PAPER

Shrink-Wrapped Isosurface from Cross Sectional Images

Young Kyu CHOI[†a)], ***Member and*** **James K. HAHN**[††b)], ***Nonmember***

SUMMARY This paper addresses a new surface reconstruction scheme for approximating the isosurface from a set of tomographic cross sectional images. Differently from the novel *Marching Cubes (MC) algorithm*, our method does not extract the *iso-density surface (isosurface)* directly from the voxel data but calculates the *iso-density point (isopoint)* first. After building a coarse initial mesh approximating the ideal isosurface by the *cell-boundary representation*, it metamorphoses the mesh into the final isosurface by a relaxation scheme, called shrink-wrapping process. Compared with the MC algorithm, our method is robust and does not make any cracks on surface. Furthermore, since it is possible to utilize lots of additional isopoints during the surface reconstruction process by extending the adjacency definition, theoretically the resulting surface can be better in quality than the MC algorithm. According to experiments, it is proved to be very robust and efficient for isosurface reconstruction from cross sectional images.

key words: *surface reconstruction, isosurface, marching cubes, cell-boundary representation, shrink-wrapping algorithm*

1. Introduction

Three-dimensional surface information is very useful in many applications such as medical imaging and topological modeling. In these applications the surface information is often available as a sequence of cross-sectional images. In the medical area, for instance, the computed tomograpy (CT) or the magnetic resonance imaging (MRI) make it possible to obtain cross-sectional images of human body. The problem to be addressed in this paper is that of reconstructing surface representation from tomographic cross sectional images.

Lorensen and Cline [1] have proposed a novel method, called the *marching cubes (MC)* algorithm, which can obtain high resolution isosurface from 3D medical data. Since its original conception, it has been the subject of much further research to improve its quality of surface representation and its performance on large data sets. The quality issue includes handling the ambiguities in surface definition process. After reporting the ambiguity and possible "holes" in the resulting isosurface by Durst [2], successive authors have sought to improve the topological correctness and accuracy[3]–[6]. Nielson and Hanmann [3] pointed out that there is an ambiguity in the face of a cube when all four edges of the face are intersected. They proposed a strategy based on *saddle point value* of the bilinear interpolant to dictate the edge connection on an ambiguous face. Natarajan [4] and Chernyaev [5] independently recognize that there are additional ambiguities in the representation of the trilinear interpolant in the interior of the cube, and they tried to solve this problem by extending the concept of the *body saddle point*. Recently, Lopes et al. [6] tried to create a representation that correctly models the topology of the trilinear interpolant within the cell and that is robust under perturbations of the data and threshold value. To achieve this, they identified a small number of *key points* in the cell interior that are critical to the surface definition. All of the previous efforts tried to modify the MC algorithm by removing the ambiguities, especially attempted to improve the representation of the surface in the interior of each grid cell, but the basic strategy of extracting the isosurface is similar to the MC algorithm.

Recently, Joy et al. proposed a point-based method for isosurface visualization, called *iso-splatting* [7]. Point samples are generated throughout the volumetric domain, and they are projected onto the isosurface of interest. Finally the points are rendered by a surface splatting algorithm. Eventhough it uses point primitives for *representing* and *rendering* the isosurface to enhance the level of *interactivity*, it does not provide any type of surface mesh which is very useful in geometric modeling applications such as volume calculation, manipulation and so on.

In this paper, we propose a quite different approach for extracting the *isosurface model* from a set of tomographic cross sectional images. We introduce the conception of *"isopoint"*, and utilize a relaxation scheme, called *shrink-wrapping process*, to reconstruct the isosurface from an *initial mesh* representing a coarse approximation of the isosurface.

This paper is organized as follows. Previous works and the motivation of our paper are listed in Sect. 2 and we generalize the concept of isopoint in Sect. 3. The initial mesh generation method is introduced in Sect. 4, and in Sect. 5, the shrink-wrapping based surface reconstruction scheme is described. Experimental results are given in Sect. 6 and Sect. 7 concludes this paper.

2. Motivation

The original MC identifies 256 configurations for the cube, depending on whether each of the eight vertices is pos-

Manuscript received February 26, 2007.
Manuscript revised June 13, 2007.

[†]The author is with the School of Information Technology, Korea University of Technology and Education, Rep. of Korea.
[††]The author is with the Department of Computer Science, The George Washington University, USA.
a) E-mail: ykchoi@kut.ac.kr
b) E-mail: hahn@gwu.edu

DOI: 10.1093/ietisy/e90–d.12.2070

itive (*voxelDensity* $\geq$ *densityThreshold*(T_d)) or negative (*voxelDensity* $< T_d$). Any edge with endpoints of opposite sign is intersected by the isosurface and inverse linear interpolation yields an estimate of the intersection point. The set of points can be triangulated to yield an approximation to the isosurface within the cube.

Figure 1 shows the basic cases in the MC algorithm and examples of how the points of intersection between the isosurface and the edge of the cube can be connected. Notice that all of the intersection points, we call them the *isopoints (iso-density points)*, are only on the edges of the cube. The key feature of our approach is the generalization of the conception of the isopoints: allowing isopoints not only on the edges but also on the faces and interior of the cube.

Figure 2 illustrates an example of extracting additional isopoints on the faces and inside the cube ((b) and (c), respectively) for the case 1 of the MC algorithm shown in Fig. 1. In the case of the MC algorithm, the isopoints are restricted on the edge as shown in Fig. 2 (a), but we believe that the isopoints on the face (b) and inside the cube (c) can also be utilized in the isosurface reconstruction. The problem is that it is not easy to find a general methodology for *triangulating the isopoints* such as shown in (d), and thus we adopted a quite different approach. Instead of triangulating directly the isopoints, we adopted the techniques for the problem of *surface reconstruction from unorganized points* (not from the voxel data (an *organized* data)).

A typical solution for this problem was provided by Hoppe et al. [8]. They estimated tangent plane for each point and introduced a *signed distance* function. Finally an isosurface was extracted by a volume-based reconstruction scheme. Kobbelt et al. [9] proposed the mesh generation algorithm based on the shrink-wrapping scheme for the first time, and Jeong et al. extended the concept to produce a mesh model from unorganized 3D points [10]. For a given 3D points, they make a bounding box and linearly subdivide the 6 faces (not the volume) to get an initial cube-shaped mesh. They repeatedly apply the projection (shrinking) and smoothing operations to metamorphose the initial mesh into one similar to the surface of original object. Recently, Koo et al. [11] generalized the initial mesh to overcome the *genus-0 spherical topology restriction* of [10]. We adopted the relaxation methodology into the isosurface reconstruction from tomographical cross sectional images. But in the case of our problem, we can utilize not only the set of isopoints but also the original image data. Thus the initial mesh can be extracted more robustly and accurately than in the previous algorithms for surface extraction from unorganized points by making full use of the cross sectional images.

In the next section, we generalize the concept of isopoint. The initial mesh generation and the shrink-wrapping based surface reconstruction scheme is described in Sects. 4 and 5, respectively.

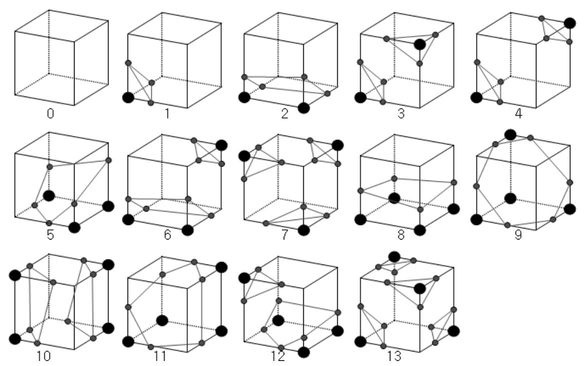

Fig. 1 Basic cases in the MC algorithm and examples of how the isopoints can be connected. Positive vertices are marked.

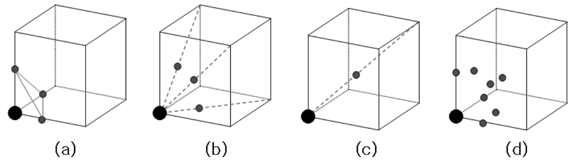

Fig. 2 An example of the case 1 of the MC algorithm. (a) 3 isopoints on the edges, (b) additional isopoints on the face, (c) additional isopoints in the interior the cube, and (d) total isopoints (7 points).

3. Iso-Density Point (Isopoint) Model

In the *cuberille model* of 3D space proposed by Herman [12], a *cuberille* is a dissection of space into equal cubes by three orthogonal set of equally spaced parallel planes, and each cube, a component of the cuberille, is called *voxel*.

If we assume 3D space is dissected into a set of cubic voxels, an instance of the cuberille space V is defined to be a *scene* V, denoted by the tuple (g, V), where $g(v)$ represents the *density* of voxel v. A set of stacked images of a human body obtained by CT or MRI is an example of a scene in the cuberille space.

For a given scene V, a binary scene V_s can be defined by the tuple (g_s, V), where g_s is a *segmentation function* of V. Lots of region segmentation algorithms have been developed especially in the field of medical imaging [13]. In this paper, we only consider the simple *density threshold* function denoted as $g_s(v)$.

In a binary scene, each voxel can be classified into two groups: the *inner (positive)* voxel v^1 and the *outer (negative)* voxel v^0.

$$\begin{aligned} v^0 &\in N_s(V) = \{v \in V | g_s(v) = 0\} \\ v^1 &\in U_s(V) = \{v \in V | g_s(v) = 1\} \end{aligned} \tag{1}$$

Generally, $U_s(V)$ represents the structure of interest (the interior of the object), and $N_s(V)$ indicates the empty null space. The *neighbors* of a voxel v, denoted as $n(s)$, are assumed to be the voxels which are *adjacent* to v. Udupa et al. used the term *O(1)-adjacent* when a pair of voxels share a face [14]. In *O(2)-adjacency*, two voxels sharing an edge are also defined to be adjacent, and the voxel sharing a vertex with v is also included in $n(v)$ in *O(3)-adjacency*. Figure 3 illustrates the neighbors (solid line) of a voxel (dotted line) in O(1)-adjacency (a) and O(3)-adjacency (c), respectively.

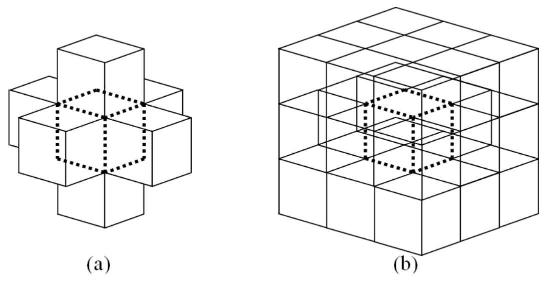

Fig. 3 (a) O(1)-adjacenct neighbor voxels (6 neighbors), (b) O(3)-adjacent neighbor voxels (26 neighbors).

Assume that a scene V is given as the form of cross sectional images. We want to extract all the points (isopoints) supposed to have the same voxel density of T_d. After segmenting V into a binary scene V_s with T_d, they can be acquired between a pair of adjacent voxels by linearly interpolating their densities as follows.

Definition 1: Assume that a pair of *adjacent voxels*, u and v, and a density threshold T_d is given. If u is positive and v is negative, the *iso-density point* (or *isopoint*), denoted as $p_{T_d}(u, v)$, is defined as follows:

$$p_{T_d}(u, v) = p(v) + (p(u) - p(v))\frac{T_d - g(v)}{g(u) - g(v)} \qquad (2)$$

where, $p(v)$ is the 3D coordinates of the center of v.

Consequently, the iso-density point model is defined as follows.

Definition 2: The *iso-density point model* of a scene V under the density threshold T_d, denoted as $p_{iso}(T_d)$, is defined to be the set of all possible isopoints:

$$p_{iso}(T_d) = \{\forall p_{T_d}(v^0, v^1) | v^0 \in n(v^1), \forall v^0, v^1 \in V_s\} \qquad (3)$$

The definition of the *voxel adjacency* described above affects the generation of isopoints. Let's denote $n_1(v)$, $n_2(v)$ and $n_3(v)$ as the neighbor voxels of v in O(1), O(2), and O(3)-adjacency, respectively. In the case of $n_1(v)$, the maximum number of isopoints related to a voxel v is limited to six because $n_1(v)$ has 6 voxels. In the same way, maximum 26 isopoints can be interpolated for a voxel in O(3)-adjacenct neighbor definition.

In the viewpoint of a *cell* as shown in Fig. 2, there are *at most 12 isopoints* in a cell in O(1)-adjaceny since a cell has 12 *edges* (case 13 of Fig. 1). In the case of O(2)-adjacency, maximum *12 additional* points can be extracted because each face of a cell has two diagonals (a cell has 6 faces). The voxels on the end of a *diagonal of a cell* are also defined to be adjacent by extending the voxel adjacency into O(3), and thus no more than 4 additional points can be extracted.

In the MC algorithm, the isopoints are always limited to the O(1)-adjacency, and thus less than 12 isopoints can be participated in the isosurface of a cell. Our method overcomes this restriction by adopting $n_2(v)$ or $n_3(v)$ rather than $n_1(v)$. It can utilize lots of extra isopoints which are very useful in isosurface reconstruction. Consequently, better quality in resulting surface can be expected than the MC algorithm.

4. Initial Mesh by the Cell-Boundary Representation

Basically, our reconstruction scheme follows the *shrink-wrapped boundary face (SWBF)* algorithm [11], which produces a surface mesh from unorganized 3D points cloud. SWBF makes a coarse initial mesh using the boundary faces, and applies an iterative relaxation scheme, called shrink-wrapping process, to approximate the fine surface representing the 3D points cloud. In our case, the isopoints extracted in O(3)-adjacency can be utilized as the target 3D points: *actual 3D points assumed to be sampled from the ideal isosurfaces. The problem is how to generate the initial mesh from the original voxel data.* In this section, our scheme of generating the initial mesh from the input cross-sectional images is introduced.

In our observation, a desirable initial mesh should have the following properties:

- Correctness. Although the initial mesh is a coarse representation of the underlying isosurface, the method should be able to approximate the ideal isosurface as correctly as possible. It is an essential property and can minimize the complexity of the following relaxation process.
- Uniqueness. The initial mesh should be always the same for the same input data. In some algorithm such as the PVP method [15] proposed by Yun et al., the selection of the normal estimation algorithm can affect the resulting surface shape, and thus can not guarantees the uniqueness of the resulting mesh.
- Robustness. Any ambiguity is not permitted in defining the initial mesh at any cell configuration. For example, the case 10 of the MC algorithm shown in Fig. 1 yields an ambiguity of triangulating the isopoints, and thus it is not easy to avoid some holes on surface. Thus, a desirable initial mesh should be robust in any cases of cell configuration.
- Simplicity. The simplicity is needed to reduce the complexity of the resulting surface. Generally the MC algorithm produces highly detailed surface, and consequently generates so many triangular isosurface patches. A desirable initial mesh should be simple in surface complexity than the MC algorithm.

From these considerations, the *cell-boundary representation (CBR)* [16] proposed by Lee et al. was adopted to generate the initial mesh from the cross sectional images. The CBR does not give an accurate surface like MC, but is a robust method for approximating surface from voxel data in that it provides a unique representation without having any ambiguity in surface definition with simplified surface complexity than the MC algorithm.

Let V be a scene in a cuberille of 3D space. The cell

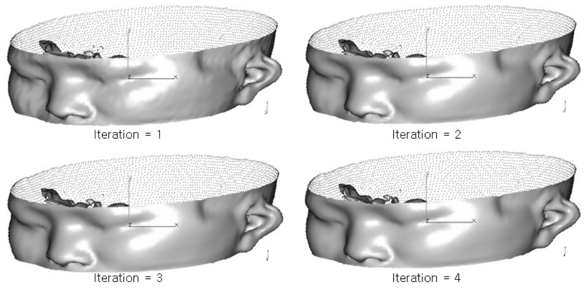

Fig. 4 11 volumetric group of the modeling primitives in CBR [16].

space C is an infinite collection of closed cubes, where the eight vertices of each cube correspond to eight adjacent voxels, called *supporting voxels*, in the cuberille V. Each component of the cell space, a cube as shown in Figs. 1 and 2, is a *cell* and denoted by c. The cell space can be interpreted as a *transformed space* or *shifted space* of the cuberille space. A *boundary cell* in CBR is defined by a cell which contains at least one v^0 and at least one v^1 as the supporting voxels. The all cases in Fig. 1 except the case 0 are typical examples of the boundary cell. The set of boundary cells can be denoted as

$$BC = \{c \mid c \in C, g_s(v) = 1, g_s(u) = 0 \\ \text{for some } v, u \in n_s(c)\} \tag{4}$$

where $n_s(c)$ represents the eight supporting voxels of c.

Definition 3: A cell-boundary of a scene V in a cuberille of 3D space is a tuple (BC, Δ), where BC is the set of boundary cells and Δ, called the *encoding function*, assigns the configuration of eight supporting voxels to each component of BC. The voxel configuration of a cell c, denoted as $\delta(c) \in \Delta$, is an 8-bit configuration code which represent whether each of the eight supporting voxels of the cell is positive (inner voxel) or not.

With the notion of the cell-boundary representation, they provided 19 modeling primitives (MP) and the relation between the resulting surface and the modeling primitive as the surface patch table. Figure 4 illustrates the 11 volumetric groups of the modeling primitives, and Fig. 5 describes their algorithm to extract the surface mesh from the cell-boundary representation.

The cell-boundary representation could not produce highly detailed isosurface since it does not include any subvoxel interpolation as in the MC algorithm. But it guarantees the uniqueness of the mesh for the same input data. It is also very robust since there is not any ambiguity in surface definition and does not make any cracks on surface. Furthermore, it is very efficient in time and space since it only uses pre-defined look-up table with simple scan line algorithm, and reduces the number of surface patches by 40 ~ 50% than in the MC algorithm. Thus it is a perfect scheme for generating the initial surface mesh, and we adopted this method to acquire the coarse initial representation of the isosurface from the input cross-sectional images.

```
Procedure GenCBoundary
Input: 2 slices of scenes at adjacent level with a proper segmentation function $g_s$.
Output: A cell-boundary $C = (BC, \Delta)$.
Begin
[1] Generate binary scenes for each scenes using $g_s$.
[2] Scan two slices of binary scenes and create a cube from supporting voxels: 4 neighbors on one slice and 4 neighbors on the next slice.
[3] Calculate the number of 1-voxel $n_1$ in the cube.
[4] If $2 < n_1 < 8$ then
  [4.1] Compute the 1-voxel configuration $\Delta$ for the cell.
  [4.2] Insert the cell into $C$.
[5] If scanning is not over, goto 2.
[6] Return the cell-boundary $C$.
End
```

Fig. 5 The surface generation algorithm [16].

5. Isosurface Reconstruction by the Shrink-Wrapping Process

Assume that an isopoint model $p_{iso}(T_d)$ and the initial mesh M^I are given. M^I is a crude approximation of the isosurface of the input cross-sectional images, and $p_{iso}(T_d)$ can be regarded as the isopoints sampled from the ideal isosurface with voxel density T_d. In our method, M^I is iteratively metamorphosed into $p_{iso}(T_d)$ by the *shrinking* and *smoothing* operations. For these processes, we adopted the same procedures used in SWBF [11].

5.1 Shrinking

A shrinking step is applying an attracting force to each vertex, that is, a vector between M^I and $p_{iso}(T_d)$. For a vertex q_i of M^I, there is a boundary cell c_b containing it. We search for the nearest isopoint p_i in $p_{iso}(T_d)$ minimizing the Euclidean distance between q_i and p_i. After finding the nearest isopoint p_i, the attracting force vector $f_{p_i} = q_i - p_i$ pushes the mesh vertex q_i toward p_i as follows.

$$q_i \leftarrow q_i + \alpha f_{p_i} \tag{5}$$

The weight (0.0 to 1.0) controls the amount of the attracting force. Since there is a possibility of sharing the same point by more than two mesh vertices, the full attraction force (= 1) may cause a non-manifold region in the surface. To avoid this, we provided a weight of less than 1.0 and experimentally chose to be 0.5.

Unlike in the method of [10] requiring a global search for finding the nearest point, our method can be done in a local manner to obtain the same result. Since the nearest isopoint from q_i should always be inside of c_b and its O(3)-adjacent neighbors, it is sufficient to search the 27 cells for the optimal p_i. It greatly reduces the processing time for the shrinking process compared with [10]. Furthermore, our method overcomes the *surface duplication problem* of SWBF. If the *unorganized points* are acquired only from the surface (not the volume) of an object, it is impossible to determine whether *a non-boundary cell* is inside of the object or not, and consequently, the surface mesh by SWBF may contain two layers of surface of opposite orientation. In our method, we can avoid this problem because the input volumetric data makes it possible to specify a cell is inside or not.

The processing time of the shrinking operator is proportional to the number of vertices of the initial mesh M^I. As described above, it is sufficient to consider the isopoints inside only 27 cells to find the nearest point p_i from a mesh vertex: for a mesh vertex q_i, the distance calculation from q_i to the points can be done only for those isopoints embraced in the cell c_b containing q_i and 26 O(3)-adjacent neighbors of c_b. In the viewpoint of a cell, the maximum number of possible isopoints inside the cell is 28: 12 in the edges of a cell (O(1)-adjacent), 12 in the diagonals of the six faces (O(2)-adjacent), and 4 in the diagonals of the cell (O(3)-adjacent). Thus, the number of candidate isopoints in $p_{iso}(T_d)$ for finding p_i can not exceed 756 (= 27×28). The calculation of the attracting force can be done in a constant time C. Consequently, the time complexity for the shrinking step becomes $O((756 + C)n) = O(n)$, where n is the total number of the vertices in M^I.

5.2 Surface Smoothing

The smoothing step tries to relax the shrink-wrapped surface to achieve a uniform vertex sampling. We have adopted the same method used in SWBF, which is employing the approximation of Laplacian L. This is the average vector of *1-neighbor* edge vectors of a given vertex, and thus a surface shrinkage effect may occur. Thus the tangential component L_t of L, which is perpendicular to the vertex normal $\vec{n}$, was utilized as follows.

$$L(q_i) = \frac{1}{m} \sum_{j \in 1-nbr(i)} (q_j - q_i) \tag{6}$$

$$L_t(q_i) = L(q_i) - (L(q_i) \cdot \vec{n})\vec{n} \tag{7}$$

$$q_i \leftarrow q_i + \lambda L_t(q_i) \tag{8}$$

The weight λ determines the amount of the smoothing operator. If we apply a big λ value, we can get uniformly sampled mesh, but our shrink wrapping procedure can fail to capture big convex or concave region because mesh can be shrink even though we apply tangential motion only. In the case of a small λ, it is not easy to get uniformly distributed mesh. We chose 0.4 as the proper λ value by experiments.

The processing time of the smoothing step is also proportional to the number of vertices in the initial mesh M^I since for a mesh vertex q_i there are at most six neighboring vertices, $O(6n) = O(n)$.

6. Experiments

Our algorithm was implemented in C and C++ under Windows XP. For the experiment, we used the *human head*, the *plastic skull*, and the *head aneuyrism* data. The human head consists of 30 slices and each slice is 128×128. We assume that the CT images are stacked along z-direction and x, y-direction is defined within each image plane. Figure 6 shows 10 slices of the input data. Using the density thresholding (T_d = 128), we segmented the input data for extracting the isosurface.

Figure 7 illustrates the isopoints extracted in O(1) and O(3)-adjacency, respectively, and we used the O(3)-adjacency in our method. In the case of the MC algorithm, only 26,788 points are utilized in surface reconstruction as shown in (a). By adopting the O(3)-adjacency, our method is able to make use of additional isopoints from the faces and the diagonals of each boundary cell. The total number of the isopoints (132,260) is much larger than that of the MC algorithm (26,788), and theoretically better quality in resulting surface can be expected.

Figure 8 shows the initial mesh produced by the cell-boundary representation. After applying 4 iterations of shrink-wrapping process as shown in Fig. 9, the crude initial mesh was metamorphosed into the smooth surface representing the isosurfaces as shown Fig. 10. Compared with the surface by the MC algorithm as shown in Fig. 11, our method produces more smooth surface (less jagged sur-

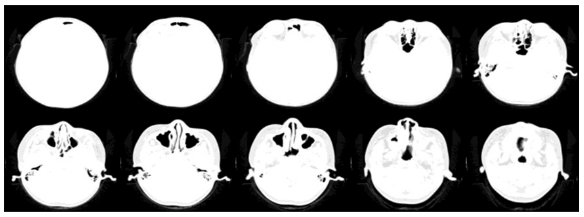
Fig. 6 The human head data. (10 slices of overall 30 sections)

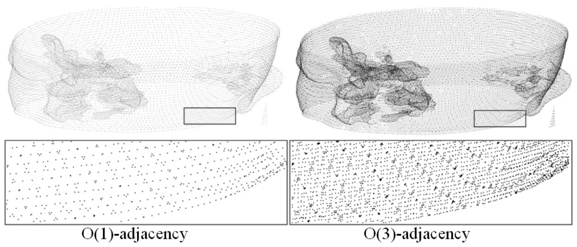

Fig. 7 The isopoint model (T_d = 128): 26,788 isopoints in O(1)-adjacency (left) and 132,260 points ispoints in O(3)-adjacency (right).

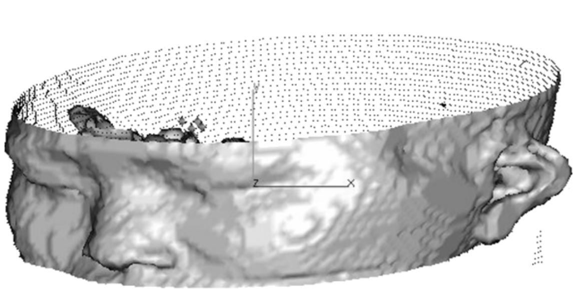
Fig. 8 The initial mesh by CBR. (T_d = 128)

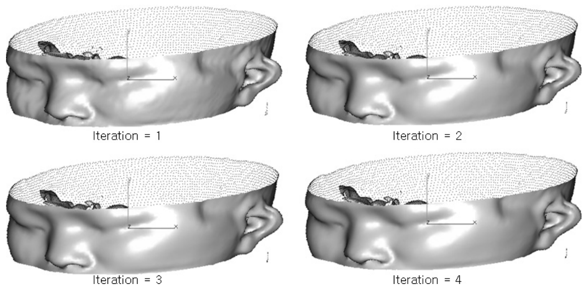

Fig. 9 Shrink-wrapping process. (4 iterations)

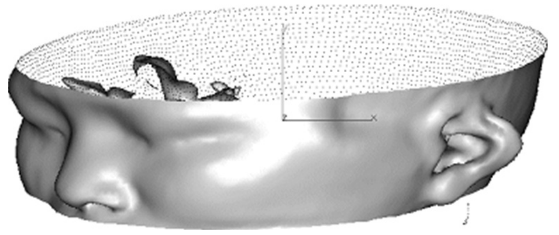

Fig. 10 Isosurface reconstructed by our method (O(3)-adj).

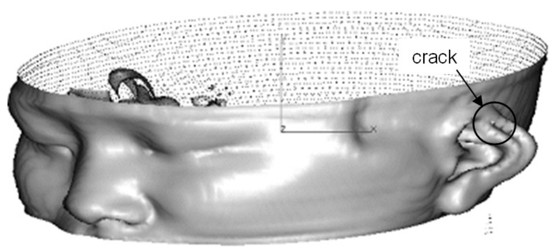

Fig. 11 Surface reconstructed by MC algorithm with some cracks and jagged surface. (T_d = 128)

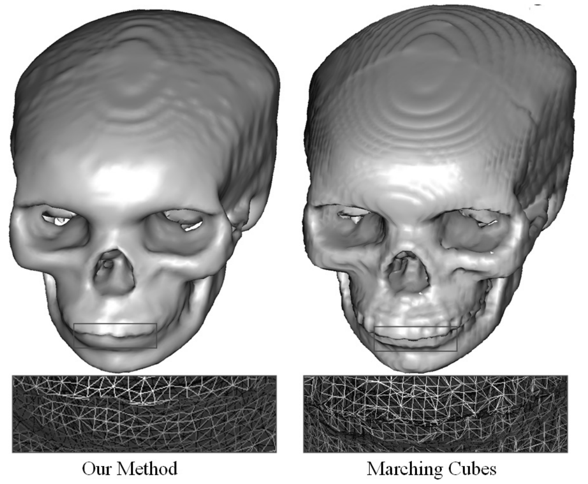

Fig. 12 Skull data. (128 × 128 × 60 slices, T_d = 80)

faces) without having any cracks on surface. The complexity of the surface was also simplified about 42%.

Figure 12 shows experimental result for the plastic skull data (60 slices of 128 × 128 × 8 bit image). Although it does not contain any cracks on surface, the mesh around the teeth seems to be not satisfactory compared with the MC algorithm. Since our initial meshing algorithm does not provide highly detailed surface patches (less than 41% compared with MC), it can miss some regions of high curvature. The surface subdivision scheme [10], [17] could be appended to our method to enhance the mesh quality in high curvature regions.

Figure 13 illustrates comparisons of our method with MC for the head aneurysm data (256×256×256 slices). It is rotational angiography scans of a head with an aneurysm obtained from http://www.volvis.org. In this data, our method produces only 53% of triangles with similar mesh quality of blood vessels compared with MC algorithm.

Table 1 provides the surface reconstruction summary for all data sets used in our experiments. Table 2 summa-

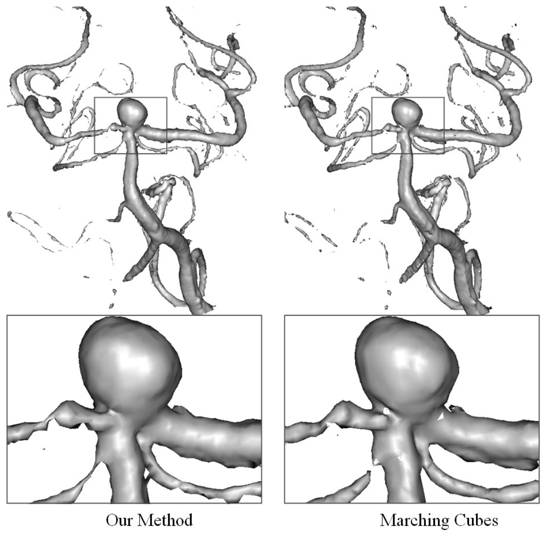

Fig. 13 Head aneurysm data. (256 × 256 × 256 slices, T_d = 60)

Table 1 Comparison with the MC algorithm.

		Our method	Marching Cubes
	Adjacency	O(3)	O(1)
Head	Number of isopoints	132,260	26,788
	Number of vertices	16,345	26,788
	Number of triangles	32,259	55,536
Skull	Number of isopoints	248,165	49,201
	Number of vertices	28,841	49,201
	Number of triangles	57,843	98,072
Head aneurysm data	Number of isopoints	193,954	34,058
	Number of vertices	17,661	34,058
	Number of triangles	34,985	66,864

Table 2 Comparison of execution time for the head data (msec).

	Adjacency	O(1)	O(2)	O(3)
Our Method	Initial Mesh by CBM	78	78	78
	Isopoint Extraction	37	62	94
	Shrinking Operator	47	91	109
	Smoothing Operator	16	16	16
	Overall (4 iterations)	367	568	672
Marching Cubes	Overall	121	-	-

rizes the actual computation times for the head data. Since we adopt a relaxation scheme, the shrink-wrapping process takes most of the computation time. The number of the isopoints also affects the computation cost. According to our implementation, the overall computation time of our method in O(3)-adjacency is less than a second (about 6 times of the MC algorithm) in the case of applying 4 iterations of shrink-wrapping process for human head data.

7. Conclusion

Our method surmounts the O(1)-adjacency limitation of the MC algorithm by generalizing the concept of isopoints, and it can exploit more information in surface reconstruction than the MC algorithm. By adopting the cell-boundary representation, it also can avoid any ambiguity in defining the initial mesh, and produces the final isosurface without any cracks by iteratively applying the shrink-wrapping process. According to experiments, our method is proved to be very robust and efficient in isosurface reconstruction from tomographic cross sectional images.

We believe that our method opens up some new topics for future work. Our relaxation based isosurfacing technique can be combined with the surface subdivision approaches to produce higher quality surface. It is also valuable to confirm the effectiveness of using O(3)-adjacency in isosurface reconstruction.

In the future, we also intend to apply the O(3)-adjacent isopoints to the view-dependent approaches. The point-based rendering schemes such as the iso-splitting technique can be a good candidate for further study.

We also expect that our method can be successfully applied to the problem of surface reconstruction from various kinds of non-orthogonal volumes data (non-cubic voxels such as parallelepiped or hexagonal prism). Conventional techniques producing the isosurface at a time such as MC looks more difficult to generalize than the relaxation scheme. Our method does not extract the isosurface directly from the voxel data. All componets of our method, generating a crude initial mesh, acquiring O(3) isopoints, and metamorphosing the initial mesh using a relaxation scheme, seem to be easier to extend to a non-cubic voxel data. Thus our method can be a good solution to reconstruct the isosurface from the non-orthogonal volumes.

References

[1] W. Lorensen and H. Cline, "Marching cubes: A high resolution 3-d surface construction algorithm," Comput. Graph., vol.21, no.4, pp.163–169, 1987.

[2] M. Durst, "Letters: Additional reference to marching cubes," Comput. Graph., vol.22, no.2, pp.72–73, 1988.

[3] G. Nielson and B. Hamann, "The asymptotic decider: Resolving the ambiguity in marching cubes," Proc. IEEE Visualization '92, pp.83–91, 1992.

[4] B. Natarajan, "On generating topologically consistent isosurfaces from uniform samples," Visual Computer, vol.11, pp.52–62, 1994.

[5] E. Chernyaev, "Marching cubes 33: Construction of topologically correct isosurfaces," Technical Report CN/95-17, CERN, 1995, http://wwwinfo.cern.ch/asdoc/psdir/mc.ps.gz

[6] A. Lopes and K. Brodlie, "Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing," IEEE Trans. Vis. Comput. Graphics, vol.9, no.1, Jan.-March 2003.

[7] C. Co, B. Hamann, and K. Joy, "Iso-splatting: A point-based alternative to isosurface visualization," Pacific Graphics 2003, pp.325–334, Oct. 2003.

[8] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," Comput. Graph. (SIGGRAPH '92 Proceedings), pp.71–78, July 1992.

[9] L. Kobbelt, J. Vorsatz, U. Labsik, and H. Seidel, "A shrink wrapping approach to remeshing polygonal surfaces," Computer Graphics Forum, Proceedings of Eurographics '99, vol.18, no.3, pp.119–129, Sept. 1999.

[10] W. Jeong and C. Kim, "Direct reconstruction of displaced subdivision surface from unorganized points," Graphical Models, vol.64, no.2, pp.78–93, March 2002.

[11] Y. Choi, B. Koo, and B. Choi, "Shrink-wrapped boundary face (SWBF) algorithm for mesh reconstruction from unorganized 3D points," IEICE Trans. Inf. & Syst., vol.E87-D, no.9, pp.2283–2285, Sept. 2004.

[12] G. Herman and H. Liu, "Three-dimensional display of human organs from computed tomograms," Computer Graphics and Image Processing, vol.9, pp.1–21, 1979.

[13] M. Kim, S. Nam, and H. Hong, "Three dimensional visualization technologies in medical imaging," Communications of the Korea Information Science Society, vol.16, no.12, pp.13–21, Dec. 1998.

[14] J. Udupa, S. Srihari, and G. Herman, "Boundary detection in multidimensions," IEEE Trans. Pattern Anal. Mach. Intell., vol.4, no.1, pp.41–50, 1982.

[15] H. Yun and K. Park, "Surface modeling method by polygonal primitives for visualizing three-dimensional volume data," Visual Computer, vol.8, pp.246–259, 1992.

[16] E. Lee, Y. Choi, and K. Park, "A method of 3D object reconstruction from a series of cross-sectional images," IEICE Trans. Inf. & Syst., vol.E77-D, no.9, pp.996–1004, Sept. 1994.

[17] C. Loop, Smooth subdivision surfaces based on triangles, Master's Thesis, Department of Mathematics, University of Utah, Aug. 1987.

Young Kyu Choi received the degree of B.S. from Kyungpook National University, Daegu, Korea in 1989, the M.S. and Ph.D. in Electronic Engineering from the KAIST, Daejon, Korea in 1991 and 1995, respectively. He worked for LG Industrial systems from 1995 to 1998. He is now an associate professor at the School of Information Technology, Korea University of Technology and Education since 1999. His current interests include volume visualization, virtual reality, medical imaging and computer vision.

James K. Hahn is currently a full professor and the chairman of the Department of Computer Science at the George Washington University where he has been a faculty since 1989. He is the founding director of the Institute for Biomedical Engineering and the Institute for Computer Graphics. His areas of interests are: medical simulation, image guided surgery, medical informatics, visualization, and motion control. He received his Ph.D. in Computer and Information Science from the Ohio State University in 1989, an MS in Physics from the University of California, Los Angeles in 1981, and a BS in Physics and Mathematics from the University of South Carolina in 1979.