



Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section: CAD and graphics

A novel region-growing based iso-surface extraction algorithm

Yongjian Xi*, Ye Duan

Department of Computer Science, University of Missouri at Columbia, MO 65203, USA

A B S T R A C T

In this paper, we propose a new region-growing based iso-surface extraction algorithm that can generate high-quality curvature-adaptive semi-regular meshes, preserve sharp features and will extract all the disjoint components of the iso-surface. More importantly, in this paper, we propose a novel normal consistency constraint that ensures the intersection of the Delaunay sphere of the new triangle and the iso-surface is a topological disk, an important property that makes the new algorithm very robust when dealing with large scale of volumetric datasets of complex topology and geometry.

© 2008 Published by Elsevier Ltd.

1. Introduction

Implicit surfaces have become very popular in computer graphics, computer vision, geometric modeling, and computer animation, etc. Their advantages are numerous, ranging from ease of use to compact storage and powerful shape blending operations. They are particularly convenient for modeling smooth objects of arbitrary topology and geometry such as objects with holes, branches, and handles. Nonetheless, for many applications, an explicit polygonal representation of the implicit surface, defined by the zero-level iso-surface of a three-dimensional (3-D) scalar field is still necessary and preferred. On the other hand, recent technical breakthroughs in new imaging modalities such as CT, MRI, and ultrasound as well as other 3-D scanning technologies have given rise to massive volumetric datasets. How to extract and reconstruct the shape of 3-D objects from these datasets accurately and efficiently remains to be both extremely challenging and significant in computer graphics, medical imaging, and visualization, etc. In a nutshell, a good iso-surface extraction algorithm should meet the following three criteria: (1) *geometrically accurate*, i.e. the extracted polygonal mesh should be as close to the original shape as possible and preserves small features; (2) *topologically correct*, i.e. the polygonal mesh is homeomorphic to the original shape; and (3) *representation efficient*, i.e. the sampling rate of the polygonal mesh is adaptive to the local geometric properties (such as curvature) and maintains a good triangle aspect ratio thus no small thin triangles will be generated.

To date, there have been a lot of methods proposed for extracting 3-D surface from volumetric datasets. They can be classified into two main categories: volumetric-based cell

decomposition method and surface-based region growing method. The most popular volumetric-based approach is the Marching Cubes algorithm proposed by Lorensen and Cline [1] in 1987. In their algorithm, a cube is bounded by eight voxels located on two adjacent slices. Each grid point is coded as either inside or outside the object w.r.t. the surface-defining threshold. Based on the configuration of vertices that lie inside and outside the object, the cube is triangulated. Because Marching Cubes algorithm generates at least one triangle per voxel through which the surface passes, this may result in an enormous number of extremely small or thin triangles. Postprocessing algorithms such as mesh optimization [2] or mesh simplification [3] are often needed to improve the mesh quality and reduce the mesh size.

Over the years, many variants of the Marching Cubes algorithm have been published [4,5]. To resolve ambiguities for cube types, Montani et al. [6] proposed a lookup table that prevents holes by consistently separating positive vertices on ambiguous faces. Nielson et al. [7] made use of bilinear contour topology to resolve ambiguities on boundary faces. Cignoni et al. [8] extended this concept by examining the trilinear interpolant in a cell's interior and completely determining piecewise trilinear isosurface topology. Another powerful method to resolve ambiguity is adaptive subdivision, such as the octree technique. In principle, subdivision-based, adaptive techniques also improve the accuracy and efficiency of the surface approximation through the use of spatial partitioning [9,10]. Kobbelt et al. [11] presented an extended marching cube algorithm for feature-sensitive surface extraction from volumetric data using directed distance fields, in order to reduce the alias effect.

Gibson et al. [12] proposed another type of volumetric-based algorithm “dual methods” that generates one vertex lying on or near the contour for each cube that intersects the contour. For each edge in the grid that exhibits a sign change,

* Corresponding author.
E-mail address: yxy2@mizzou.edu (Y. Xi).

the vertices associated with the four cubes that contain the edge are joined to form a quad. Topologically, the mesh generated by the dual methods is the dual of the mesh generated by the Marching Cubes algorithm. Since the vertices of the mesh can move within the cube instead of being restricted to the edges of the grid as in Marching Cubes algorithm, the mesh quality is generally better. Recently, several new “dual methods” algorithms [13–16] have been proposed that can either represent sharp feature [13,14], generate smoother meshes [15], or extract small thin features adaptively without excessive subdivision of the underlying volumetric grid [16].

While most of existing iso-surface extraction algorithms are based on cell decomposition, Hilton et al. proposed a surface-based region growing algorithm—the marching triangles algorithm [17]. Comparing with the aforementioned volumetric-based algorithms, the marching triangles algorithm can generate much higher quality evenly-sized and quasi-equilateral meshes by iteratively joining new triangles to the current region boundary. The original marching triangles algorithm however does not adapt well to local surface properties and thus requiring large numbers of small triangles to approximate surfaces with large variations in curvature. Recently, researchers have proposed several curvature-adaptive iso-surface extraction algorithms [18–20] based on the concept of the marching triangles algorithm [17].

In this paper, we propose a new region-growing based iso-surface extraction algorithm that is also based on the concept of the marching triangles algorithm. The main contribution of this paper is that we propose a novel normal consistency constraint that ensures the intersection of the Delaunay sphere of the new triangle and the iso-surface is a topological disk, an important property which is lacking in the original marching triangles algorithm [17] and its existing variants [18–20]. In addition, by employing a prioritized seed initialization scheme, the new algorithm can generate meshes (e.g. Fig. 1) that are not only curvature-adaptive, but also semi-regular, which may be more preferable for most applications. Moreover, it will extract all the disjoint components of the iso-surface, and can preserve sharp features.

2. Algorithm

The entire pipeline of the algorithm (Fig. 2) consists of the following five main steps:

1. Boundary particle sampling and classification.
2. Feature position extraction.
3. Prioritized seed initialization.
4. Front propagation.
5. Visited boundary particle identification.

After the first two preprocessing steps (Steps 1 and 2), the algorithm will loop from Steps 3 to 5 until all the iso-surfaces are extracted.

2.1. Boundary particles sampling and classification

The first step of the algorithm is boundary particles sampling. Fig. 3 shows a 2-D illustration. Boundary particles (empty circles) are sampled at the intersections of the iso-surface (dashed lines) and the edges (solid lines) of the boundary cells (i.e. cells that intersects the iso-surface). These boundary particles will then be further classified into n classes according to their local geometric properties such as curvature (we use the maximum absolute principle curvature in this paper). To estimate these geometric properties such as gradient and curvature, instead of using the commonly used trilinear interpolation, we use a more accurate cubic interpolation filter as suggested in [21]. The classified boundary particles will then be inserted into the seeding priority list and will be used for seed initialization. The whole algorithm will stop only after all the boundary particles are visited by one of the propagating front (Section 2.5). This will ensure all the disjoint components of the iso-surface will be extracted.

2.2. Feature positions extraction

Our algorithm can preserve sharp features. For example, if Hermite type of datasets (i.e. datasets with exact intersection points and normals) are available, we can extract the sharp feature points/edges by solving the quadratic error function (QEF) as

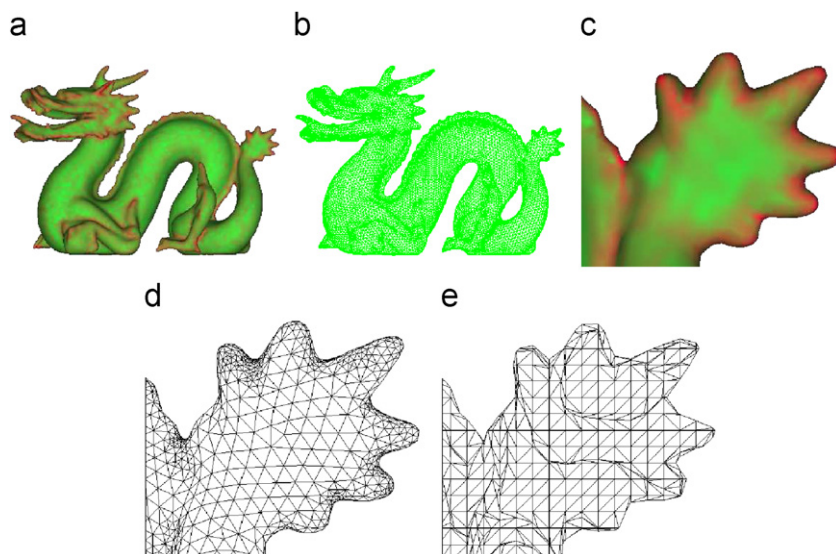


Fig. 1. Semi-regular curvature adaptive iso-surface extraction. (a) Curvature map of the volumetric dragon; (b) extracted mesh; (c) close-up view of the curvature map; (d) close-up view of the corresponding mesh; (e) close-up view of mesh generated by Marching Cubes.

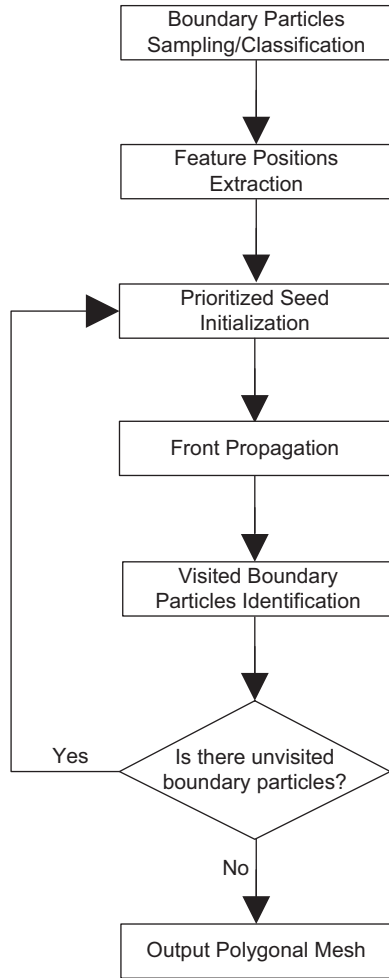


Fig. 2. Algorithm flow chart.

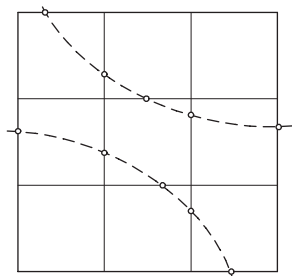


Fig. 3. Boundary particles (empty circles) are sampled at the intersections of the iso-surface (dashed lines) and the edges of the boundary cells (solid lines). They will then be classified according to their local geometric properties (e.g. curvature) and will be used for seed initialization.

suggested in [11]–[13]. The extracted feature positions will then be inserted with the highest priority into the seeding priority list and will be used for seed initialization.

2.3. Prioritized seed initialization

Seed initialization will be conducted in a hierarchical fashion by placing all the candidate seeding positions into a priority list. The seeding priority list can be implemented as a bucket linked list (Fig. 4). Extracted feature positions are assigned the highest

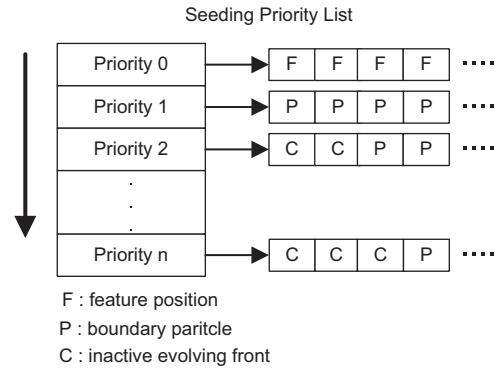


Fig. 4. Prioritized seed initialization. Candidate positions for seed initialization are organized into a priority list. Extracted feature positions are assigned the highest priority and are put in the top row (Priority 0) of the priority list. Boundary particles and pointers of all inactive evolving fronts are put into the next n rows (Priority 1 to Priority n) of the priority list according to the classifications.

priority and are put in the top row (Priority 0) of the priority list. Boundary particles are put into the next n rows (Priority 1 to Priority n) of the priority list according to their geometric properties such as curvature. In this paper, we use the maximum absolute principle curvature to classify the boundary particles. A suggested step size is associated with each row in the priority list with the largest step size in the top two rows and decreasing step sizes (e.g. one half of the previous step size) in the following $n - 1$ rows. In general, the higher curvature the boundary particle has, the lower priority it will be and the smaller step size it will be assigned to. In our implementation, we initialize the priority list with three rows: the top row (Priority 0) stores all the extracted features; the second row (Priority 1) stores all the boundary particles whose curvature is below the average curvature of all the boundary particles; the third row (Priority 2) stores all the boundary particles whose curvature is greater or equal to the average curvature of all the boundary particles. The top row and the second row will start with the original step size, while the third row will have one half of the original step size. When an propagating front seeded from the Priority k row of the priority list becomes inactive, it will be frozen and the pointer of the front will be inserted into the beginning of the Priority $k + 1$ row of the priority list. This way, when this front is later reactivated, it will evolve at a step size smaller than its current step size (e.g. one half).

This prioritized seed initialization scheme will make the propagating front grows in a layer-by-layer fashion so that at each layer the majority of the triangles will be about equilateral. Hence our algorithm can generate meshes not only curvature adaptive, but also semi-regular (i.e. the majority of the vertices have valence 6).

2.3.1. New triangle creation

Given a candidate seeding position, to create a new triangle, we will need to create the base edge of the new triangle first. For extracted feature position, the adjacent feature information is always available and will be used to create the base edge. Fig. 5 shows an illustration. For a boundary particle m with step size s , we will randomly create a line segment uv of length $\frac{2\sqrt{3}}{3}s$ on the tangent plane of m , centered at m with two endpoints u and v , and then project the two points u and v onto the iso-surface as u' and v' , respectively. The new line segment $u'v'$ will then serve as the base edge for this new triangle. Given a base edge $u'v'$ and step size s , either obtained from the above seeding positions or from the boundary edge of an evolving

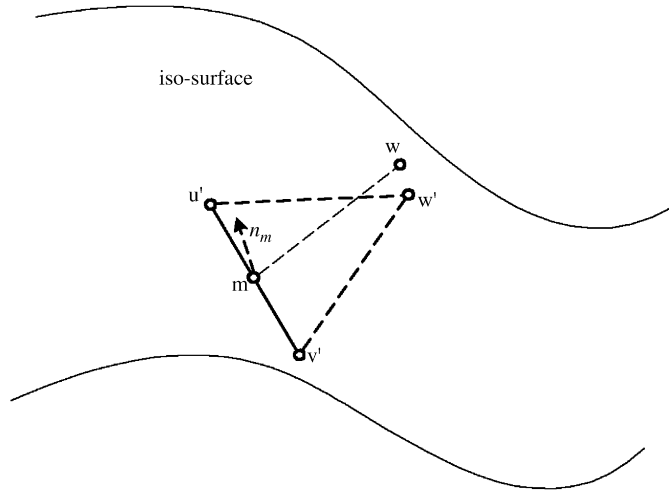


Fig. 5. New triangle creation: a new triangle $u'v'w'$ is created by placing the third vertex w' on the iso-surface (Section 2.3.1).

front, to create a new triangle, we will need to create the third vertex. First the middle point m between u' and v' is calculated. A vertex w is then created at distance s away from m along the direction of the cross product between $u'v'$ and \vec{n}_m . \vec{n}_m is the outward-orienting unit vector obtained by the normalized gradient of the iso-surface at position m . w is then projected to w' on the iso-surface and will be the third vertex of the new triangle $u'v'w'$. Since the edge length of $u'v'$ is $\frac{2\sqrt{3}}{3}s$, and the edge length of mw is s , hence the triangle $u'v'w$ will be equilateral and the corresponding new triangle $u'v'w'$ will be very close to equilateral.

2.3.2. Projection

To project a given point onto the iso-surface (e.g. w and w' in Fig. 5), we employed the iterative Newton–Raphson root-finding method as also used by Co et al. in [22]. Let $\phi(x)$ represents the density function of the volume data, the iso-surface of the volume data is then implicitly defined as $\{x|\phi(x) = 0\}$. The projection of a vertex v onto the iso-surface can be found by a few steps of iterations along the gradient direction $\nabla\phi(x)$ of the density function $\phi(x)$ until converge:

$$v_k = v_{k-1} - \frac{\phi(v_{k-1})}{\|\nabla\phi(v_{k-1})\|^2} \nabla\phi(v_{k-1}), \quad k = 1, \dots, n \quad (1)$$

with $v_0 = v$. The equilibrium position v_n obtained from this Newton–Raphson root finding method however, does not always stay on the iso-surface. Hence, in this paper, we propose to conduct a finite number (e.g. three) iterations of additional line search after the Newton–Raphson root finding method converges to position v_n . More specifically, we will start from v_n and search along the gradient direction of v_n until two consecutive points on the searching sequence v_{n_i} and $v_{n_{i+1}}$ are found to be on the different sides of the iso-surface. A binary search is then conducted between the two points v_{n_i} and $v_{n_{i+1}}$ until the final projection point v_p is found, which can be as close to the iso-surface as needed. See Fig. 6 for an illustration. If after a finite number of line search we can not find two consecutive points on the search direction that are on the opposite side of the iso-surface, the projection will be marked as invalid and the base edge will be frozen, which will be reactivated later at a smaller step size.

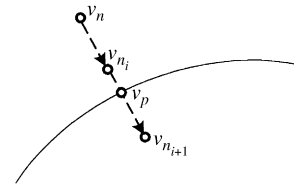


Fig. 6. Enhanced projection scheme: after the equilibrium position v_p is obtained from the Newton–Raphson root finding, several additional iterations of line search is conducted along the gradient of v_n to find the final projection point v_p which can be as close to the iso-surface as possible (Section 2.3.2).

2.4. Front propagation

After a new triangle is created (Section 2.3.1), several tests need to be conducted before it can be added into the existing propagating front. One of the test is to ensure the Delaunay face property, which is first proposed in the original marching triangles algorithm [17]. However, the Delaunay face property will work only if the intersection of the Delaunay sphere and the iso-surface is a topological disk. An important property lacking in the original marching triangles algorithm [17] and its existing variants [18–20]. This problem is fixed in this paper by our new normal consistency constraint to be explained later in this section.

2.4.1. Delaunay face property

We will briefly review the Delaunay face property in this section. For more details, please refer to the original paper [17]. In a nutshell, the Delaunay face property is: “A new triangle xyv with new vertex v can be created if and only if the Delaunay sphere of the triangle xyv does not contain any existing mesh vertices, and any Delaunay spheres of existing triangles do not contain the new vertex v ”. If all the triangles satisfy the Delaunay face property, then the final triangulation will be the Delaunay triangulation, which is the optimal triangulation [17].

To enforce the Delaunay face property, our algorithm performs the following two steps as illustrated in Fig. 7. First, for new vertex v , we will check whether it is enclosed by the Delaunay spheres of any existing triangles. If yes, then one of such triangle’s boundary vertices (e.g. a) will be used as the candidate for the third vertex of new triangle (Fig. 7(a)).

The second step (Fig. 7(b)) is to check whether there are any existing boundary vertices enclosed by the Delaunay sphere of the new triangle xyv . If yes, then one of such boundary vertices (e.g. a) will be used as the candidate and conduct this second step again until the Delaunay sphere of the new triangle does not include any existing vertices.

The above two tests however does not guarantee no thin triangles will be created as is illustrated by Fig. 7(c). Here the new vertex v is not included by any Delaunay sphere of existing triangles and the Delaunay sphere of the new triangle does not include any existing vertices. However since v is very close to a , a thin triangle (passing through v and a) will be created in later steps. To avoid creating such thin triangles and thus further improving the triangulation quality, in the first step when checking whether the new vertex v is enclosed by the Delaunay spheres of any existing triangles, we will increase the radius of the sphere by an extra d (Fig. 7(c)). d is the user-defined minimum edge length of the triangles. By enlarging the sphere by an extra d , we ensure no thin triangles whose minimum edge length is less than d will be created. If the new triangle xyv passes these tests, it will be tested for the normal consistency constraint to be explained in the following section.

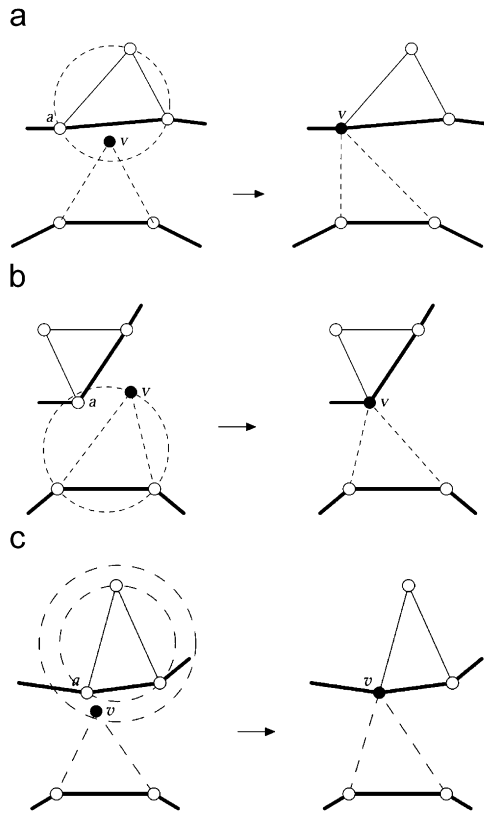


Fig. 7. Enforcing Delaunay face property. (a) New vertex v is inside the Delaunay sphere of an existing boundary triangle, and is snapped into the existing boundary vertex a ; (b) The Delaunay sphere of the new triangle encloses an existing boundary vertex a , hence the vertex v is snapped into vertex a ; (c) The new vertex v is very close to an existing boundary vertex a , hence it is snapped into vertex a to avoid creating a thin triangle passing through vertices a and v .

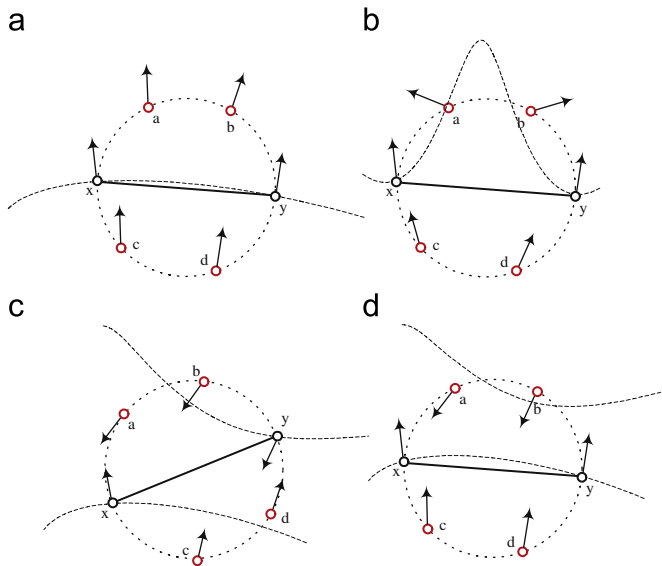


Fig. 8. A 2-D illustration of the normal consistency constraint. A new triangle (a new edge in this case) is valid only if it passes the Normal consistency constraint. This is used to ensure that the intersection of the Delaunay sphere (dashed circle) of the current edge (solid line) and the iso-surface (dashed curve) is a topological disk (Section 2.4.2). Among the four cases shown ((a)–(d)), only the edge in (a) passes the Normal consistent constraint.

2.4.2. Normal consistency constraint

The above Delaunay face property, however, will only work if the intersection of the Delaunay sphere and the iso-surface is a

topological disk. One of the main contribution of this paper and the main difference between this paper and other marching triangles algorithms [17–20] is that we propose a specific normal consistency checking step to enforce that the intersection of the Delaunay sphere of the new triangle and the iso-surface is a topological disk. More specifically, given a new triangle, we will uniformly sample its Delaunay sphere and calculate the maximum normal variation among the sampled positions. In particular, besides the three vertices of the new triangle, we will sample three extra points on each half of the Delaunay sphere divided by the new triangle. If the maximum normal variation among the sampled positions is less than a threshold, the new triangle is valid and can be added to existing mesh, otherwise the new triangle will not be created, and the base edge of the new

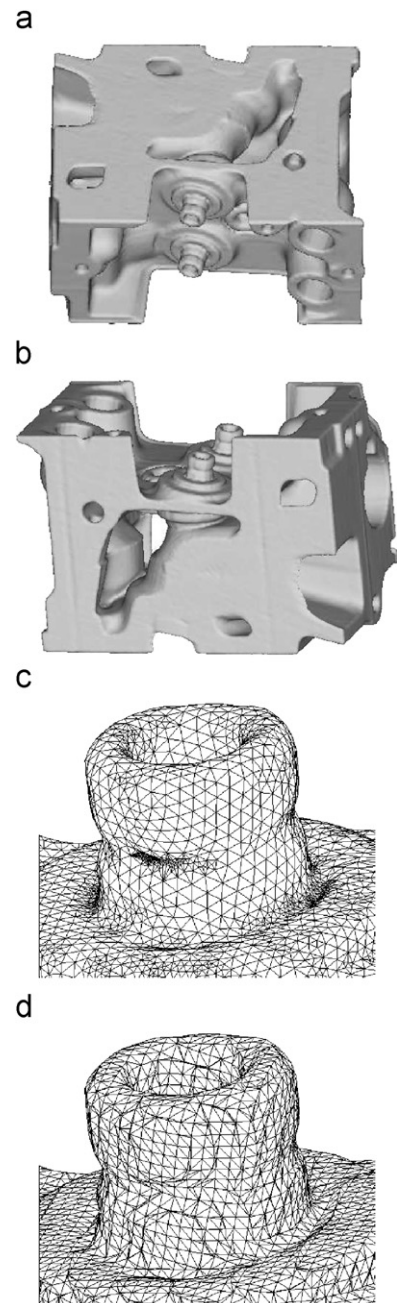


Fig. 9. Iso-surface extraction of a CT data of Engine. (a) and (b) are the front and back view of the smooth rendered mesh. (c) is a close-up view of mesh generated by our algorithm. (d) is a close-up view of mesh generated by Marching Cubes.

triangle will be marked as inactive and will be frozen. Based on our experiment, the threshold of $\pi/3$ has worked very well. When all the boundary edges of a propagating front becomes inactive, the whole front will become inactive and will be inserted into the seeding priority list at a row one level lower than its current priority (Section 2.3). Fig. 8 shows a 2-D illustration of the normal consistency constraint. Among the four cases shown (Fig. 8(a)–(d)), only the case in Fig. 8(a) passes the normal consistent constraint, i.e. the intersection of the Delaunay sphere (dashed circle) of the current edge (solid line) and the iso-surface (dashed curve) is a topological disk.

2.5. Identify visited boundary particles

After a new triangle is created, all the boundary particles enclosed by the triangle's Delaunay sphere will be in the topological disk generated by the intersection of the Delaunay sphere of the new triangle and the iso-surface and will be propagated by the front associated with the new triangle. These boundary particles will be marked as visited and will be removed from the seeding priority list (Section 2.3).

3. Experimental results

Figs. 1 and 9–14 demonstrate some of the experimental results obtained by our new algorithm. The dark colored shape is rendered by smooth shading, while the green lines show the wireframe view of the mesh. For example, from Fig. 10(b), we can clearly see the high quality of the mesh that is adaptive to the local shape curvature. The new algorithm can also recover sharp features as can be seen in Fig. 11. The sharp features are detected similar to [11], [13]. Fig. 14 shows a side-by-side comparison between our new algorithm and the original marching triangle algorithm [17]. As we can see, by enforcing normal consistency constraints proposed in this paper, our algorithm accurately extract the iso-surface in the regions around the small extrusion in the data (Fig. 14(a)). On the other hand, the original algorithm [17] can not correctly extract the iso-surface and an self-intersection will be created (Fig. 14(b)). Table 1 summarizes the statistics of the examples, including the (x, y, z) dimensions of the input volume data, the model complexity (number of vertices, number of edges, and number of faces), and the running time (measured in seconds). All the experiments are conducted on a Pentium M 1.6 GHz Notebook PC with 1 GB RAM. As can be seen

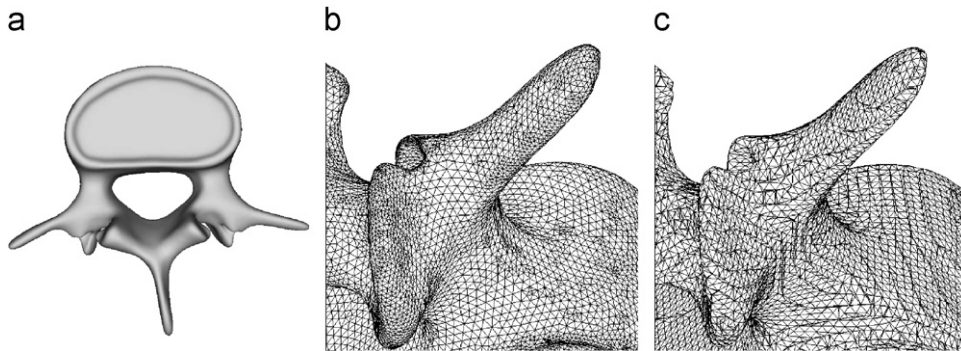


Fig. 10. Iso-surface extraction of a CT data of vertebra phantom. (a) Smooth rendering of the extracted vertebra; (b) a close-up view; (c) a close-up view of mesh generated by Marching Cubes.

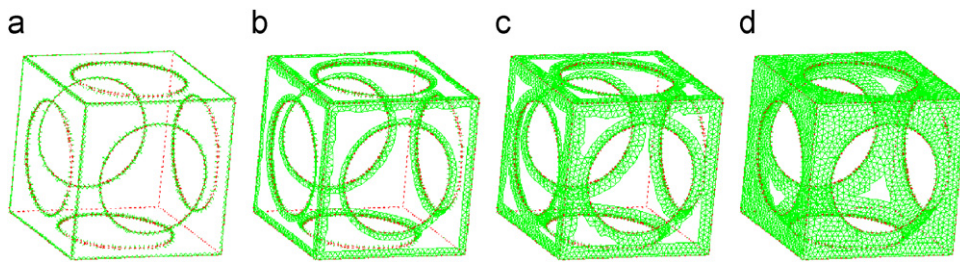


Fig. 11. Iso-surface extraction of a volume data with sharp edges (shown as red lines). (a) is the seed initialization step; (b) and (c) are two snapshots during the front propagation; (d) is the final mesh.

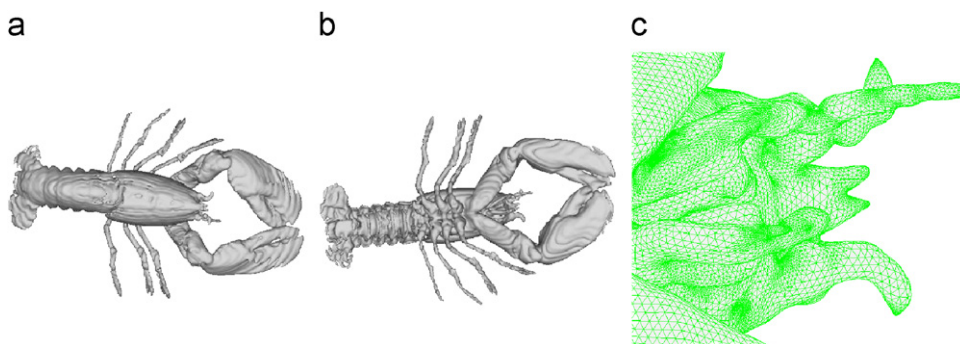


Fig. 12. Iso-surface extraction of a CT data of a lobster. (a) and (b) are the front and back view of the smooth rendered mesh; (c) is a close-up view of (b).

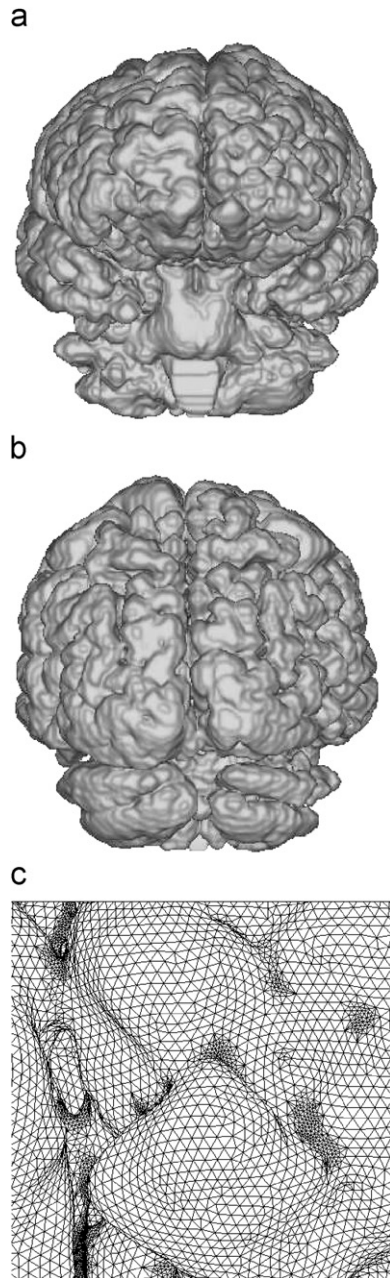


Fig. 13. Iso-surface extraction of a MRI data of brain. (a) Front view; (b) back view; (c) close up view of the generated mesh.

from the running time table, the computational complexity of the new algorithm is linear of the size of the extracted mesh, which is similar to other region growing based algorithms [17–20].

4. Conclusions

In this paper, we propose a new region-growing based iso-surface extraction algorithm that significantly improves the original marching triangles algorithm. Besides using the Delaunay face property to check for existing triangulation, local iso-surface information is also checked by the new normal consistency constraints to ensure the intersection of the Delaunay sphere and iso-surface is a topological disk. As can be seen from the examples, the new algorithm is very robust for data sets of complex topology and geometry. In addition, by employing a prioritized seeding

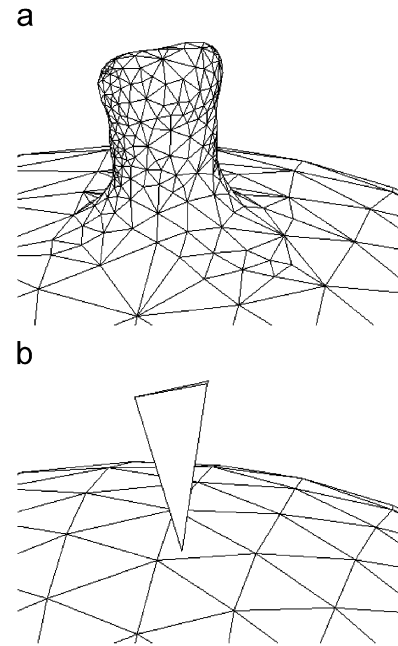


Fig. 14. A comparison between our method and the original marching triangle algorithm [17]. By enforcing normal consistency constraints, our algorithm can accurately extract the iso-surface in the regions around the small extrusion in the data (a). On the other hand, the original algorithm [17] cannot correctly extract the iso-surface and a self-intersection will be created (b).

Table 1
Running time information

Dataset	Volume size	Times (s)	Vertices	Edges	Faces
Dragon	149 × 129 × 89	95	26 724	159 807	53 269
Vertebra	132 × 124 × 56	218	62 494	370 938	123 646
Engine	256 × 256 × 128	1055	182 598	1 095 522	365 174
Lobster	128 × 128 × 128	2520	549 118	2 803 527	934 509
Brain	256 × 256 × 124	2835	638 081	3 613 359	1 204 453

initialization scheme, our algorithm can generate high-quality semi-regular meshes with different level-of-details and can preserve sharp features. In the future, we would like to extend the algorithm to parallel seed initialization and out-of-core computation which will further improve its performance.

Acknowledgments

This work is supported in part by the Thompson Center for Autism and Neurodevelopmental Disorders and the Shumaker Endowment in Biomedical Informatics Graduate Fellowship.

References

- [1] Lorensen WE, Cline HE. Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics (Proceedings of SIGGRAPH 1987)* 1987;21:163–9.
- [2] Hoppe H, Deroose T, Duchamp T, McDonald J, Stuetzle W. Mesh optimization, in: *Proceedings of the SIGGRAPH 93 conference*, 1993. p. 19–26.
- [3] Garland M, Heckbert PS. Surface simplification using quadric error metrics, in: *Proceedings of the SIGGRAPH 97 conference*, 1997. p. 209–16.
- [4] Nielson GM. On marching cubes, in: *IEEE transactions on visualization and computer graphics*, vol. 9, no. 3, 2003. p. 283–97.
- [5] Ho C-C, Wu F-C, Chen B-Y, Chuang Y-Y, Ouhyoung M. Cubical marching squares: adaptive feature preserving surface extraction from volume data, in: *Computer graphics forum*, vol. 24, no. 3, 2005. p. 537–45.

- [6] Montani C, Scateni R, Scopigno R. A modified look-up table for implicit disambiguation of Marching Cubes. *The Visual Computer* 1994;10(6):353–5.
- [7] Nielson G, Hamann B. The asymptotic decider: resolving the ambiguity in marching cubes, in: *IEEE visualization*, 1991. p. 83–91.
- [8] Cignoni P, Ganovelli F, Montani C, Scopigno R. Reconstruction of topologically correct and adaptive trilinear iso-surfaces. *Computers and Graphics* 2000; 22(1):312.
- [9] Wilhelms J, Gelder AV. Octrees for faster isosurface generation. *ACM Transactions on Graphics* 1992;11(3):201–27.
- [10] Wyvill G, McPheeters C, Wyvill B. Data structure for soft objects. *The Visual Computer* 1988;2(4):227–34.
- [11] Kobbelt LP, Botsch M, Schwanecke U, Seidel H-P. Feature sensitive surface extraction from volume data. *Computer Graphics (Proceedings of SIGGRAPH 2001)* 2001:57–66.
- [12] Gibson S. Using distance maps for accurate surface representation in sampled volumes, in: *IEEE symposium on volume visualization*, 1998. p. 23–30.
- [13] Ju T, Losasso F, Schaefer S, Warren J. Dual contouring of hermite data. *ACM Transactions on Graphics* 2002;21(3):339–46.
- [14] Ohtake Y, Belyaev A. Dual/primal optimization of polygonized implicit surfaces with sharp features. *Journal of Computing and Information Science in Engineering* 2002;2:23–45.
- [15] Nielson GM. Dual marching cubes, in: *15th IEEE visualization 2004 (VIS'04)*, 2004. p. 489–96.
- [16] Schaefer S, Warren J. Dual marching cubes: primal contouring of dual grids, in: *Proceedings of pacific graphics 2004*, 2004. p. 70–6.
- [17] Hilton JIA, Stoddart A, Windeatt T. Marching triangles: range image fusion for complex object modeling, in: *International conference on image processing*, 1996.
- [18] Akkouche S, Galin E. Adaptive implicit surface polygonization using marching triangles. *Computer Graphic Forum* 2001;20(2):67–80.
- [19] Karkanis T, Stewart A. Curvature-dependent triangulation of implicit surfaces. *IEEE Computer Graphics and Applications* 2001;21(2):60–9.
- [20] de Araujo BR, Jorge JAP. Curvature dependent polygonization of implicit surfaces, in: *XVII Brazilian symposium on (SIBGRAPI'04) computer graphics and image processing*, 2004. p. 266–73.
- [21] Moller T, Muller K, Kurzion Y, Machiraju R, Yagel R. Design of accurate and smooth filters for function and derivative reconstruction, in: *Proceedings of IEEE symposium on volume visualization*, 1998. p. 143–51.
- [22] Co C, Hamann B, Joy K. Iso-splatting: a point-based alternative to isosurface visualization, in: *11th Pacific conference on computer graphics and applications (PG'03)*, 2003. p. 325–34.