An Octree Isosurface Codification Based On Discrete Planes

Imma Boada
Institut Informàtica i Aplicacions
Universitat de Girona
Girona, Spain
imma@ima.udg.es

Isabel Navazo
Dept. Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Barcelona, Spain
isabel@lsi.upc.es

Abstract

The purpose of this work is to describe a method to code a decimated model of an isosurface on an octree representation while maintaining volume data if it is needed. The proposed technique is based on grouping the Marching Cubes (MC) patterns in five configurations according the topology and the number of planes of the surface contained into the cell. Moreover, a discrete number of planes, where the surface lain on, is fixed. Starting from a complete volume octree with the isosurface codified at terminal nodes according to the new configurations, a bottom-up strategy is taken in merging cells. Such strategy allows to implicitly represent coplanar faces in upper octree levels without introducing error. At the end of this merging process, when is required, a reconstruction strategy is applied to generate the surface contained in the octree intersected leaves. Some examples with medical data demonstrate that reduction up to 50 per cent on the the number of polygons could be achieved.

1. Introduction

Multiresolution has become one of the most promising optimization methods for resolving the conflict among the increasing size of volume data sets and the capabilities of available hardware. Encoding in a single structure a large number of different representations, multiresolution schemes allow the extraction of variable resolution representations according to the user or the viewing parameters specifications. Many different approaches have been proposed for the multiresolution management of surfaces (see [6] for a survey). Multiresolution volume data management has been also extendely analised, with methods based on classical wavelets [26, 14], on multidimensional trees [29], or more sophisticated methods that take advantage of texture hardware capabilities [12, 2, 30]. However, despite the improvements achieved in these areas the idea of combining on a single structure surface and volume data is still in a developing stage.

The method we are going to present, is *the first step* on the definition of an hybrid framework able to maintain multiresolution surface and volume data renderings. The model allows to codify a surface on an octree data structure that, different of previous octree based methods, maintains the surface-volume data connection.

Based on the classification of original Marching Cubes patterns in five classes, characterized by the number and the position of the planes that define the surface contained into an intersected cell, the surface is initially represented at maximal division nodes of an octree. Then, a merging process is applied in a bottom-up traversal and sibling nodes, which can be represented in the parent node, in one of the five possible configurations, without introducing error, are purged. At the end of this process, the surface is codified in a set of nodes distributed at different levels of the octree. To generate the surface, a reconstruction technique is applied at each terminal intersected node, achiving, in the best situation, simplifications of almost 50% on the number of polygons.

The main advantages of this codification are that not only reduces surface output fragmentation it also provides a framework able to support multiresolution renderings, if an error is introduced in the merging process. Moreover, for each isosurface terminal node its volume information is maintained on its lower octree nodes.

The paper is organized as follows. After a brief description in Section 2 of previous work, the proposed surface codification technique is introduced in Section 3. A complete description of the algorithm proposed to codify the surface in the octree is given in Section 4. In Section 5, the reconstruction algorithm applied to generate the surface is described. Several results achieved with the proposed technique are shown in Section 6. Finally, concluding remarks and future work are presented in Section 7.

0-7695-1215-1/01 $10.00 © 2001 IEEE

2. Previous Work

The Marching Cubes (MC) has become the best known of the surface-based rendering techniques [11]. Since its publication, in 1987, to overcome its main disadvantages several improvements have been proposed. Algorithm's efficiency has been enhanced with strategies that avoid processing all volume data cells [3, 4, 13, 24, 28]. Several techniques to solve ambiguities have been proposed [7, 25, 16]. And a lot of strategies to reduce the large amount of generated polygons defining surfaces have been also presented [8, 9, 10, 22].

Despite the hight quality renderings generated with the method, interactive renderings are difficult to be achieved. A common solution to overcome this problem is the combination of the MC with surface simplification strategies. In the literature, a variety of surface simplification techniques have been proposed. These techniques can be classified into two categories.

The first category, is characterized by the application of a simplification process previously to the surface generation phase. In general, strategies in this group exploit spatial coherence defining coarser representations of the original volume data, just averaging or subsampling neighboring vertices of a given grid into the vertices of a new coarser grid. As the new grid has fewer cubes, the number of faces defining the surface decreases. However, the new surface may show differences of the surface generated from the original data. Such approaches, combined with adaptive isosurface extraction techniques, are an efficient tool to achieve interactive exploration of volume models. Nevertheless, the surface reconstruction process becomes more complex, due to the possible cracks that could appear on the generated surface [18, 23, 20, 27].

The second group, is characterized by the application of the simplification process on the reconstructed surface, commonly represented as a triangular mesh. Most of the techniques in this group, simplify triangular meshes, either by merging elements or by re-sampling vertices using different error criteria to measure the fitness of the approximated surfaces [22, 10, 6].

The surface codification we propose, in next section, exploits the simplifications introduced in the Discretized Marching Cubes (DiscMC). The DiscMC [15, 17], is a derivation of the MC, that replaces the edge interpolation, applied for the vertex polygon computation in the standard MC, by a midpoint selection. This simplification reduces the set of vertices that can be generated and implies that the generated facets, that define the surface, can lie only on a finite number of distinct plane incidences. The ability to merge co-planar elements of this method achieves a high simplification rate. In our approach this merging process is based on the proposed configurations and the geometrical codification of the MC patterns on the octree.

3. The Octree Surface Codification

The hierarchical structure of octrees has lead this representation scheme a widespread used framework to maintain multiresolution on regular data sets. In the classical octree representation [21] nodes are labeled as black, white or grey. *Black* nodes correspond to regions completely inside the object. *White* nodes correspond to regions completely outside the object. White and black nodes are leaves, i.e. they are no further subdivided. Nodes containing part of the surface are labeled as *grey* and are recursively subdivided. Leaf grey nodes are called terminal grey (TG).

In the recent years, several extensions of the octree model have been proposed. One of them is the Extended Octree (EO) model [19]. The EO incorporates new terminal nodes that contain part of the surface object, yielding exact representations for polyhedra and reducing the required storage. In particular, the EO introduces as a terminal nodes: *Face nodes* (crossed by a single planar face of the solid), *Edge nodes* (contain part of two neighboring faces and a part of their common edge), *Vertex nodes* (contain a vertex of the polyhedron and a part of the faces and edges converging to it) and *Nearly Vertex nodes* (contain two or more faces that converge to a single vertex outside the node).

Driven by the simplification introduced in the DiscMC and by the original MC patterns, we propose a variation of the EO to codify the surface. The specific terminal surface nodes (TS) of the proposed octree codification are described in next section.

3.1. Characterization of Terminal Surface Nodes

The selection of the terminal surface nodes has been done according the number and the topology of the planes that define the surface contained in an intersected cell (see figure 2).

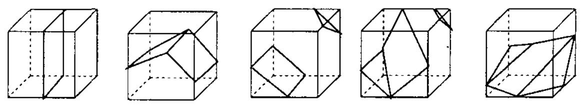

Figure 1. The surface contained in a cell can codified as a Face, an Edge,a Band^n, a Band* or a DoubleEdge.

Let consider $P^* = \{p_0, p_1, \ldots, p_n\}$ the set of polygons vertices obtained from the application of the DiscMC to an intersected cell. According the position of these polygons this cell can be classified as:

- a **Face**, denoted as F, if exists a π plane such that, all $p_i \in P^*$ vertices lye on it. Patterns 1,2,8 and 9 of the original MC are classified as F cells (see figure 2).
- an **Edge**, denoted as E, if exist two planes π_0 and π_1 such that, given a point $p_i \in P^*$ one of the following situations is given: (a) p_i lies on the π_0 plane; or (b) p_i lies on the π_1 plane, or (c) p_i lies on π_0 and π_1 (this situation appears when the vertex lies on the π_0, π_1 intersection). Pattern 5 of the MC configurations corresponds to an Edge cell (see figure 2).
- a **Double Edge**, denoted as DE, if exist three planes π_0, π_1 and π_2 such that, given a point $p_i \in P^*$ one of the following situations is satisfied: (a) p_i lies on the π_0 plane, or (b) p_i lies on the π_1 plane, or (c) p_i lies on the π_2 plane, or (d) p_i lies on π_0 and π_1 (this situation appears when the vertex lies on the π_0, π_1 intersection), or (e) p_i lies on π_1 and π_2 (this situation appears when the vertex lies on the π_1, π_2 intersection). Patterns 11 and 14 of figure 2 are examples of DoubleEdge cells.
- a **Band** , denoted as B^n, if exist n plane equations, that never intersect into the cell and given a point $p_i \in P^*$ always lies in one and only one of these planes. The number of planes, n, contained in a Band^n could be 2,3 or 4. Patterns 3, 4, 6, 7, 10 and 13 of figure 2 correspond to a Band cell configuration.
- a **Special Band**, denoted as B^*, if it can be represented as the union of a Face cell and an Edge cell. In this case, three plane equations are required to represent the surface contained in the cell. Pattern 12 of figure 2 is an example of a Special Band.

The correspondence between the MC patterns and the five configurations is represented in Table 1. The non intersected node is denoted as **Null**.

Based on this classification we define as terminal surface nodes those that can be codified as a Face, an Edge, a DoubleEdge, a Band or a SpecialBand (see figure 1).

3.2. Terminal Surface Nodes Codification

The surface of a terminal surface node n_i is codified as

$$s(n_i) = \{K, < V_0, d_0 >, < V_1, d_1 >, \ldots, < V_n, d_n >\}$$

where K represents the class at which the node belongs to (i.e. F, E, DE, B^n or B^*) and ($< V_0, d_0 >, < V_1, d_1 > , \ldots, < V_n, d_n >$) represents the set of plane equations that define the surface. Each couple $< V_i, d_i >$ corresponds to a π_i plane where $V_i = (A_i, B_i, C_i)$ is the normal vector and d_i is the plane distance to the node origin. Note that, applying the simplification introduced by the DiscMC, the number of plane incidences is limited, and thus, each V_i could be represented as an integer pointer to a pre-computed look-up-table that stores all possible plane incidences. This integer pointer is also used to implicitly maintain the information of the inner and outer side of the object. The codification of the surface is reduced to a set of integers.

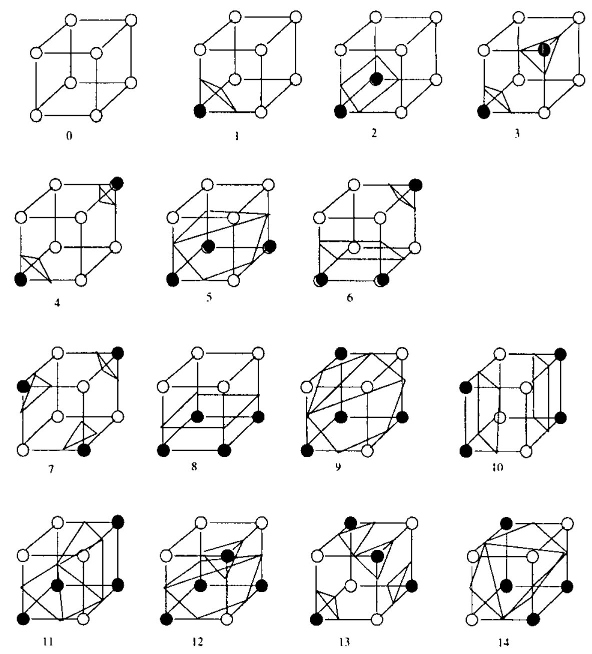

Figure 2. The patterns of the original MC.

Class Configuration	MC pattern
Face	1,2,8,9
Edge	5
DoubleEdge	11,14
Band	3,4,6,7,10, 13
SpecialBand	12
Null	0

Table 1. The correspondence between MC patterns and the five possible configuration.

4. The Octree Construction Algorithm

The key point of the surface codification algorithm is the application of a simplification strategy that reduces the number of polygons defining the surface. This simplification is based on the detection and merging of coplanar polygons.

The algorithm starts with the surface (initially obtained with the DiscMC), codified at maximal subdivision inter-

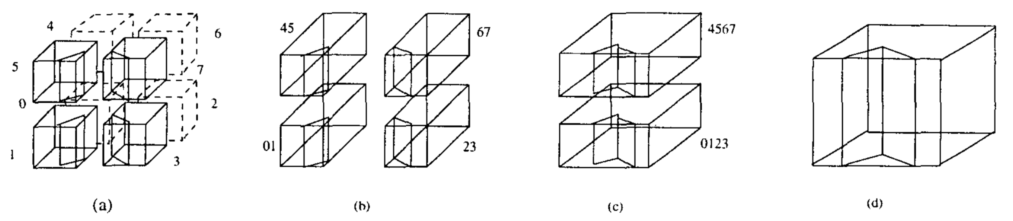

Figure 3. The reduction of a set of Face nodes to an Edge. The merging process starts grouping these descendents by pairs $(n_0, n_1), (n_2, n_3), (n_4, n_5)$ and (n_6, n_7). At this first step the Plane Position test hasn't to be applied, the resultant configuration can be obtained from the MT table directly (see Table 2, where $(N, F) \to F$). The resultant configurations $\{F, F, F, F\}$ are grouped again and evaluated. In this case, the Plane Position test has to be applied to select if the pair (F, F) generates a Face, an Edge or a Band^n. As planes intersected into the node, the generated codifications are Edges. Finally, (n_{0123}, n_{4567}) is evaluated. In this case, the (E, E) configurations generate an Edge.

sected nodes according to its configuration. On a bottom-up traversal of this octree, coplanar polygons are detected and merged taking into account the proposed terminal surface configurations. Starting at level $n - 1$ (i.e. parents of maximal subdivion nodes) the eight descendants of each intersected node n_i are evaluated:

- if child nodes can be codified in the parent node, n_i, as a Face, an Edge, a DoubleEdge, a Band^n or a Band*, descendants information is represented in n_i, and these child nodes are pruned. To represent the surface in the parent node, the planes equations defining the surface have to be updated.
- conversely, if this codification is not possible the process stops and no more ancestors of the node are processed. In this case, the child nodes correspond to a TS node.

Summarizing, if we define a *codificable node* as a node that corresponds to one of the five defined TSN configurations, the simplification of child nodes in the parent, is possible if they can be reduced to a codificable node. Note that the merging process always starts with a set of eight *codificable* nodes and the simplification is possible if they can be reduced to a codificable node. All possible configurations can be pre-computed and stored in a table to facilitate the application of this process. This look-up-table is described in next section.

4.1. The Merging Table

The *Merging Table* (MT) determines when the merging is possible and how has to be done. To generate the MT we have considered all possible combinations of two codificable nodes and for each pair of configurations the set of resultant codificable configurations that can be generated (see Table 2).

Given a pair of codificable nodes the resultant codification not always can be obtained directly from the table. See for example the (F, F) pair. In this case, three possible configurations can be obtained a F, an E or a B^n. To select which of these possible codifications correspond to the analyzed pair, a `Plane Position test` has to be applied. If $\pi_i : A_i x + B_i y + C_i z = d_i$ and $\pi'_i : A'_i x + B'_i y + C'_i z = d'_i$, are the plane equations defining the surface contained in F nodes, respectively, the resultant codification after the merging is: (i) F, if the planes are *Coplanar*; (ii) B^2 if the planes are *Parallel*; (iii) E if the planes are *Incident* and its intersection falls within the parent node; (iv) B^2 if the planes are *Incident* and the intersection does not fall within the parent node. Note that, as all the plane equations are codified as pointers to a table of planes, the `Plane Position` test is reduced to a set of integer comparisons.

In figure 3, we have illustrated the application of the merging process to a node n_i, where $\{n_0, n_1, n_2, .., n_7\}$ and $\{N, F, N, F, N, F, N, F\}$ represent its codificable child nodes and their configurations.

5. Surface Reconstruction Algorithm

Once the octree has been codified if an explicit representation of the surface is needed, for rendering purposes for example, a surface reconstruction algorithm has to be applied.This algorithm could be decomposed in two steps. The first, traverses in a top-down manner the octree and selects all terminal intersected nodes. The set of selected nodes is represented as S.

The second step applies a reconstruction process to each node $n_i \in S$. Before the description of this process some

	N	**F**	**E**	**DE**	**B^n**	**B***
N	N	F	E	DE	B^n	B^*
F	F	F E B^n	E DE B^*	DE	B^n B^*	B^*
E	E	E DE B^*	E DE	NC	B^*	NC
DE	DE	DE	NC	DE	NC	NC
B^n	B^n	B^n B^*	B^*	NC	B^n	NC
B*	B^*	B^*	NC	NC	NC	NC

Table 2. Given a pair of nodes the MT Table indicates the set of codificable configurations that can be generated. NC represents a no codificable configuration.

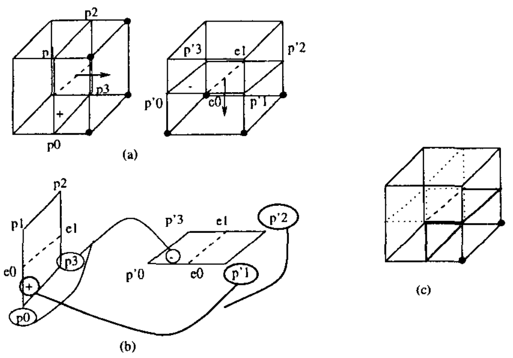

Figure 4. The surface information of the node is $s(n_i) = (E, \pi_0, \pi_1)$. (a) In a first step the intersection of each plane with the node is computed obtaining: $P_0 = \{p_0 . p_1, p_2, p_3\}$ and $P_1 = \{p'_0 . p'_1, p'_2, p'_3\}$. The edge vertices (e_0, e_1) are identified. (b) Point selection is done according the plane signs: $\pi > 0 \rightarrow p'_0, p'_3$ are eliminated ;$\pi' < 0 \rightarrow p_1, p_2$ are eliminated. (c) $\{p_0, p_3, e_1, e_0\}$ and $\{p'_1, p'_2, e_1, e_0\}$ polygons define the surface.

considerations have to be taken into account: (i) each plane equation represented in $s(n_i)$ has to be represented by one polygon; (ii) this polygon is defined from the connection of *all* or *some* of the intersection points between the plane and the edges of the node. (iii) the number of polygons required to define the surface varies with the configuration of the node. In particular, the configuration-polygon relation is F:1, E:2, DE:3, B^n:n and B*:3.

To define the surface of an intersected node n_i we apply the three following steps:

(1) *Intersection Point Computation.* On a first step $s(n_i)$ is evaluated and for each plane equation $< V_i, d_i > \in s(n_i)$ (see Subsection 3.2), the intersection points between the plane and the edges of the node are computed. The limited number of plane incidences allows the definition of a look-up-table in which the intersection points generated for each plane are stored. Thus, the set of intersection points P_i of each plane in $s(n_i)$ will be obtained directly from this table just applying a translation respect the origin of the node. When the node corresponds to an Edge, a DoubleEdge or a *Band** the points defining the generated edges are also computed. (see figure 4(a))

(2) *Point Selection.* Once the intersection points are known, a selection process is applied to determine which of these points are members of the surface. This selection depends of the node's configuration and it is done evaluating the normal vector direction. Selected vertices are represented in P_i^* (see figure 4(b)).

(3) *Points Connection.* Finally, P_i^* vertices are connected. To proper counter clock-wise ordering these points we follow the order established for the intersection point computation. To guarantee the correct connection possible configurations of 4,5 and 6 intersection points have been evaluated and stored in a look-up-table (see figure 4(c)).

The error involved in all the surface extraction process is the error introduced by the DiscMC, i.e. an error of 1/2 of the cell edge size.

This reconstruction algorithm can be extended to allow multiresolution renderings. The first step, is substituted by a selection process that selects nodes according to the distance to a focus of interest. The reconstruction step is also modified to reduce the number of polygons that define the surface of non-interesting nodes [1].

6. Experimental Results

Four data models have been used for the tests: a CT-vertebra of 128x128x80; a CT-head of 96x96x69; a CT-scanned jaw of 128x128x40 and a sphere of 64^3. The method has been implemented on a Pentium III at 450MHz.

In our tests we have evaluated the degree of simplification that can be achieved with respect to standard MC and the quality of the renderings. In table 3 the comparison of the performance of the proposed simplification algorithm and the original MC in terms of: number of polygons, number of intersection points defining the surface and surface

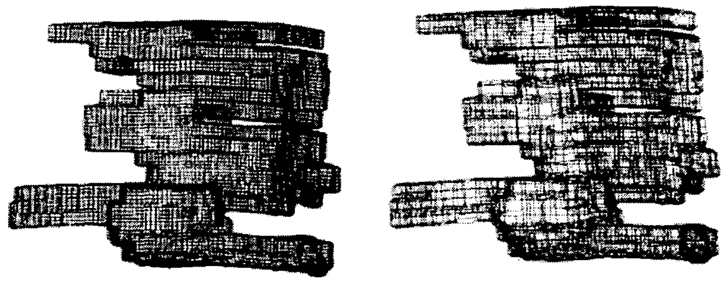

Figure 5. The CT-vertebra wire-mesh model before and after the simplification process.

extraction time, are presented. Best results are obtained on the CT-vertebra (see figure 8). In this case the reduction on the number of polygons is of a 50%. Despite the simplification on the number of polygons (see figure 5) the quality of the renderings is comparable with the MC renderings. The simplification on the CT-head only achieves a reduction of 15 % (see figure 7). While in the CT-jaw the reduction is approximately of a 38% (see figure 6). The sphere achieves a reduction of a 35%.

The execution time is reasonable longer than that of the MC, however, the reduction on the number of polygons will reduce the complexity of posterior operations.

We are conscious of the similarity with the DiscMC algorithm and that the degree of simplification of our method is lower than the one obtained with the DiscMC. Despite this similarity, note that in the DiscMC the face merging is performed on all adjacent faces which lie on the same plane while we are constrained to the eight descendants of an octree node. On the other hand, note that in [17] to give in output a triangle-cased representation, it needs to triangulate these merged polygons; but these polygons might be non convex, or can contain holes. Therefore, it adopts an efficient solution for triangulating them on the fly (and thus times are fast, in most cases slightly better than standard MC). A disadvantage of this approach is that it usually produces in output very thin triangles, which can give problems both in rendering and in further geometrical processing of the mesh. Our approach, in contrast is based on a hierarchical approach so, by definition, the merged region should be much more regular (even if in some cases of lower extension than the DiscMC); therefore also the triangulated representation should posses a more regular shape.

Moreover, we have to keep in mind that our main objective is twofold: to reduce the number of polygons defining the surface and to maintain the connection between surface and volume data information.

Data set	**num. polygons**	**num. inter. points**	**time**
CT-vert. (MC)	13.597	52.755	3,59
CT-vert.*	7.501	28.637	3,8
CT-head (MC)	33.480	120.629	6,8
CT-head*	28.990	104.033	8,4
CT-jaw (MC)	38.491	144.470	8,9
CT-jaw*	23.656	86.357	9,83
sphere (MC)	24.464	78.936	2,7
sphere*	15.720	47.732	2,9

Table 3. Experimental results. All times are in seconds

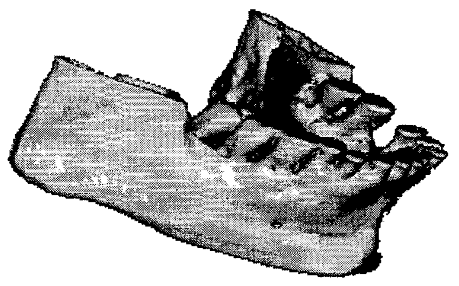

Figure 6. a rendering of the CT-jaw.

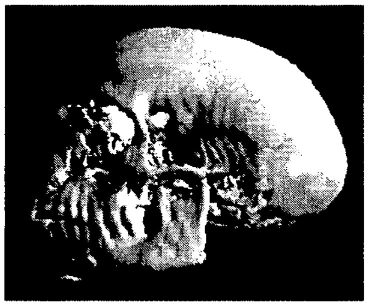

Figure 7. A rendering of the CT-head.

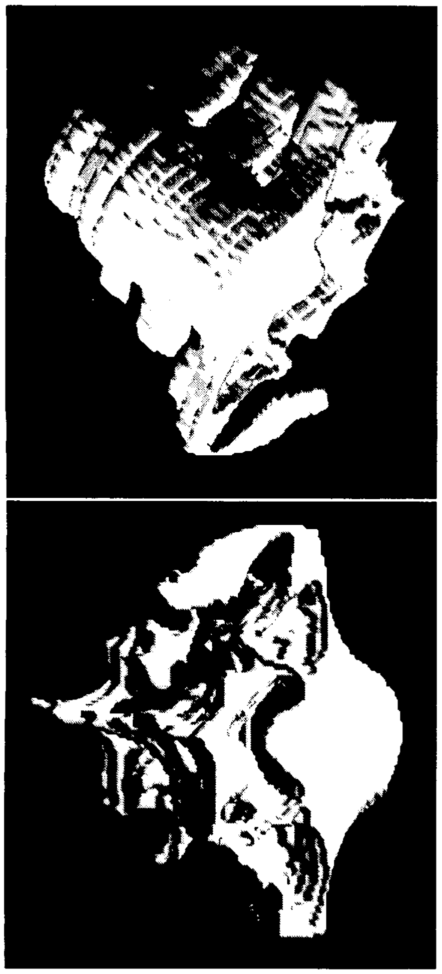

Figure 8. Different views of the CT-vertebra.

7. Conclusions and Future Work

We have presented a method to codify a surface in an octree data structure. The proposed surface codification is based on the classification of the original MC patterns in five configurations according the position and the number of planes that define the surface contained in an intersected node. Starting with the surface codified at maximal subdivision nodes, we have defined a simplification strategy that detects coplanar polygons and represents them in upper levels of the octree. Such simplification process achieves reductions of almost a 50% on the number of facets with respect to the standard MC in real medical data sets. The algorithm to reconstruct the surface of an intersected node has been also described.

The main advantages of the proposed codification are that not only reduces the output fragmentation it also maintains the connection between surface and volume data. The method is the first step on the definition of an hybrid framework able to maintain multiresolution surface and volume data renderings. Our future work will be centered on multiresolution surface fitting and volume hybrid renderings.

References

[1] I.Boada and I.Navazo: Multiresolution Isosurface Fitting using an Octree based Surface Hierachy. Research Report IIiA 01-02-RR, Institut Informàtica i Aplicacions, University of Girona, February 2001.

[2] I.Boada, I.Navazo and R.Scopigno. Multiresolution Volume Visualization with Texture-based Octree. The Visual Computer, Springer International, 2001 (in press).

[3] C.L. Bajaj, V. Pascucci and D.R. Schikore: Fast Isocontoruing for improved interactivity. In 1996 ACM Symposium on Volume Visualization, pages 39-46. IEEE Computer Society Press, Los Alamitos, CA, October 1996.

[4] P. Cignoni,P. Marno, C.Montani, E.Puppo and R.Scopigno: Speeding up isosurface extraction using interval trees. IEEE Transactions on Visualization and Computer Graphics, 3(2):158-170, 1997.

[5] P. Cignoni, C.Montani, E.Puppo and R.Scopigno: Optimal Isosurface Extraction from Irregular Volume Data. In 1996 Symposium on Volume Visualization, pages 31-49. IEEE Computer Society, IEEE Computer Society Press. Los Alamitos, CA, October 1996.

[6] P.Cignoni, C. Montani and R.Scopigno. A comparison of mesh simplification algorithms. Computer and Graphics 22 (1), pp.37-54, 1998

[7] M.J. Durst: Additional Reference to Marching Cubes (letter). Computer Graphics 22(4), pages 72-74, 1988.

[8] M. Deering: Geometry Compression. In Computer Graphics (SIGGRAPH '95 Proceedings), pages 13-25, 1995.

[9] F. Evans, S. Skiens and A. Varshney: Optimizing triangle strips for fast rendering. In IEEE Visualization 1996, pages 319-326, 1996.

[10] Hugues Hoppe: Progressive Meshes. Computer Graphics (SIGGRAPH 96 Proceedings), 99-108, 1996.

[11] W.Lorensen and H.Cline: Marching cubes a high resolution 3D surface construction algorithm, ACM Computer Graphics (Proceedings of SIGGRAPH '87), vol.21, n 4, pp 163-170, 1987.

[12] E.LaMar, B.Hamman and K.Joy. Multiresolution Techniques for interactive texture-based volume visualization. In IEEE Visualization 99. IEEE CS Press, pp.355-361 October 1999.

[13] Y.Livnat, H.W. Shen and C.R. Johsons. A Near Optimal Isosurface Extraction Algorithm using the Span Space. IEEE Transactions on Visualization and Computer Graphics, 2(1):73-84, 1996.

[14] [Mur95] S.Muraki. Multiscale Volume Representation by a dog wavelet. IEEE Trans. on Visualization and Computer Graphics, 1(2):109:116, June 1995.

[15] C.Montani, R.Scateni and R.Scopigno: Discretized Marching Cubes, in Visualization '94 Proceedings, R.D. Bergeron and A.E.Kaufman, Eds. 1994, pp.281-287, IEEE Computer Society Press.

[16] C.Montani, R.Scateni and R.Scopigno: A Modified Look-Up-Table for Implicit Disambiguation of Marching Cubes. The Visual Computer 1994, (10) pp.353-355, IEEE Computer Society Press.

[17] C.Montani, R.Scateni and R.Scopigno: Decreasing Isosurface Complexity via Discrete Fitting, Computer Aided Geometric Design, 17 (2000) 207-232.

[18] H.Mueller and M.Stark: Adaptive Generation of Surface in Volume Data. The Visual Computer 9 (4), 182-199.

[19] I.Navazo: Extended Octree Representation of General Solids with Plane Faces: Model Structure and Algorithms, Computer and Graphics, vol 13, 1, 1989, pp.5-16.

[20] M.Ohlberger and M. Rumpf: Hierarchical and Adaptative Visualization on Nested Grids. Computing , 59(4), pages 269-285, 1997

[21] H.Samet. Applications of Spatial Data Structures. Addison Wesley, Reading, MA, 1990.

[22] W.Schroeder, J.A.Zarge and W.E. Lorensen: Decimation of Triangle Meshes. Proceedings SIGGRAPH 92, 65-70, 1992.

[23] R.Shekhar, E.Fayyad, R.Yagel and J.Cornhill. Octree based Decimation of Marching Cubes surfaces. Visualization 96, 335-342, 1996.

[24] H.W. Shen, C.Hansen, Y.Livnat and C.R.Johson: Isosurfacing in Span Space with Ulmost Efficiency (ISSUE). Proceedings IEEE Visualization 96, 287-294, 1996.

[25] A.Van Gelder and J. Wilhems: Topological Considerations on Isosurface Generation. ACM Transactions on Graphics, 13(4). 337-375, October 1994.

[26] R.Westemann. A Multiresolution Framework for Volume Rendering. In Proceedings of 1994 Sysmposium on Volume Visualization, pp. 51-58. ACM Press October 17-18 1994

[27] R.Westemann, L.Kobbalt and T.Erl: Real-time exploration of Regular Volume Data by Adaptative Reconstruction of Isosurfaces. The Visual Computing 1998

[28] J. Wilhems and A. Van Gelder: Octrees for Faster Isosurface generation. ACM Transactions on Graphics, 11(3). 201-227, July 1992.

[29] J.Wilhems and A. Van Gelder. Multi-dimensional Trees for Controlled Volume Rendering and Compression. In proceedings of 1994 Symposium on Volume Visualization, pp 27-34. ACM Press, October 17-18 1994.

[30] M. Weiler, R. Westermann, C. Hansen, K. Zimmerman, T.Ertl: Level-of-Detail Volume Rendering via 3D Textures. In Volume Visualization and Graphics Symposium 2000, pp.7-13, 2000.