illustrates a CSG operation between Neptune model reconstructed from our method and a cylinder.

Figure 1. A CSG operation from Neptune dataset reconstructed using our partition of unity implicits method and a cylinder [9].

Briefly, the main contributions of this approach are:

Adaptive isosurface extraction with topological guarantee: in contrast to previous works, our method is able to extract the isosurface directly from the data structure which subdivides the space (the J_1^a triangulation). In addition, since J_1^a triangulation is defined by tetrahedra, the isosurface algorithm is not affected by ambiguous cases.

Numerical stability: despite their simplicity, canonical polynomial basis leads to ill-conditioned normal equations. In order to avoid such an issue, we make use of the multivariate orthogonal polynomial basis proposed by Bartels and Jezioranski [2], which improves the approximation quality without increasing the computational burden.

Adaptiveness of the local fittings: the use of orthogonal polynomials allows to increase the polynomial degree recursively. For that reason, our method is not only adaptive with respect to the spatial decomposition, but also with respect to the local fittings, defining a *twofold adaptiveness*. Figure 2 illustrates such a twofold adaptiveness.

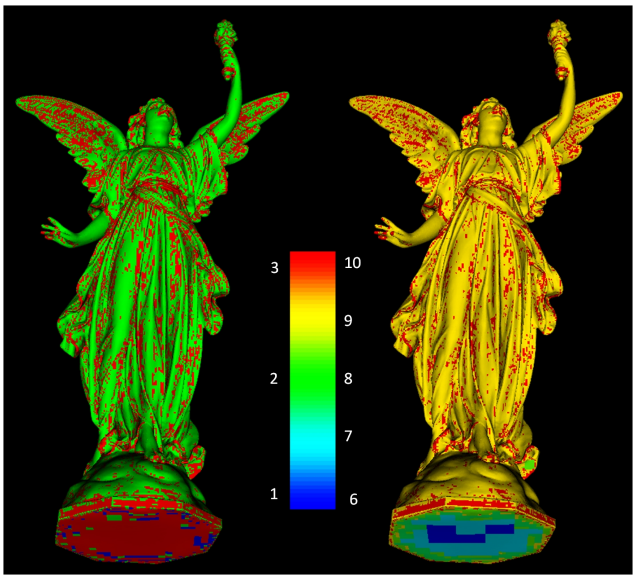


Figure 2. Twofold adaptiveness: Stanford Lucy with 16 M points. Reconstructed model with 7 M triangles. On the left, the color scale represents the polynomial degree of the local approximation in the surface. In the entire domain, the numbers of polynomials with degrees one to four are 946601, 144956, 38236 and 26862, respectively. On the right, the scale color represents the J_1^a triangulation depth level, ranging from 6 to 10 in the surface [9].

Robustness: we propose not only computationally inexpensive, but also effective robustness criteria which allow our method to produce better results than previous works based on partition of unity implicits. In fact, our method is less susceptible to generate spurious surfaces.

Mesh enhancement: the aspect ratios of the triangles generated by the isosurface extractor based on the J_1^a triangulation, in general, are poor. For that reason, we define a simple, but efficient procedure to improve the surface mesh quality. This procedure is based on the displacement of the J_1^a vertices [4]. Figure 6 presents a result attesting that the mesh quality significantly increases. For quantitative analyses, see [9, 6].

Interactive implicit function edition: in order to satisfies the robustness criteria, details in some regions can be poorly approximate, since low degree polynomials are used. On the other hand, neglecting the robustness criteria is not a convenient choice, since spurious surface and artifacts can occur. In addition, increasing the restrictions on the robustness conditions may cause undesirable smoothing of

surface details without ensuring that all imperfections will disappear. Therefore, we developed an interactive implicit function edition which allows to locally change the local fittings without the need of re-computing completely the implicit function (Figures 3-(e)–(f) and 4).

Sharp features modeling: finally, we also properly adapt the scheme from Ohtake et al. [14] for detecting and modeling sharp features to our twofold partition of unity implicits method (Figure 5).

3. Front-tracking with moving-least-squares surfaces

We propose a meshfree front-tracking method for numerical simulation of fluid flows, where the interface is represented by an algebraic-moving-least-squares (AMLS) surface. Our method is interesting for being capable of preserving mass and geometry as well as mesh-based front-tracking methods [19] and being as flexible as level-set methods [15].

Let us again consider a finite set of points $\mathscr{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_m\}$ almost regularly spaced at a distance h and $\{q_k\}$ an algebraic sphere basis [16], i.e., $q_1(\mathbf{x}) = 1$, $q_2(\mathbf{x}) = x_1$, $q_3(\mathbf{x}) = x_2$, $q_4(\mathbf{x}) = x_3$, $q_5(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2$, where $\mathbf{x} = (x_1, x_2, x_3)$. Thus, we define a function $f_\alpha(\mathbf{x})$ whose zero level approximates $\mathscr{P}$:

$$f_\alpha(\mathbf{x}) = \sum_{k=1}^{5} \alpha_k \, q_k(\mathbf{x}) \tag{3}$$

where $\alpha = (\alpha_1, \cdots, \alpha_5)$ are the coefficients which completely defines f_α and α satisfies:

$$\alpha = \arg\min_{\beta \in Q} \sum_{j=1}^{m} w_j |f_\beta(\mathbf{p}_j)|^2, \tag{4}$$

where $w_j = w(\mathbf{x}) = \theta\left(\frac{\|\mathbf{x}-\mathbf{p}_i\|}{3h}\right)$, defining non-negative weights ($\theta(s) = (1-s^2)^4$ if $s < 1$; $\theta(s) = 0$ otherwise) and $Q = \{\beta \in \mathbb{R}^5 : \beta_2^2 + \beta_3^2 + \beta_4^2 - 4\beta_1\beta_5 = 1\}$, being the restriction used to ensure that $f_\alpha(\mathbf{x})$ approximates the signed distance of $\mathbf{x}$ to the surface $f_\alpha \equiv 0$.

The minimization (4) can be computed by a generalized eigenvalue problem in $\mathbb{R}^5$ [11]. In addition, for each point $\mathbf{p}_i$, the approximate normal is given by:

$$\mathbf{N}_i = \nabla f_{\alpha_i}(\mathbf{p}_i) / \|\nabla f_{\alpha_i}(\mathbf{p}_i)\|. \tag{5}$$

The sphere-fitting problem, previously described, can be substantially improved by introducing the normals [11]. Furthermore, the introduction of normal in the minimization problem allows the splitting of the problem in two main steps. First, we compute the subset:

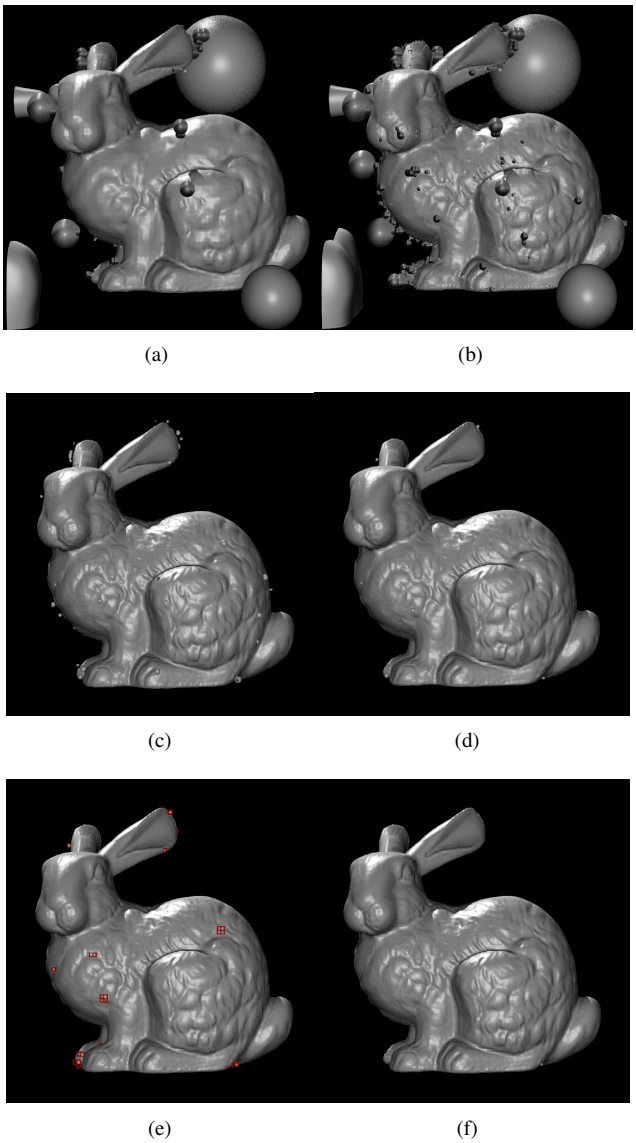
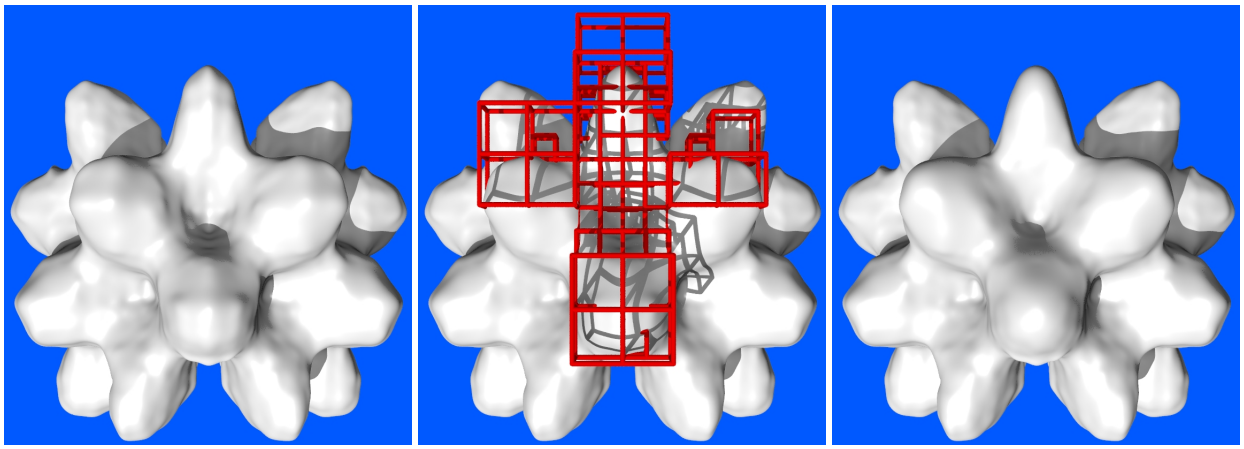


Figure 3. Interactive function edition: (a)-(b) present results from Ohtake et al. [14] with their default parameters and those one that we suggested, respectively; (c) our method but not using the coverage domain criterion [9]; (d) our method making use of the coverage domain criterion; (e) selecting surface artifacts; (f) the edited function eliminates imperfections [9].



(a) (b) (c)

Figure 4. Reconstruction enhancement by the interactive function edition: (a) original result with low degree approximations, (b) support selection for function edition (c) final result [9].

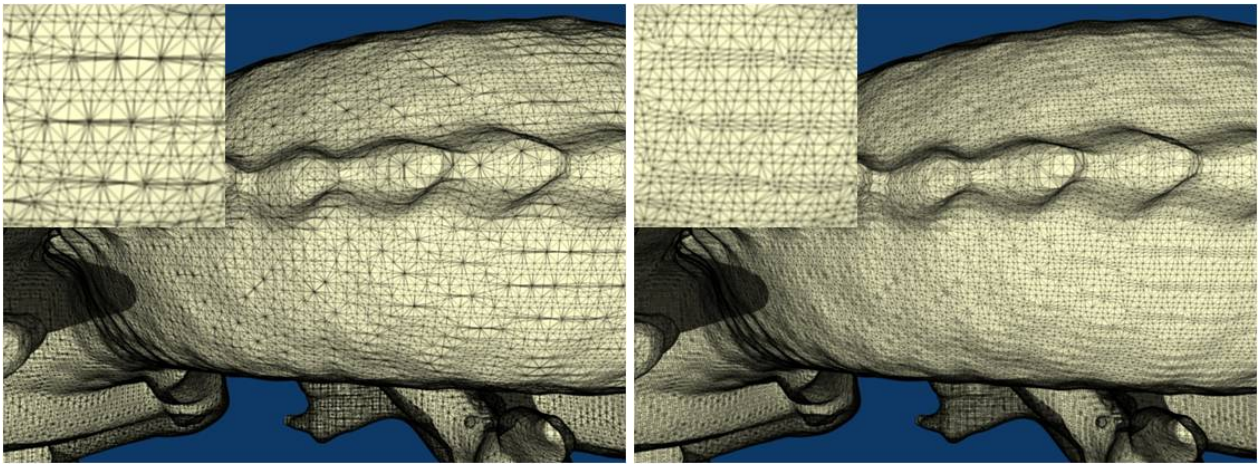


Figure 6. Mesh enhancement: Comparison between the mesh produced by the J_1^a triangulation without the displacement vertices procedure (left) and the mesh produced by the displacement vertices procedure (right) [9].

$\hat{\gamma}(\mathbf{x}) = (\gamma_2(\mathbf{x}), \ldots, \gamma_5(\mathbf{x}))$ of $\gamma = (\gamma_1, \cdots, \gamma_5)$ finding:

$$\hat{\gamma}(\mathbf{x}) = \underset{\hat{\beta}\in\mathbb{R}^4}{\arg\min} \sum_{i=1}^{m} w_i \|\nabla f_{\hat{\beta}}(\mathbf{p}_i) - \mathbf{N}_i\|^2. \tag{6}$$

where $\mathbf{N}_i$ is computed as previously described. Notice that the solution $\hat{\gamma}(\mathbf{x})$ of (6) can be determined simply by solving a 4×4 linear system.

Finally, the first term γ_1 of γ is computed by:

$$\sum_{i=1}^{m} w_i |F_\beta(\mathbf{p}_i)|^2 \tag{7}$$

where $\hat{\beta} = \hat{\gamma}(\mathbf{x})$. Thus:

$$\gamma_1(\mathbf{x}) = -\frac{\sum_{i=1}^{m} w_i \left(\sum_{k=2}^{5} \gamma_k(\mathbf{x})\ q_k(\mathbf{p}_i)\right)}{\sum_{i=1}^{m} w_i}. \tag{8}$$

We name such an approximation *RAMLS-Robust Algebraic Moving Least-Squares*. It can be seen as an improvement of the Guennebaud and Gross method [11], since we eliminate the parameter that weighs the normal restriction into the minimization problem (in fact, we assume that the parameter β, in [11], as $\beta \to \infty$).

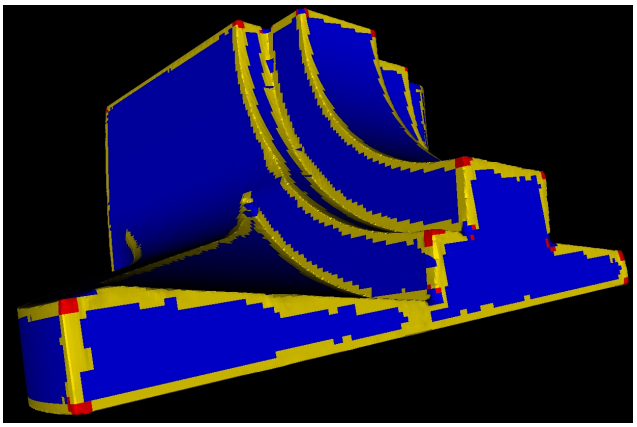


Figure 5. Sharp features detection and modeling (Fan Disk dataset): in yellow are the faces labeled as *sharp edges*, whereas in red are the faces labeled as *sharp corners* [9].

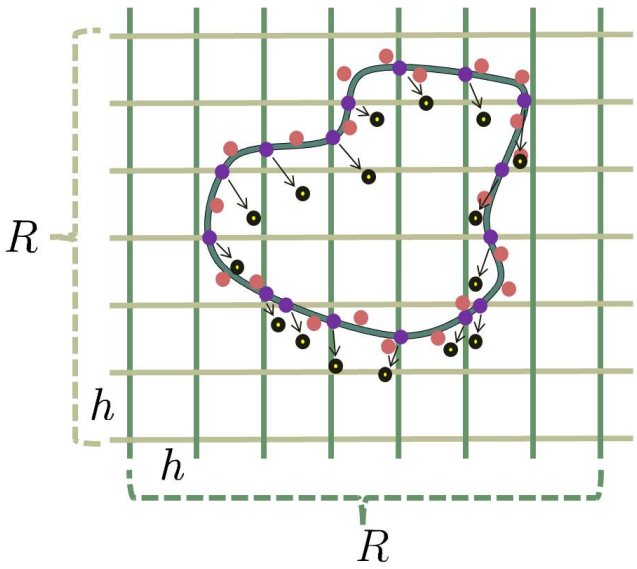


Figure 7. The set of rays R and the points $\mathscr{P}(t)$ (red) define the RAMLS surface $\mathscr{S}(t)$ (green curve). The new set of points $\mathscr{P}(t)$ (purple) is created at the intersections between R and $\mathscr{S}(t)$. The set of points is, thus, transported, defining the set $\mathscr{P}(t+\Delta t)$ (black points), which will define the new surface RAMLS $\mathscr{S}(t+\Delta t)$.

3.1. Point Advection

In order to update the RAMLS surface during the simulation, we move the points $\mathscr{P}$ according to the velocity field $\mathbf{v}$. Mathematically, we solve the following equation:

$$\frac{d\mathbf{p}_i}{dt} = \mathbf{v}(\mathbf{p}_i(t), t)\ , \tag{9}$$

where $\mathbf{p}_i(0)$ are the initial points.

We denote by $\mathscr{P}(t) = \{\mathbf{p}_i(t)\}_{i=1,\dots,m}$ the set of points to be advected from t to $t+\Delta t$ (Δt is the time step). During the transport, the points in $\mathscr{P}(t)$ can become poorly distributed, which could compromise the surface $\mathscr{S}(t)$. In order to avoid such an issue, we propose to re-generate the point set periodically in a regularly spaced grid.

Re-generation points process: Let us consider a set of regularly spaced rays R with spacing h (Figure 7). The set of points $\mathscr{P}(t)$ is transported along the trajectories given by Equation 9. The point transportation is used to define the new RAMLS surface. The intersections between R and the surface define the new set of points $\mathscr{P}(t+\Delta t)$. We depict such a re-generation process in Figure 7.

It is important to mention that Equation 9 is solved by the fourth-order Runge-Kutta method. A ray-tracing-like algorithm is used to detect the intersections between R and the RAMLS surface. Since points can be re-generated very close, we remove points that are closer than $\frac{h}{2}$. Finally, the normals remains unchanged during the transportation in order to be used to orient the new normals in $\mathscr{P}(t+\Delta t)$.

Considering the convergence rates, for the two-dimensional case, we are able to show that RAMLS accomplishes order $O(h^3), O(h^2)$ and $O(h)$ for geometrical, normal and curvature, respectively. Considering the three-dimensional case, RAMLS achieves rates of $O(h^2)$ and $O(h)$ for geometrical and normal curvature, respectivelly. During the transportation, the error is $O(h^r/\Delta t)$, where $r = 2,3$ according to the geometrical error, previously argued.

For the Zalesak Sphere Test, we consider a sphere with radius 0.15, slope with length 0.10 and width 0.20, centered at $(0.5, 0.75, 0.75)$ in a unity cube. The velocity field is given by a rigid rotation:

$$\mathbf{v}(x_1, x_2, x_3) = \frac{\pi}{314}(0.5 - x_2, x_1 - 0.5, 0). \tag{10}$$

We assume $\Delta t = 1$ and a complete turn takes 628 time steps. We show the geometry of the Zalesak spheres at the times $t = 0$, 79, 157, 236, 314, 393, 471, 550 and 628 in Figure 8. We adopt the mesh size $h = 1/256$, and the number of particles as about 19000 for each time step. It can be noticed that the sharp features become smooth along the time, but the volume and shape remain similar [7, 6].

We also test our method to a three-dimensional deforming problem. We consider a sphere as $\mathscr{S}(0)$ with

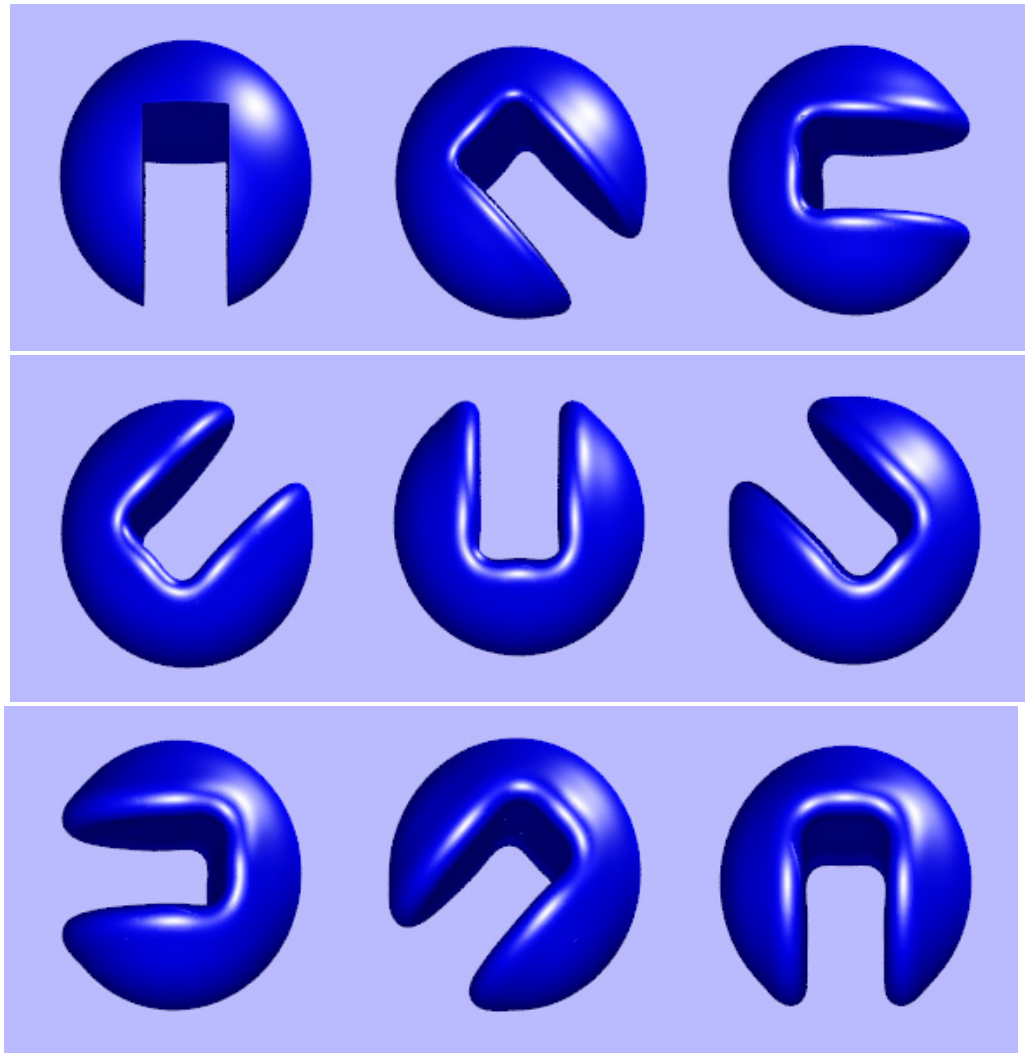


Figure 8. Zalesak sphere rotation ($h = 1/256$): (about 19000 points) and time steps $t = 0, 79, 157, 236, 314, 393, 471, 550$ and 628 time units (from left to right and from top to bottom) [7].

radius 0.15, centered at $(0.35, 0.35, 0.35)$. The computational domain is the unity cube. The velocity field is given by:

$$\mathbf{v}(x_1, x_2, x_3) = \begin{pmatrix} 2\sin^2(\pi x_1)\sin(2\pi x_2)\sin(2\pi x_3) \\ -\sin(2\pi x_1)\sin^2(\pi x_2)\sin(2\pi x_3) \\ -\sin(2\pi x_1)\sin(2\pi x_2)\sin^2(\pi x_3) \end{pmatrix}.$$

At $t = T/2 = 1$ time units, we reverse the velocity field in order to check the capability of the method in preserving topology and geometry. Figure 9 presents results of such a simulation, using $h = 1/512$, $\Delta t = 0.0064$ and 628 time steps.

4. Iterate Approximate Moving Least Squares Surfaces

One of the challenges to develop efficient surface reconstruction methods in modern GPU is the conformance of numerical computations to textures [18], represented by 4×4 matrices.

When implementing a moving-least-squares surface method, in order to achieve high order approximations, the local approximations must be constructed by higher order polynomial basis, and consequently, a single 4×4 matrix does not allow to achieve efficiently such a requirement. Studies to select a reduced polynomial basis [18, 10] have been done. However, the local approximation quality and high order approximations are not ensured.

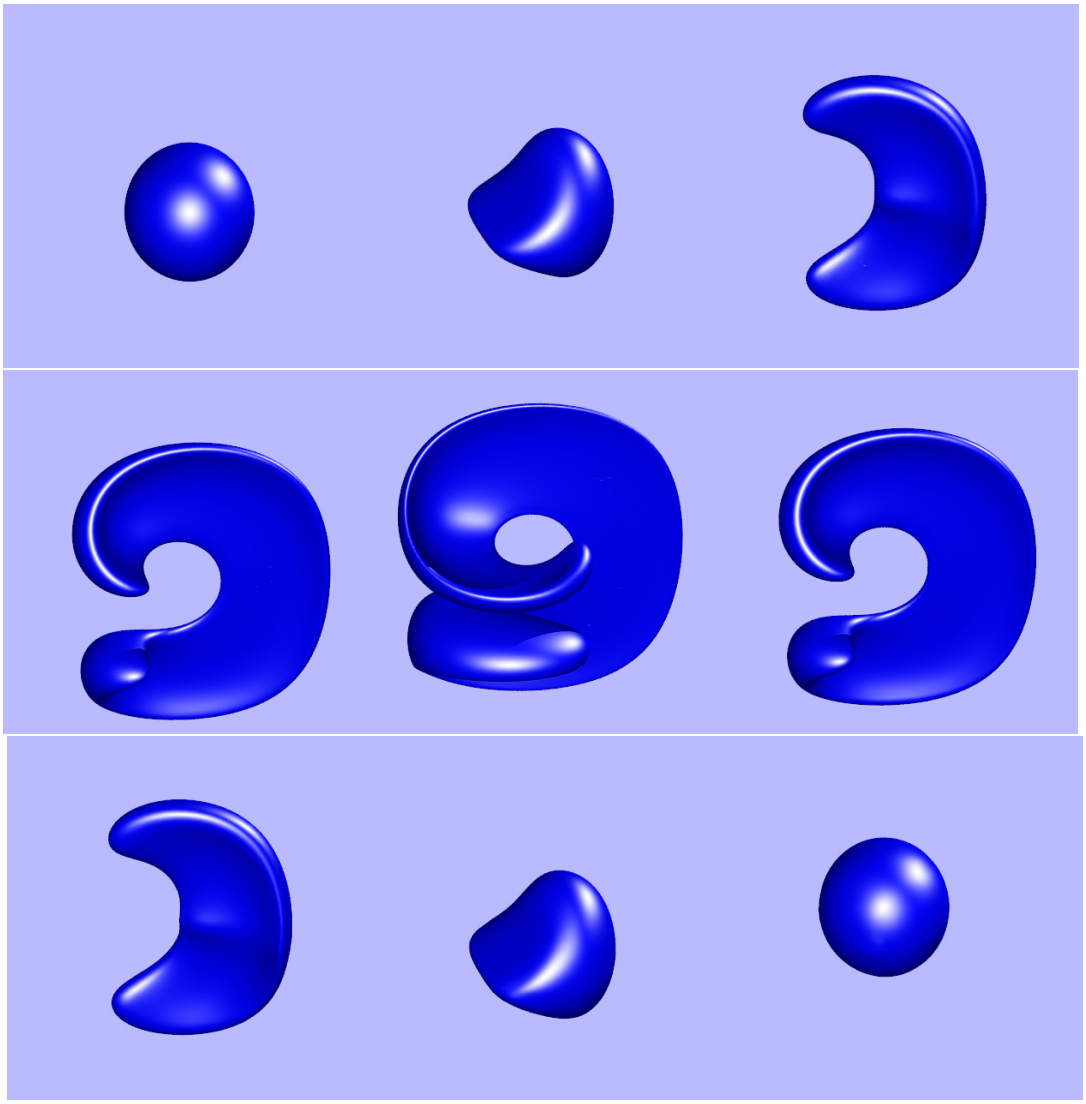


Figure 9. Three-dimensional torsion ($h = 1/512$): the number of points for each set is (from left to right and from top to bottom): 65000, 69314, 109728, 187571, 277021, 193238, 114898, 72774 **and** 68031 **[7].**

To overcome the matrix reliance, we propose an iterative method based on the "iterate approximate moving least squares – iaMLS" [5]. This method is interesting for being able to enhance features simply by controlling the number of iterations and the single smooth parameter ϵ. Formally, let us consider a function $F : \mathbb{R}^3 \to \mathbb{R}$, which its zero set approximates the surface. Let us also consider a set of points $\mathbf{x}_i$, equipped with their normal vector $\mathbf{n}_i$. Thus, we define:

$$F(\mathbf{x}) = \sum_{i=1}^{m} \mathscr{G}_i \Psi_i(\mathbf{x}), \tag{11}$$

where $\mathscr{G}_i =< \mathbf{x}_i - \mathbf{x}, \mathbf{n}_i >$, $\Psi_i(\mathbf{x}) = \frac{\epsilon^3}{\pi^{3/2}} (r_i(\mathbf{x})) \exp(-r_i(\mathbf{x}))$, $r_i = \frac{\|\mathbf{x}-\mathbf{x}_i\|}{h}$ and h is the average spacing between the points [6]. The functions Ψ_i are named second order generating functions [5].

From these definitions, and making use of the iterative process of the iaMLS [5], we define the *iaMLS surface* from the following iterative process:

$$F^{(0)}(\mathbf{x}) = \sum_{i=1}^{m} \mathscr{G}_i \Psi_i(\mathbf{x}) \tag{12}$$

$$F^{(n+1)}(\mathbf{x}) = F^{(n)}(\mathbf{x}) + \sum_{i=1}^{m} \left[\mathscr{G}_i - F^{(n)}(\mathbf{x}_i)\right] \Psi_i(\mathbf{x}), \tag{13}$$

where $F \equiv 0$ gives the implicit surface reconstruction.

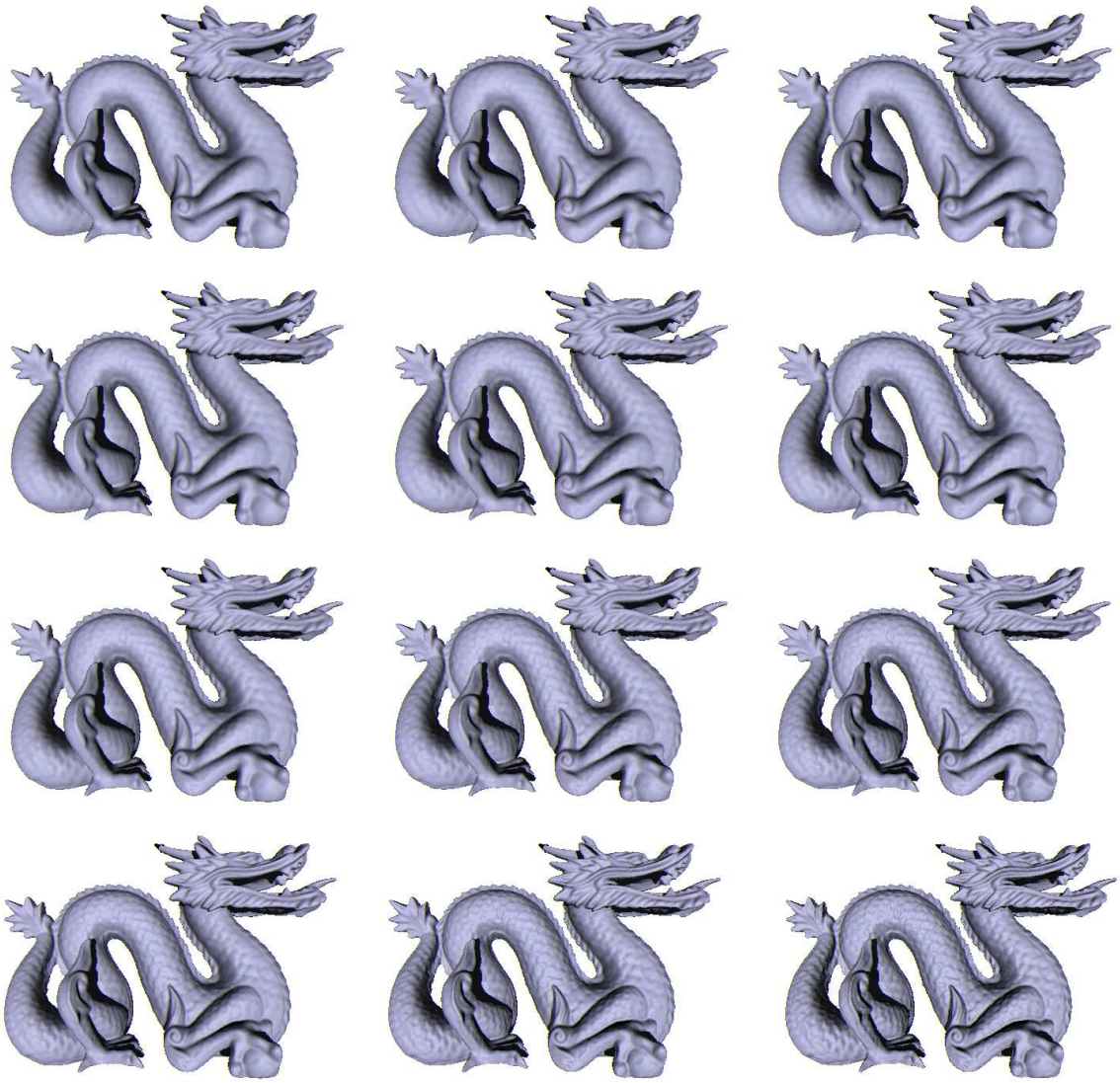


Figure 10. Effects caused by the number of iterations and the ϵ parameter. From left to right: 3,4 and 5 iterations. From top to bottom: $\epsilon = 0.4$, $\epsilon = 0.8$, $\epsilon = 1.22$, $\epsilon = 2.0$. The greater the number of iterations or ϵ are, the more detailed the surface becomes.

Figure 10 depicts how the number of iterations and the parameter ϵ affect the solution. Figure 11 presents comparisons of our approach with other implicit MLS surface methods. It can be noticed that, even using a small number of iterations, we are able to enhance object characteristics. We present comparisons assessing the computational cost of this scheme [6, 17].

5. Volumetric approximation from unorganized points

The last contribution of this work is related to unorganized volumetric data rendering. In fact, the goal is the definition of a method for rendering arbitrary meshes, i.e., a unified approach that explores the iaMLS. Firstly, in order to make the method more efficient for all mesh types, we make use of anisotropic spaces. Notice that the original iaMLS, to the best of our knowledge, does not make use of any anisotropy scheme [6, 17].

Our method is also able to render isosurfaces from

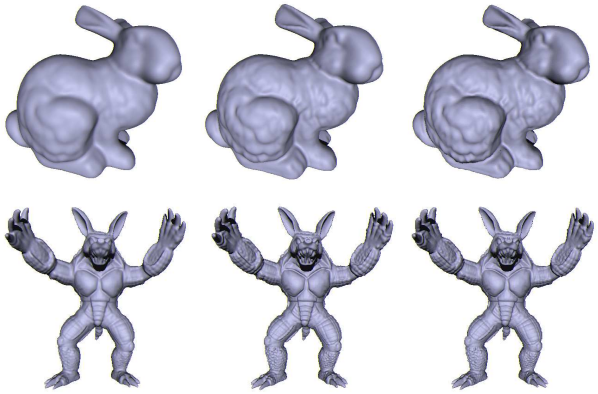


Figure 11. Comparison among Adamson and Alexa [1] (left), Kolluri [12] (middle) and iaMLS [6, 17] (right) methods. We use the same h parameter for each model. We performed 3 iterations and $\epsilon = 0.8$.

unorganized volumetric data. For that purpose, we also apply the iaMLS to estimate the gradient function [6, 17]. In fact, it is required to define, for each mesh vertex $\mathbf{x}_i$, a gradient vector estimate, using a least-squares approach. It is achieved by extending the approach by Mavriplis to the three-dimensional space [13]. It is important to notice that the use of such an approach remains our method independent of meshes.

Let us consider a mesh vertex $\mathbf{x}_i$, in which we evaluate the gradient, and its neighbors $\mathbf{x}_k = (x_k, y_k, z_k)$ in its star $\mathbb{V}_i$. Thus, we minimize the following equation with respect to $\nabla f(\mathbf{x}_i) = ((f_x)_i, (f_y)_i, (f_z)_i)$:

$$\sum_{\mathbf{x}_k \in \mathbb{V}_i} \exp(- < \mathbf{x}_k - \mathbf{x}_i, \mathbf{x}_k - \mathbf{x}_i >)(E_{ik})^2 \tag{14}$$

where:

$$E_{ik} = \left((f_x)_i \cdot dx_{ik} + (f_y)_i \cdot dy_{ik} + (f_z)_i \cdot dz_{ik} - df_{ik}\right) \tag{15}$$

and $df_{ik} = f(\mathbf{x}_k) - f(\mathbf{x}_i)$. By the same token, $dx_{ik} = x_k - x_i, dy_{ik} = y_k - y_i, dz_{ik} = z_k - z_i$.

The gradient vector, in an arbitrary point $\mathbf{x}$, indeed, is given by the iterative process:

$$\mathcal{Q}_{\nabla f}(\mathbf{x})^{(0)} = \sum_{\mathbf{x}_i \in \mathbb{X}} \nabla f(\mathbf{x}_i) \Psi_i(\mathbf{x}), \tag{16}$$

$$\mathcal{Q}_{\nabla f}^{(n+1)}(\mathbf{x}) = \mathcal{Q}_{\nabla f}^{(n)}(\mathbf{x}) + \sum_{\mathbf{x}_i \in \mathbb{X}} \left[\nabla f(\mathbf{x}_i) - \mathcal{Q}_{\nabla f}^{(n)}(\mathbf{x}_i)\right] \Psi_i(\mathbf{x}). \tag{17}$$

Another interesting feature of our approach is its capability to be efficient for GPU implementation. We are able to provide pre-computations, which are stored in GPU textures. We perform two pre-computations based on the iterative process of iaMLS. They are related to the function evaluation and the gradient estimate.

Thus, we can rewrite the iterative process by Fasshauer [5] as:

$$\mathcal{Q}_f^{(n+1)}(\mathbf{x}) = \sum_{\mathbf{x}_i \in \mathbb{X}} \left[f(\mathbf{x}_i) + \sum_{j=0}^{n} \left(f(\mathbf{x}_i) - \mathcal{Q}_f^{(j)}(\mathbf{x}_i) \right) \right] \Psi_i(\mathbf{x}), \tag{18}$$

where, we can accumulate the results for each mesh vertex as:

$$g(\mathbf{x}_i) = f(\mathbf{x}_i) + \sum_{j=0}^{n} \left(f(\mathbf{x}_i) - \mathcal{Q}_f^{(j)}(\mathbf{x}_i) \right), \tag{19}$$

and store them in textures which accommodate the unorganized volumetric data. Then, during the rendering, the reconstructed function value is simply given by:

$$\mathcal{Q}f^{(n+1)}(\mathbf{x}) = \sum_{\mathbf{x}_i \in \mathbb{X}} g(\mathbf{x}_i) \Psi_i(\mathbf{x}). \tag{20}$$

Rendering results are presented in Figure 12 for different meshes. We use $h = 0.25$ and $\epsilon = 0.9$. Both parameters are chosen empirically after numerical tests.

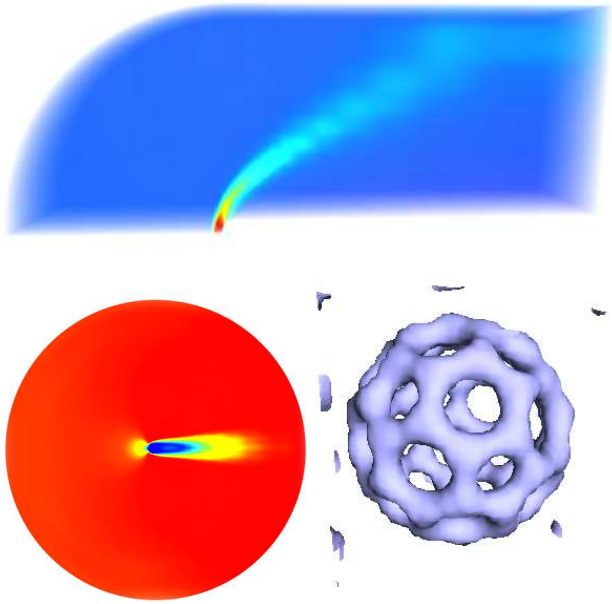


Figure 12. Volumetric rendering of Blunt Fin and Oxygen Post datasets. Isosurface extracted from the Bucky Ball dataset [6, 17].

6. Conclusion

In this work, we presented novel and important contributions towards the field of dynamic and static surface reconstruction methods from unorganized points.

There are several ways to extend the techniques here presented. First, we intend to investigate theoretically our surface reconstruction method based on iaMLS.

With respect to the twofold adaptive partition of unity implicits, we intend further to improve the function edition process by incorporating new local approximating functions. In addition, we want to improve the mesh enhancement technique, making use of the whole scheme by Dietrisch et al. [4], but, in our case, considering the adaptiveness of J_1^a triangulation.

Considering the front-tracking with AMLS surfaces approach, the definition of a good curvature from RAMLS is a mandatory future work. In addition, we aim at applying this surface representation for numerical simulation of fluid flows. We are also investigating schemes to perform topological changes efficiently.

Finally, we intend to investigate the possibility to estimate automatically the parameters h and ϵ of the iaMLS defined in anisotropic domains for volumetric rendering.

Acknowledgments

This work was supported by FAPESP (Brazil) and DAAD (Germany). The methods in Sections 2 and 5 were developed also with Eduardo Tejada and Tomas Ertl (University of Stuttgart–Germany). The work in Section 4 was developed also with Gustavo C. Buscaglia and Anderson Nakano. The work on Section 3 was also developed together with Tiago Etiene, Valdecir Polizelli, Eduardo Tejada and Thomas Ertl. We gratefully thank Luis G. Nonato, an important collaborator for all of these works.

References

[1] A. Adamson and M. Alexa. Ray-tracing point set surfaces. In *Proceedings of the Shape Modeling International*, pages 272–279, 2003.

[2] R. H. Bartels and J. J. Jezioranski. Least-squares fitting using orthogonal multinomials. *ACM Transactions on Mathematical Software*, 11(3):201–217, 1985.

[3] A. Castelo, L. G. Nonato, M. Siqueira, R. Minghim, and G. Tavares. The j1a triangulation: An adaptive triangulation in any dimension. *Computer & Graphics*, 30(5):737–753, 2006.

[4] C. Dietrich, C. Scheidegger, J. Schreiner, J. Comba, L. Nedel, and C. Silva. Edge transformations for improving mesh quality of marching cubes. *IEEE Transactions on Visualization and Computer Graphics*, accepted.

[5] G. Fasshauer and J. Zhang. Iterated approximate moving least squares approximation. In C. A. V. M. A. Leitao and C. A. Duarte, editors, *Advances in Meshfree Techniques*, to appear. Springer, 2007.

[6] J. P. Gois. *Mínimos-Quadrados e Aproximação de Superfície de Pontos: Novas Perspectivas e Aplicações*. PhD thesis, Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo, 2008.

[7] J. P. Gois, A. Nakano, L. G. Nonato, and L. C. Buscaglia. Front tracking with moving-least-squares surfaces. *Journal of Computational Physics*. DOI: 10.1016/j.jcp.2008.07.013

[8] J. P. Gois, V. Polizelli-Junior, T. Etiene, E. Tejada, A. Castelo, T. Ertl, and L. G. Nonato. Robust and adaptive surface reconstruction using partition of unity implicits. In *Brazilian Symposium on Computer Graphics and Image Processing*, pages 95–102, IEEE CS, 2007.

[9] J. P. Gois, V. Polizelli-Junior, T. Etiene, E. Tejada, A. Castelo, L. G. Nonato, and T. Ertl. Twofold adaptive partition of unity implicits. *The Visual Computer*. DOI: 10.1007/s00371-008-0297-x

[10] J. P. Gois, E. Tejada, T. Etiene, L. G. Nonato, A. Castelo, and T. Ertl. Curvature-driven modeling and rendering of point-based surfaces. In *Brazilian Symposium on Computer Graphics and Image Processing*, pages 27–36. IEEE CS, 2006.

[11] G. Guennebaud and M. Gross. Algebraic point set surfaces. In *ACM SIGGRAPH 2007*, Article 23

[12] R. Kolluri. Provably good moving least squares. In *Proc. of the 16th annual ACM-SIAM symposium on Discrete algorithms*, pages 1008–1017, 2005.

[13] D. J. Mavriplis. Revisiting the least-squares procedure for gradient reconstruction on unstructured meshes. Technical Report CR-2003-212683, NASA, 2006.

[14] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel. Multi-level partition of unity implicits. *ACM Trans. Graph.*, 22(3):463–470, 2003.

[15] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*, volume 153 of *Applied Mathematical Sciences*. Springer, 2003.

[16] V. Pratt. Direct least-squares fitting of algebraic surfaces. *SIGGRAPH*, 21(4):145–152, 1987.

[17] E. Tejada. *Towards Meshless Volume Visualization*. PhD thesis, University of Stuttgart, Germany, 2008.

[18] E. Tejada, J. P. Gois, A. Castelo, L. G. Nonato, and T. Ertl. Hardware-accelerated extraction and rendering of point set surfaces. In *EUROGRAPHICS - IEEE VGTC Symposium on Visualization*, 2006.

[19] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y. J. Jan. A Front-Tracking Method for the Computations of Multiphase Flow. *J. Comput. Phys.*, 169:708–759, 2001.