Dynamic Meshes for Accurate Polygonization of Implicit Surfaces with Sharp Features

Yutaka Ohtake
d8011101@u-aizu.ac.jp
The University of Aizu
Aizu-Wakamatsu 968-8580 Japan

Alexander Belyaev
belyaev@u-aizu.ac.jp
The University of Aizu
Aizu-Wakamatsu 968-8580 Japan

Alexander Pasko
pasko@k.hosei.ac.jp
Hosei University
Tokyo 184-8584 Japan

Abstract

The paper presents a novel approach for accurate polygonization of implicit surfaces with sharp features. The approach is based on mesh evolution towards a given implicit surface with simultaneous control of the mesh vertex positions and mesh normals.

1 Introduction

Polygonization of an isosurface of a function of three variables (or implicit surface) includes sampling the function at selected points, estimating the positions of the mesh vertices, and connecting them to form polygons. This usually results in space aliasing (faceting) which is a common problem arising whenever some continuous object is represented by a set of discrete samples. Such faceting is most pervasive for shapes with sharp features (edges, corners, spikes, etc.).

Sharp edges and corners appear naturally if the implicit surface is constructed using set operations (union, intersection, difference). The defining function for such surface can be obtained by applying min/max functions or more general R-functions [21, 22, 25] to the defining functions of the arguments of set operations. If the binary tree of operations is available, then the sharp features can be found by a numerical method that analyzes the functions of both arguments for each operation [31]. However, if the resulting function is evaluated by a procedure making the tree of operations not available to the application, this method is not applicable.

One possible antializing solution consists of adaptive-resolution polygonization, in which the sampling rate adapts to the changing feature size (curvature scale) of a surface being polygonized. A good review of adaptive methods can be found in [5].

Another remedy to reduce apparent faceting is smoothing. Two main approaches for smoothing polygonal meshes consist of geometric signal processing [28] and mesh evolution by curvature flows [9, 24]. However, conventional smoothing schemes blur high curvature regions and, therefore, are not acceptable for antializing surfaces with sharp edges and corners.

New approaches for mesh smoothing with simultaneous preserving and enhancing salient mesh creases were recently proposed in [3, 17, 19] where nonlinear averaging (diffusion) of mesh normals was used as a main constituent, in [7] where an anisotropic weighted mean curvature flow was employed, and in [8] where a curvature-driven mesh evolution was used. See also [27] where a crease enhancement diffusion was developed for image processing purposes and [10] where an anisotropic diffusion was used for feature-preserving height data denoising.

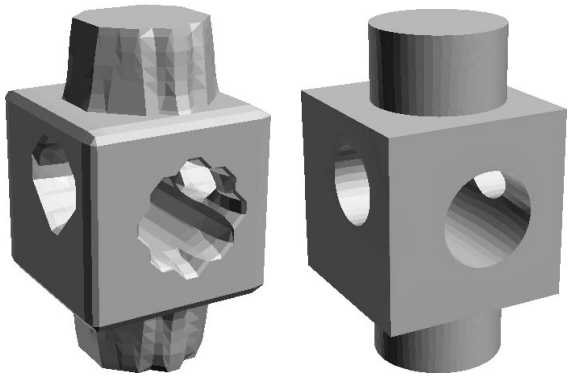

Fig. 1. Left: a triangulated implicit surface constructed by the marching cubes method. Right: after applying the mesh evolution process developed in this article. Flat shading is used.

In this paper, we adapt methods developed in [3, 17, 19] for crease enhancement and combine them with a mesh regularity improving technique [18, 30] in order to fit implicit surfaces with sharp edges by triangle meshes. Consider an

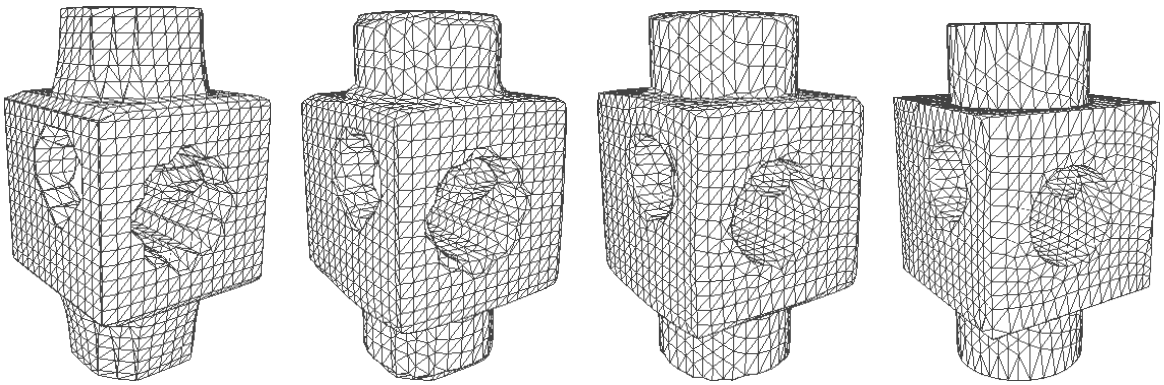

Fig. 2. The leftmost and rightmost images demonstrate wireframe images of the initial and final triangle surfaces exposed in Fig. 1, respectively. The two middle images show intermediate stages of the evolution from the initial mesh to the final mesh. Note that mesh vertices and edges align with sharp features of the implicit surface.

implicit surface and its triangulation obtained by a standard isosurface extraction technique (in our numerical experiments we use a VTK implementation [23] of the marching cubes method [13]).

Given an implicit surface $f(x, y, z) = 0$ and its initial triangulation of the surface, we act on the mesh vertices by three forces. Two forces optimize positions of the mesh vertices according to the values of the function and its gradient at the vertices. The third force improves mesh regularity. Combination of these three forces allows us to achieve an accurate approximation of the implicit surface by a high quality mesh.

Fig. 1 shows an initial triangulation of a surface of a solid with sharp edges (left) and a triangulated surface obtained after applying the method developed in this article.

The leftmost and rightmost images of Fig. 2 demonstrate wireframe images of the initial and final triangle surfaces exposed in Fig. 1, respectively. The two middle images of Fig. 2 show intermediate stages of the evolution from the initial mesh to the final mesh. Note that mesh vertices and edges align with sharp features of the implicit surface.

The idea to use meshes whose vertices act as dynamic particle system was proposed in [11, 12] for 2D and in [15] for 3D and since then was extensively used for segmentation and tracking in multidimensional images and volume data (see, for example, [4, 14, 16] for recent achievements in shape modeling and analysis with deformable contours and surfaces). Recently dynamic meshes were applied to surface extraction from distance volume datasets [30]. However, in contrast to previous works, our method is developed to obtain an accurate mesh approximation of an implicit surface with sharp edges and corners. It is achieved by a simultaneous adjustment of the mesh vertices and mesh normals.

2 Mesh Evolution Toward The Implicit Surface

Consider a family of meshes M^n evolving according to the following equation

$$M^{n+1} = \mathcal{D}(M^n) = M^n + \mathbf{D}(M^n), \tag{1}$$

where $\mathcal{D}$ is a mesh updating operator according to displacement vectors $\mathbf{D}$ defined at the mesh vertices. In many situations (1) can be considered as an explicit approximation of an evolution of a family of smooth surfaces.

Consider an implicit surface $S = \{f(x, y, z) = 0\}$ and its polygonization M^0. We want to construct a mesh updating operator $\mathcal{D}$ such that the mesh evolution process (1) with M^0 taken as the initial condition converges to the surface, $M^n \to S$, as $n \to \infty$.

We implement the operator $\mathcal{D}$ satisfying the above requirement as the superposition of three mesh updating operators (mesh flows) depending on f and local mesh properties:

$$\mathcal{D} = \mathcal{R} \circ \mathcal{Z} \circ \mathcal{N}$$

These local operators are defined by the following vertex update procedures

$$\mathcal{N}\text{-flow}: \quad P_{new} \longleftarrow P_{old} + \mathbf{N}(P_{old}, \nu(P_{old}), f) \tag{2}$$

$$\mathcal{Z}\text{-flow}: \quad P_{new} \longleftarrow P_{old} + \mathbf{Z}(P_{old}, f) \tag{3}$$

$$\mathcal{R}\text{-flow}: \quad P_{new} \longleftarrow P_{old} + \mathbf{R}(P_{old}, \nu(P_{old})) \tag{4}$$

where $\nu(P)$ denotes the set of neighbors of vertex P in M, $\mathbf{N}$, $\mathbf{Z}$, $\mathbf{R}$ are vector functions (forces).

In the subsequent subsections we define these vector functions such that $\mathcal{N}$-flow corrects the mesh normals according to the implicit surface normals, $\mathcal{Z}$-flow pushes

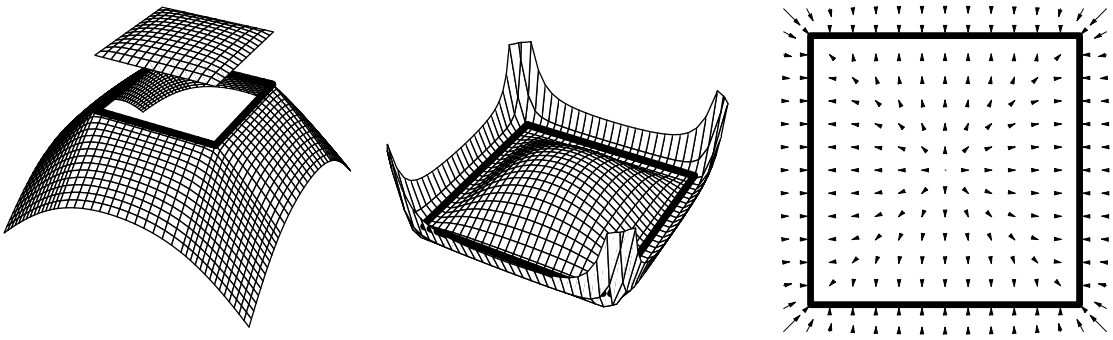

Fig. 3. Left: A square (bold line) defined as the zero level set of a function $w = f(x, y)$ (the function is continuous; the "roof" is separated for visualization purposes). Middle: the graph of the function $w = f(x, y)^2$. Right: the square and a gradient vector field $-\nabla \left[f(x, y)^2\right]$.

mesh vertices toward the implicit surface $f(x, y, z) = 0$, and $\mathcal{R}$-flow equalizes the mesh sampling rate.

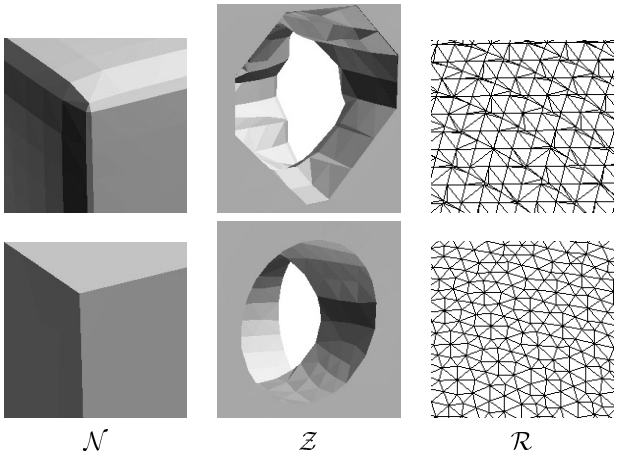

Fig. 4. Top row: fragments of the triangulated block model shown in the left image of Fig. 1. Bottom row: the same fragments after a number of repeated applications of the operators $\mathcal{N}$, $\mathcal{Z}$, and $\mathcal{R}$.

Fig. 4 demonstrates how these mesh updating operators (flows) act on a triangle mesh. The top row demonstrates fragments of the triangulated block model shown in the left image of Fig. 1. The bottom row shows the same fragments after a number of repeated applications of the operators $\mathcal{N}$, $\mathcal{Z}$, and $\mathcal{R}$.

2.1 Moving mesh vertices toward the implicit surface

The simplest way to move mesh vertices toward a given implicit surface $f(x, y, z) = 0$ is to introduce an attracting force defined via the gradient of f. In our implementation we define **Z**-force by

$$\mathbf{Z}(f) = -\tau\nabla \left[f(x, y, z)^2\right] = -2\tau f(x, y, z)\nabla f(x, y, z),$$

where τ is a small positive parameter and ∇ denotes the gradient. Since the implicit surface $f(x, y, z) = 0$ is the minimal level set of the function $w = f(x, y, z)^2$, **Z**-force defined as above moves the mesh vertices towards the surface.

A similar force $-2\tau\,\mathrm{sign}(f)\nabla f$ was used in [29] in connection with adaptive-resolution polygonization of implicit surfaces.

Fig. 3 demonstrates a 2D level set $f(x, y) = 0$ defining a square and the associated **Z**-force.

The value of τ is chosen to ensure numerical stability of (3). Based on an analogy with a stability analysis for the first-order linear partial differential equations (the Courant-Friedrichs-Levy stability criterion [20]) we select τ such that

$$\frac{\max(|f\nabla f|)\tau}{h} < c, \tag{5}$$

where h characterizes the mesh sampling rate and c is a constant independent of the mesh and function f. In our current implementation, h is the step-size in the marching cubes method used to generate the initial mesh and the time step-size τ is defined by

$$\tau = \frac{h}{100\max(|f\nabla f|)}.$$

The gradient ∇f can be computed analytically only for simple functions $w = f(x, y, z)$. So we use standard central-difference formulas to estimate the gradient.

2.2 Mesh optimization according to implicit surface normals

A better approximation of the implicit surface $f(x, y, z) = 0$ is achieved if we modify positions of the mesh vertices such that the triangle normals are close to corresponding implicit surface normals. Fig. 5 illustrates advantages of such mesh optimization. The left image shows approximating a smooth curve by a polygonal line whose vertices are close to the curve. The right image demonstrates the same curve approximated by a polygonal line whose vertices and normals are close to curve points and normals.

Fig. 5. A curve approximated by two polygonal lines. Left: fitting the vertex positions only does not assure a good approximation. Right: fitting the vertex positions and normals simultaneously improves the approximation drastically.

An appropriate error function will be introduced and discussed in Subsection 2.4

Consider a unit vector field

$$\mathbf{m}(x, y, z) = \nabla f(x, y, z)/||\nabla f(x, y, z)||$$

It is orthogonal to the implicit surface $f(x, y, z) = 0$ at the implicit surface points. Fig. 6 presents a 2D example: a square defined as an implicit surface $f(x, y) = 0$ and its associated unit vector field $\mathbf{m}(x, y)$.

The figure demonstrates also that smoothness of the normalized gradient vector field $\mathbf{m}$ depends on functions chosen for basic Boolean operations. We use R-functions [21, 22, 26] since they have better differential properties than the commonly used $\min$ / $\max$ functions.

We want to move mesh vertices such that for every mesh triangle T the mesh normal $n(T)$ becomes closer to $m(T) = \nabla f(C)/||\nabla f(C)||$, where C is the centroid of T. It is achieved by the $\mathcal{N}$ vertex update operator (2) with

$$\mathbf{N}(P, \nu(P), f) = \frac{1}{\sum A(T)} \sum A(T)\mathbf{v}(T),$$

where $\mathbf{v}(T) = \left[\overrightarrow{PC} \cdot \mathbf{m}(T)\right] \mathbf{m}(T)$ is the projection of the vector $\overrightarrow{PC}$ on the $\mathbf{m}(T)$ direction, $A(T)$ denotes the

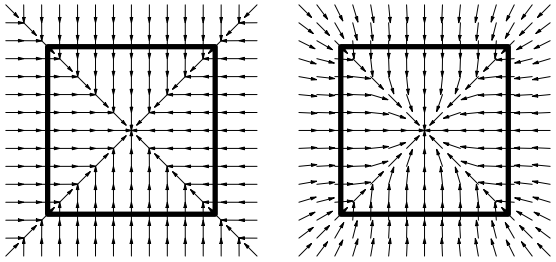

Fig. 6. A square defined as an implicit surface $f(x, y) = 0$ and the vector field $\mathbf{m} = \nabla f(x, y)/||\nabla f(x, y)||$. Left: $\min$ / $\max$ functions were used. Right: R-functions were used.

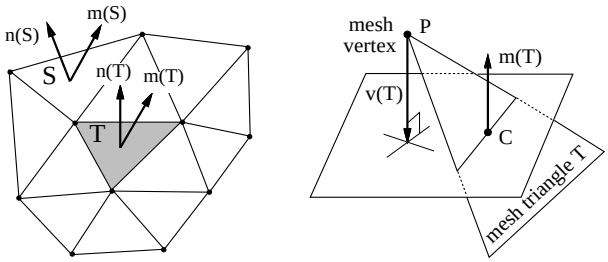

Fig. 7. Left: mesh triangles and their associated vectors. Right: updating vertex position by $\mathcal{N}$.

area of T, and the summations are taken over all P-incident triangles, see Fig. 7.

A similar vertex update operator was used in [3, 17, 19] for crease enhancement purposes.

2.3 Mesh relaxation

To improve mesh regularity we use the tangential component of the Laplacian smoothing flow. The Laplacian flow, in its simplest form, moves repeatedly each mesh vertex by a displacement equal to a positive scale factor times the average of the neighboring vertices. Consider a mesh vertex P and its neighbors $Q_1, \ldots, Q_n$. The Laplacian flow is determined by the umbrella vector $\mathbf{U}$ given by

$$\mathbf{U}(P) = \frac{1}{n} \sum_{i=1}^{n} Q_i - P. \qquad \text{See Fig. 8.}$$

The vertex update operator $\mathcal{R}$ equalizing the mesh sampling rate is defined by (4) with

$$\mathbf{R}\left(P, \nu(P)\right) = C\left[\mathbf{U} - (\mathbf{U} \cdot \mathbf{n})\,\mathbf{n}\right], \qquad (6)$$

where $\mathbf{n}$ is the mesh normal at vertex P and C is a positive constant. In our current implementation we use $C = 0.1$.

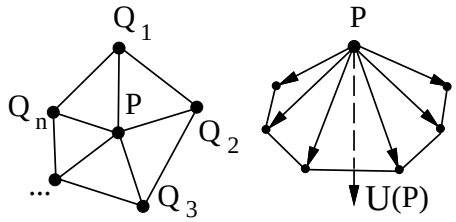

Fig. 8. The umbrella vector $\mathbf{U}(P)$.

Mesh relaxation by the tangential component of the Laplacian flow was first considered in [18]. It was also used in [30] for mesh reparameterization purposes.

Fig. 9 demonstrates how the tangential component of the umbrella operator improves the mesh sampling quality.

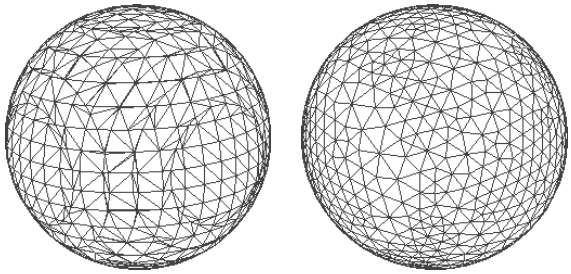

Fig. 9. Left: a polygonized sphere generated by the marching cubes method. Right: the mesh is improved by several iterations of the mesh relaxation $\mathcal{R}$-flow.

However, the $\mathcal{R}$-flow (4) does not preserve sharp edges, as it is demonstrated in Fig. 10.

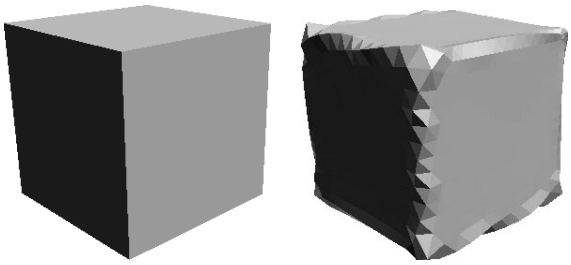

Fig. 10. Sharp edges of a cube are broken by the mesh quality improving $\mathcal{R}$-flow.

Thus dealing with an implicit surface with sharp features we have to switch off the mesh quality improving $\mathcal{R}$-flow when the flow interferes with the mesh-to-surface convergence process.

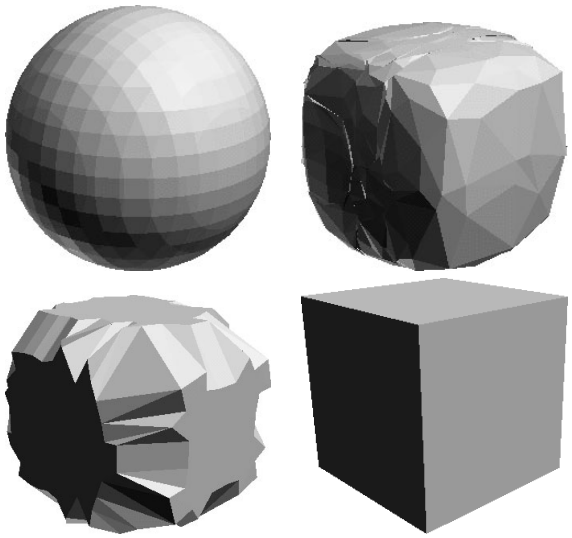

Fig. 11. Transformation of a triangulated sphere (top-left) towards a cube (bottom-right) defined initially as an implicit surface. Sphere deformation by $\mathcal{Z}$-flow only leads to a polygonal surface shown in the top-right image. Sphere deformation by $\mathcal{N}$-flow only leads to a polygonal surface shown in the bottom-left image.

Fig. 11 demonstrates how $\mathcal{N}$-flow (2) and $\mathcal{Z}$-flow (3) transform a polygonized sphere (top-left image) toward a cube (bottom-right image) defined as an implicit surface. If only $\mathcal{Z}$-flow is applied, the sphere transforms to a polygonal surface shown in the top-right image. If the mesh is moved by $\mathcal{N}$-flow only, the sphere transforms to a polygonal surface shown in the bottom-left image. A perfect metamorphosis of the sphere to the cube is achieved if at first we use the composition of the three introduced flows $\mathcal{R} \circ \mathcal{Z} \circ \mathcal{N}$ and then shift to the flow $\mathcal{Z} \circ \mathcal{N}$ according to an analysis of an error function defined and studied in the next subsection see Fig. 12.

2.4 Error analysis

To analyze how close the evolving mesh M^n approaches the implicit surface $f(x, y, z) = 0$, as $n \to \infty$, we introduce error estimator functions ε_v measuring the deviation of the mesh vertices from $f(x, y, z) = 0$ and ε_n characterizing the deviation of the mesh triangle normals from the normalized gradient vector field $\nabla f / ||\nabla f||$.

The deviation of a mesh vertex P from $f(x, y, z) = 0$ is estimated by

$$f(P)^2 \sum_{T \in \mathrm{inc}(P)} A(T)$$

where the sum is taken over $\mathrm{inc}(P)$, all mesh triangles T in-

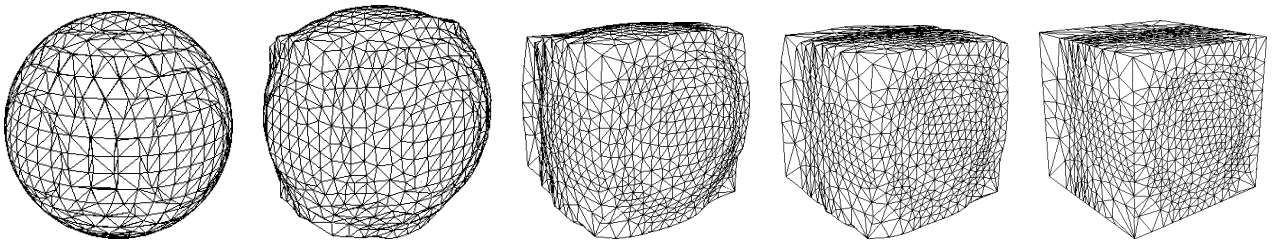

Fig. 12. Sphere-to-cube transformation: $(\mathcal{R} \circ \mathcal{Z} \circ \mathcal{N})$-flow was used for the first 100 iterations and $(\mathcal{Z} \circ \mathcal{N})$-flow was used for the last 100 iterations. Note how good mesh vertices and edges align with the cube edges.

cident with P, and $A(T)$ denotes the area of T. Summation over all the mesh vertices and normalization gives

$$\varepsilon_v = \frac{c}{3 \sum_{all\ T} A(T)} \sum_{all\ P} \left(f(P)^2 \sum_{T \in \mathrm{inc}(P)} A(T) \right)$$

with $c = 1/\max(f(Q)^2)$ where maximum is taken over all the vertices Q of the initial mesh M^0 (in our implementation M^0 is obtained by the marching cubes method).

The deviation of the mesh triangle normals from $\nabla f / ||\nabla f||$ is estimated by

$$\varepsilon_n = \frac{1}{\sum_{all\ T} A(T)} \sum_{all\ T} \left[A(T)\,(1 - |\mathbf{n}(T) \cdot \mathbf{m}(T)|) \right])$$

where the summations are taken over all the mesh triangles T, $\mathbf{n}(T)$ is the unit normal of T, $\mathbf{m}(T) = \nabla f(C)/||\nabla f(C)||$, where C is the centroid of T.

Note that ε_v mimics the sum of squared distances from the mesh vertices to the implicit surface and ε_n emulates a weighted sum (integral) of squared differences between the mesh and surface normals since

$$(\mathbf{n} - \mathbf{m})^2 = 2(1 - \mathbf{n} \cdot \mathbf{m}).$$

Note that $\varepsilon_v + \varepsilon_f$ mimics the squared H^1 norm used widely in the theory of partial differential equations [1] and for error analysis of finite element methods [6].

As we noted in the previous subsection, the mesh motion by the tangential component of the umbrella operator ($\mathcal{R}$-flow) destroys sharp edges, see Fig. 10. We switch off the mesh regularization by $\mathcal{R}$-flow after ε_n has stabilized.

Fig. 13 demonstrates the behaviors of ε_v and ε_n for the block model mesh evolution (see Fig. 1 and Fig. 2) and the sphere-to-cube transformation (see Fig. 12).

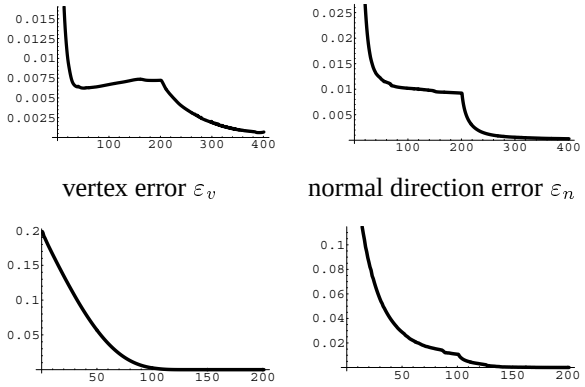

Fig. 13. The graphs of the ε_v (left) and ε_n (right) error functions for the block model mesh optimization (top graphs, see block model mesh evolution in Fig. 2) and the sphere-to-cube transformation (bottom graphs, see the sphere-to-cube transformation in Fig. 12). $(\mathcal{R} \circ \mathcal{Z} \circ \mathcal{N})$-flow was used for the first 200 iterations for the block model mesh optimization and 100 iterations for the sphere-to-cube transformation. Then $(\mathcal{Z} \circ \mathcal{N})$-flow was used.

3 Conclusion, Discussion, and Future Work

We have presented a novel approach for accurate polygonization of implicit surfaces with sharp features. The approach is based on mesh evolution towards a given implicit surface with simultaneous control of the mesh vertex positions and mesh normals.

The presented implementation of the approach can be improved in many directions. For example, if the sampling rate of the initial mesh is comparable with the size of a feature of a given implicit surface, then the feature can be lost during the mesh-to-surface evolution process. Fig. 14 demonstrates a Doraemon [1] model built with HyperFun [2] and then polygonized by the marching cubes method (top-left image) and the result after applying the mesh-to-surface

[1] Doraemon is a round cat-style robot, one of the most famous Japanese manga characters.

evolution process developed in this paper (top-right image). Note that the mesh evolution removes the cat pupils (see the bottom images). One way around this problem consists of adding a mesh subdivision procedure to the evolution.

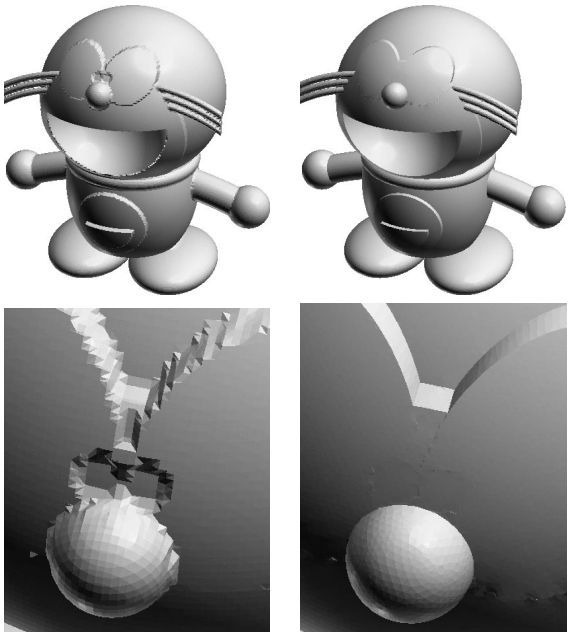

Fig. 14. Top-left: a Doraemon model built with HyperFun [2] and then polygonized by marching cubes. Top-right: the model after applying the the mesh evolution process developed in this paper. Bottom: the pupils and low eyelids are diffused by the evolution.

The speed of convergence of a dynamic mesh towards an implicit surface $f(x, y, z) = 0$ with sharp features depends on differential properties of the function $w = f(x, y, z)$. The min/max functions used commonly [23] for the basic Boolean operations, union and intersection, have poor differential properties. For example, $\max(x, y)$ and $\min(x, y)$ are not differentiable along the line $x = y$. Whereas certain R-functions [21, 22, 26] also define the union and intersection and, in addition, possess much better differential properties. Fig. 6 demonstrates advantages of R-functions over the min/max functions. R-functions are implemented in the HyperFun geometric modeling language and supporting software [2].

The approach developed in this paper can be used to improve polygonizations of isointensity surfaces of 3D voxel data. Fig. 15 demonstrates our preliminary results in this direction.

Another direction for future research consists of mesh offsetting using defining functions of implicit surfaces.

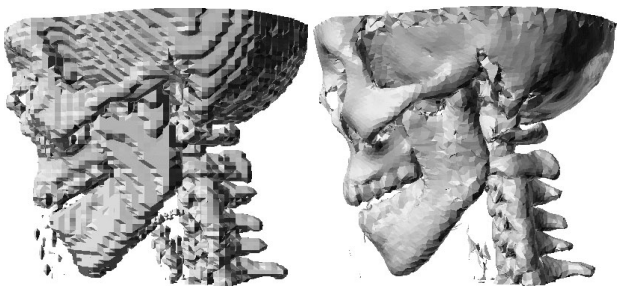

Fig. 15. An initial and improved polygonizations of an isointensity surface of a 3D medical image.

Consider a mesh and a closed implicit surface $f(x, y, z) = 0$ intersecting the mesh. We want to offset the mesh vertices situated inside the surface to one or another surface parts separated by the mesh, as seen in Fig. 16.

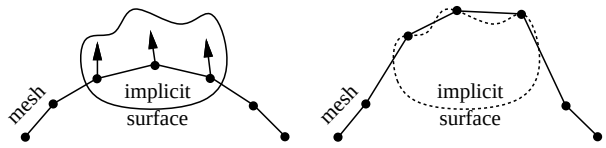

Fig. 16. Mesh offsetting with implicit functions.

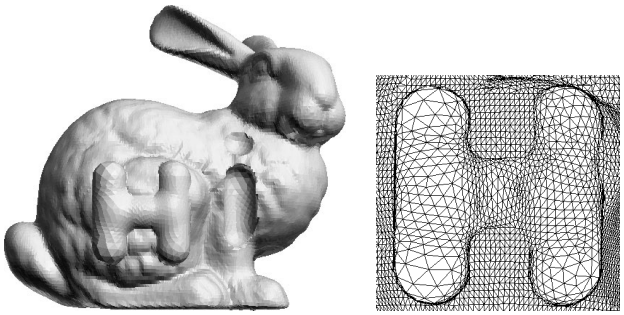

Fig. 17. Left: convex "H" and concave "i" engraved on the Stanford bunny by the offsetting flow (7). Right: zoomed image of convex "H" generated by bunny vertices.

Let us orient the mesh such that the mesh normals point toward a desired surface part. The offsetting flow $\mathcal{O}$ can be defined, for example, as follows

$$\mathcal{O}\text{-flow}: \quad P_{new} \longleftarrow P_{old} + \tau |f \nabla f| \mathbf{n}(P), \qquad (7)$$

where $\mathbf{n}(P)$ is the mesh normal at mesh vertex P and τ is the same as for $\mathcal{Z}$-flow. The flows $\mathcal{Z}$ and $\mathcal{N}$ are switched

off if $\mathbf{Z}$ and $\mathbf{N}$ are opposite to the desired direction, respectively. Fig. 17 demonstrates our preliminary results. The word "Hi" was created using an implicit modeling technique (ball sweeping along letter skeletons). Then the "Hi" implicit surface was placed properly on the Stanford bunny given as a triangle mesh. Finally the mesh vertices situated inside "H" were moved from the bunny and the vertices situated inside "i" were moved inside the bunny. The left image of Fig. 17 shows the result, the right zoomed image shows convex "H" generated by mesh vertices.

Acknowledgments

We would like to thank the anonymous reviewers of this paper for their valuable and constructive comments.

References

[1] R. A. Adams. *Sobolev Spaces*. Academic Press, 1975.

[2] V. Adzhiev, R. Cartwright, E. Fausett, A. Ossipov, A. Pasko, and V. Savchenko. HyperFun project: A framework for collaborative multidimensional F-rep modeling. In J. Hughes and C. Schlick, editors, *Implicit Surfaces '99, Eurographics/ACM SIGGRAPH Workshop*, pages 59–69, September 1999. Visit also http://www.hyperfun.org.

[3] A. G. Belyaev, Yu. Ohtake, and K. Abe. Detection of ridges and ravines on range images and triangular meshes. In L. J. Latecki, D. M. Mount, and A. Y. Wu, editors, *Vision Geometry IX, Proc. SPIE 4117*, pages 146–154, July-August 2000.

[4] A. Blake and M. Isard. *Active Contours*. Springer, 1998.

[5] J. Bloomenthal. Chapter 4: Surface tiling. In J. Bloomenthal, editor, *Introduction to Implicit Surfaces*. Morgan Kaufmann, 1997.

[6] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer, 1994.

[7] U. Clarenz, U. Diewald, and M. Rumpf. Anisotropic geometric diffusion in surface processing. In *Proceedings of IEEE Visualization 2000*, pages 397–405, October 2000.

[8] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Discrete differential-geometry operators in nD. Submitted for publication. Available from http://www.multires.caltech.edu/pubs/pubs.htm.

[9] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. *Computer Graphics (Proceedings of SIGGRAPH '99)*, pages 317–324, 1999.

[10] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Anisotropic feature-preserving denoising of height fields and bivariate data. In *Graphics Interface 2000*, pages 145–152, May 2000.

[11] R. Durbin and D. Willshaw. An analogue approach to the traveling salesman problem using an elastic net method. *Nature*, 326:348–358, 1987.

[12] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. Journal of Computer Vision*, 1(4):321–331, 1987.

[13] W. E. Lorensen and H. E. Cline. Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics (Proceedings of SIGGRAPH '87)*, 21(3):163–169, 1987.

[14] T. McInerney and D. Terzopoulos. Topology adaptive deformable surfaces for medical image volume segmentation. *IEEE Transactions on Medical Imaging*, 18(10):840–850, October 1999.

[15] J. V. Miller, D. E. Breen, W. E. Lorensen, R. M. O'Bara, and M. J. Wozny. Geometrically deformed models: A method for extracting closed geometric models from volume data. *Computer Graphics (Proceedings of SIGGRAPH 91)*, 25(4):217–226, 1991.

[16] J. Montagnat, H. Delingette, N. Scapel, and N. Ayache. Representation, shape, topology, and evolution of deformable surfaces. application to 3D medical image segmentation. Research Report RR-3954, INRIA Sophia Antipolis, France, May 2000.

[17] Yu. Ohtake and A. G. Belyaev. Nonlinear diffusion of normals for stable detection of ridges and ravines on range images and polygonal models. In *MVA2000, IAPR Workshop on Machine Vision Applications*, pages 497–500, Tokyo, November 2000.

[18] Yu. Ohtake, A. G. Belyaev, and I. A. Bogaevski. Polyhedral surface smoothing with simultaneous mesh regularization. In R. Martin and W. Wang, editors, *Geometric Modeling and Processing 2000*, pages 229–237, Hong Kong, April 2000.

[19] Yu. Ohtake, A. G. Belyaev, and I. A. Bogaevski. Mesh regularization and adaptive smoothing. *Computer-Aided Design*, Accepted for publication.

[20] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipies in C: The Art of Scientific Computing*. Cambridge University Press, 1993.

[21] V. L. Rvachev. *The Methods of the Algebra of Logic in Mathematical Physics*. Naukova Dumka, Kiev, 1973. In Russian.

[22] V. L. Rvachev. *Theory of R-functions and Some of Its Applications*. Naukova Dumka, Kiev, 1982. In Russian.

[23] W. Schroeder, K. Martin, and W. Lorensen. *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics*. Prentice Hall, 1998.

[24] J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[25] V. Shapiro. Real functions for representation of rigid solids. *Computer Aided Geometric Design*, 11(2):152–175, 1994.

[26] V. Shapiro and I. Tsukanov. Implicit functions with guaranteed differential properties. In *Proceedings of the Fifth Symposium on Solid Modeling (SOLID MODELING '99)*, pages 852–857, Ann Arbor, Michigan, June 1999.

[27] A. F. Solé, A. López, and G. Sapiro. Crease enhancement diffusion. IMA Preprint Series # 1707, Institute for Mathematics and its Applications, April 2000.

[28] G. Taubin. Geometric signal processing on polygonal meshes. In *EUROGRAPHICS '2000: State of the Art Reports*, pages 107–117, August 2000.

[29] L. Velho. Simple and efficient polygonozation of implicit surfaces. *Journal of Graphics Tools*, 1(2):5–24, 1996.

[30] Z. J. Wood, M. Desbrun, P. Schröder, and D. Breen. Semi-regular mesh extraction from volumes. In *Proceedings of IEEE Visualization 2000*, pages 275–282, October 2000.

[31] B. Wyvill and K. van Overveld. Polygonization of implicit surfaces with constructive solid geometry. *International Journal of Shape Modelling*, 2(4):257–274, 1996.