A Topological Comparison of Surface Extraction Algorithms

C. Andújar, P. Brunet, M. Fairén, I. Navazo, A. Vinacua

Modeling, Visualization, Interaction and Virtual Reality Group
Software Department
Universitat Politècnica de Catalunya, Barcelona, Spain.
{*andujar, pere, mfairen, isabel, alvar*}@cs.upc.edu

Abstract

In many application areas, it is useful to convert the discrete information stored in the nodes of a regular grid into a continuous boundary model. Isosurface extraction algorithms differ on how the discrete information in the grid is generated, on what information does the grid store and on the properties of the output surface. Recent algorithms offer different solutions for the disambiguation problem and for controlling the final topology. Based on a number of properties of the grid's grey cells and of the reconstruction algorithms, a characterization of several surface extraction strategies is proposed. The classification presented shows the inherent limitations of the different algorithms concerning global topology control and reconstruction of local features like thin portions of the volume and almost non-manifold regions. These limitations can be observed and are illustrated with some practical examples. We review in light of this classification some of the relevant papers in the literature, and see that they cluster in some areas of the proposed hierarchy, making a case for where it might be more interesting to focus in future research.

Key words: Surface extraction, Marching Cubes algorithm, topologically consistent isosurfaces, rectangular grids, discrete volume models, topology optimization.

1 Introduction

In many applications volumes are represented by a discrete volume model, sometimes because that is the way in which data have been gathered (medical

applications, numerical simulation results), sometimes as an intermediate representation to help in achieving some objective (model reparation, character sculpting, simplification). In this context, it is often necessary to recover from this volume information a surface representing the boundary of (a portion of) the volume. It is this surface-extraction problem that we focus on in this paper.

When the volume model is built from a solid model, it may contain binary information (an in-out classification of every vertex in a grid), or it may consist of a sampling of a scalar field (at the same vertices), for example a signed distance field. More information can be stored, like Hermite data, exact intersection points, or the fact that the volume enters (through a face, for example) into a cell, but these are seldom effectively used in the literature to recover the boundary of the volume.

The extracted surface is generally modeled with a *triangle mesh* M. The classical algorithm to achieve this —the original Marching Cubes— dates back to [1]. This original algorithm was soon seen to present inconsistencies in some configurations (see [2]), and much of the work done on this algorithm thereafter has centered on avoiding such inconsistencies. Different solutions to this problem, however, naturally yield different topologies of the resulting surface. These differences arise in areas where a dimension of the volume is comparable to the scale of the discretization, either thin sheets of volume or thin tubular portions.

In this paper we propose a way to classify these algorithms depending on the nature of the cells they examine, and some properties of the surfaces they produce. We review in light of this classification some of the relevant papers in the literature, and see that they cluster in some areas of the proposed hierarchy, making a case for where it might be more interesting to focus in future research.

To this end, we will start by reviewing some of the relevant previous work in section 2, to help motivate the definitions on which we base our classification, introduced in section 3. In section 4 we discuss briefly how different algorithms behave regarding the topology of the resulting surface, and then show in section 5 how different algorithms behave in reconstructing some suitably chosen test models, in accordance with the discussion in the previous sections.

2 Surface extraction algorithms

Surface extraction algorithms differ on how the grid information is obtained, on how it is represented in the discrete model, and on how the surface is recon-

structed. Most algorithms are only using information on grid vertices. But vertex values can be obtained by a simple sampling of the original model/field [3] or by analyzing the solid volume in the vertex neighborhood [4,5].

Multiple variations of the original Marching Cubes algorithm [1] give different solutions to the surface extraction problem. However, most techniques are based on local criteria and cannot offer a direct control of topological properties of the extracted mesh. As first noted by Durst [2], the original Marching Cubes algorithm [1] may produce surfaces with holes due to topologically inconsistent decisions on the reconstruction of ambiguous faces, where independent local decisions in the two adjacent cubes may lead to two different surface reconstructions in their common face.

Disambiguation techniques reported so far have focused on two major concerns: topological consistency, i.e. producing closed surfaces by proper cube polygonalization, and topological correctness, i.e. extracting a surface faithful to the geometry of the real surface.

Inconsistency of ambiguous faces (see figure 1) appears when adjacent cubes that share such a face take different local decisions (cases c and d in Figure 1) on the reconstruction of the surface within the ambiguous face. Consistency can be solved by just considering the inside/outside node classification, regardless of the actual data values. A first solution to the consistency problem was given by preferred polarity methods. These algorithms decide how to slash ambiguous faces of a cell using a uniform criterion: always join black nodes or always join white nodes. This decision can be implemented either algorithmically [6] or by using a lookup table [7]. All these techniques are generally simple to implement although they generate arbitrary topologies.

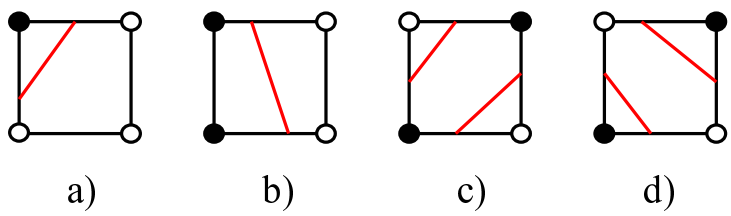


Fig. 1. Unambiguous faces (a, b) generating a single edge of the triangular mesh and the two possible choices of edges in an ambiguous face (c, d).

Techniques addressing the topological correctness problem infer the proper polygonalization of an ambiguous cube by analyzing its actual data values. These methods are required to provide different polygonalization schemes for each ambiguous cube, [8,9]. Initial methods only attempted to ensure the correctness of the returned surface on the boundary of ambiguous faces. The analysis was based on face center resampling [10,11], bilinear interpolation [12] or gradient disambiguation [13].

Initial disambiguation techniques [14,12] were trying only to disambiguate within grid faces. Nielson and Hamann [12] proposed a strategy based on the saddle point value of the bilinear interpolant to dictate the edge connections on an ambiguous face: the connection is made so as to separate the saddle point from vertices of opposite sign. This gives a solution which is topologically correct in terms of the bilinear interpolant on the face. In a similar way, Pasko et al. [14] were also taking the disambiguation decision from the position of the center of the hyperbolic trace of the trilinear interpolant on the ambiguous face. Montani et al. [15] used a simple-entry lookup table for solving the consistency problem, including an extra configuration for the complementary case in several configurations (3, 6, 7) of the classical Marching Cubes algorithm. This lookup table was also used in [16] for grids with boolean information. In this case, vertices at the middle of the grid edges were considered and a final face merging post-process was proposed for reducing the face complexity of the output model.

Tetrahedra decomposition techniques [3,17] split each cube into five or six tetrahedra, which always exhibit an unambiguous polygonalization. Gueziec et al. [18] divide each cubic cell into five tetrahedra. The first four tetrahedra are centered on four alternating vertices of the cell and split their neighbor faces through their diagonal. The last tetrahedron fills the central hole in the cubic cell. Consistency through grid faces is ensured by using alternating subdivisions: the subdivision in a particular cell is always symmetrical to the subdivision in any of its neighbor cells. The resulting topology is forced by this alternating subdivision and depends on the subdivision choice at the first cell of the grid. Pascucci [19] proposes an algorithm for isosurfacing a scalar field defined in a tetrahedral grid. His implementation is however restricted to the case of regular grids (not necessarily rectilinear). In rectilinear grids, the scheme leads to the same tetrahedral decomposition as in [18].

Some methods attempt to estimate topology also inside the ambiguous cubes either by using critical point analysis [20] or trilinear interpolation. Isosurface extraction algorithms based on trilinear interpolation address the topological correctness by imposing the local topology of the local trilinear interpolant inside every grid cubic cell. Natarajan [21] and Chernyaev [22] independently recognized that, besides the face ambiguities, there are additional ambiguities in the representation of the trilinear interpolant in the interior of the cube. In the case of a cell with two diagonally-opposite black vertices, the surface may be in two separate pieces or there may be a single tunnel piece, topologically equivalent to a cylinder (see figure 2). Natarajan used the value of the body saddle point (where all three first partial derivatives of the trilinear interpolant are zero) to discriminate the two cases. Matveyev [23] also discussed the interior ambiguity problem, resolving the ambiguous cases by considering the behavior of the trilinear function along the cell diagonals. Cignoni et al. [24] designed a new multi-entry lookup table (exhaustive lookup table, ELUT) us-

ing the Natarajan approach, which provides, for each cell configuration and for each combination of the values of the saddle points of the ambiguous faces and/or of the body saddle point, the correct isosurface patch contained in the cell. They include a final mesh refinement in order to ensure that the triangulated surface will have the same topology as the trilinear interpolant in the cell. An efficient and robust implementation of the Chernyaev method is presented in [25]. This algorithm guarantees a final two manifold surface using a battery of optimized tests and an extended Chernyaev lookup table. This sequence of papers that attempt to guarantee reconstructions with the local topology of the trilinear interpolant inside every grid cell culminates with the two papers from Nielson [26] and Lopes and Brodlie [27]. These two parallel papers present well-founded and robust reconstruction algorithms based on the analysis of face and body saddle points and give exhaustive characterizations of the possible cases.

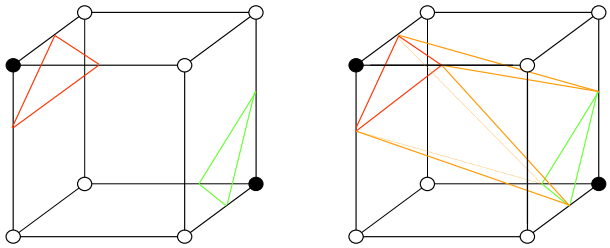


Fig. 2. The two possible topologies arising from an X-cube.

Among the algorithms that use extra information apart from values at the grid vertices we can cite the Extended Marching Cubes scheme from Kobbelt et al. [28]. Kobbelt represents a vector distance field by storing three scalar values at each grid point. In addition, a local estimation of the surface normal vector is stored at the grid vertices. The vector distance field is in fact representing the exact location of the surface intersections with the grid edges (with a maximum of one intersection per edge). Edge and vertex features can be detected and reconstructed by detecting feature cells and inserting an appropriate point in them. The Dual Contouring algorithm [29] reconstructs the surface from Hermite data: the input values are the intersection points and normal vectors of the intersections surface with the grid edges (maximum of one intersection). The algorithm uses a quadratic error metric to compute a new point inside each of the four cells around each black-white grid edge and generates a quad connecting these four new points. The algorithm is able to recover sharp features of the initial solid. Varadhan et al. [30] effectively compute up to two intersections on each edge of the grid. They then proceed with an adaptation of the Dual Contouring algorithm, resulting in a scheme that is able to effectively recover some thin features of the original. In [31] they complement this proposal with a characterization of cells where the algo-

rithm is able to reconstruct the exact topology of the original, and propose an adaptive subdivision scheme that refines the grid locally until all cells satisfy that criterion. The approach in [32] is a variation of Marching Cubes and Dual Contouring, also requiring Hermite data. A Quadric Error Metrics reconstruction is performed at each ambiguous face of a particular grid cell, computing a new isosurface point in the face. In the last step, the surface inside the cell is reconstructed.

The volume-based approach from Andujar et al. [33,34] uses the initial geometry inside the cell to detect grey cells containing parts of the initial object surface. The algorithm performs an additional subdivision of the cells and computes its vertex signs using the 26-adjacent cells. Resulting subcell configurations are guaranteed to be unambiguous. This approach allows the reconstruction of thin parts of the object and ensures a volume-Hausdorff distance condition between the initial and the reconstructed object.

A different approach for solving the topological correctness problem is presented in [35]. In this case, the optimization of the overall topological properties is achieved through a global approach. The control of the topological behavior will be discussed in Section 4.

3 Discrete volume representations: definitions and properties

Let us assume that V is a bounded closed volume in $\mathbb{R}^3$ and that S is the boundary of V. Let us also assume that we have a rectangular uniform grid R with cubical cells c with sides of length ℓ. The elements of R are its cells $c(R)$, the grid vertices $v(R)$ and the cell faces and edges, $f(R)$ and $e(R)$. We shall always consider these components as closed sets; thus, the edges include both endpoints, the faces include their edges, and the cells include all six faces. Therefore, for example, the cells $c(R)$ do not constitute a partition of R.

Definition 1 *White and black cells*

A cell c of R will be called white iff the set intersection between c and V is the void set. A cell c of R will be called black iff the set intersection between c and V is the same cell c.

Vertices v of R can also be classified as white or black in a similar way: The classification of a white vertex v is $outV$, while black vertices can be inV or onV. Edges e and faces f of R can be classified in a similar way as cells.

Cells of R not being white or black will be called grey cells. Grey cells have a non-null intersection with S. Grey cells can be of four different types:

Definition 2 *Grey-0 cells (G_0 cells)*

A grey cell c of R is in G_0 iff the number of its white vertices is greater than zero and less than eight. In other words, G_0 cells have a non-uniform set of vertices: some of their eight vertices are white while some other vertices are black.

Definition 3 *Grey-1 cells (G_1 cells)*

A grey cell c of R is in G_1 iff there exists some edge e of c such that the intersection between e and S is not null.

Definition 4 *Grey-2 cells (G_2 cells)*

A grey cell c of R is in G_2 iff there exists some face f of c such that the intersection between f and S is not null.

Definition 5 *Grey-3 cells (G_3 cells)*

A grey cell c of R is in G_3 iff the intersection between c and S is not null.

Notice that every cell in $c(R)$ is either white, black, or belongs to G_3. This again is not a partition, as a black cell whose boundary has a non-void intersection with $S = \partial V$ is also in G_3.

Abusing language we speak of G_k cells referring to cells in G_k. We will further abuse language by saying that a surface extraction algorithm is a G_k algorithm if it operates by computing G_k cells. In all cases it will be clear by the context to which of these we refer. Notice that when applied to algorithms, this classification refers to the input, or at least the part of the input the algorithm is interested in. Concerning the output, we also define:

Definition 6 *$k-$ reconstruction algorithms*

We will say that a surface reconstruction algorithm is a k-reconstruction iff the output surface is completely contained in $\cup_{c \in G_k} c$ and the output volume intersects all the cells in G_k.

3.1 Properties

From these definitions alone, a series of useful properties can be shown. We will list them here, and later apply them in the analysis of the diverse algorithms that have been proposed in the literature.

First, it is trivially seen that the G_k form a hierarchy:

Property 1 *For each $k \in \{0, 1, 2\}$, $G_k \subset G_{k+1}$*

If a cell is in G_0, it has an edge with differing classifications at the vertices. Because of Bolzano's theorem, the boundary must intersect that edge at least once, hence the cell is in G_1. The rest of the inclusions follow from the fact that we consider all elements closed, hence intersecting an edge implies intersecting the faces that share it, and intersecting a face implies intersecting the cells that share it.$\square$

If a cell is in $G_3 \setminus G_2$, then it must contain an isolated portion of V in its interior. Therefore G_3 can only differ from G_2 if V contains connected components of volume less than ℓ^3. The converse of this is obviously not true, as an arbitraryly small connected component may contain a vertex of R, but the previous obvious remark implies that

Property 2 *For any volume V, there is an $\varepsilon \in \mathbb{R}$ such that if $\ell < \varepsilon \Longrightarrow G_3(V) = G_2(V)$*

since V is bounded and closed, and hence compact.

Obviously, G_k algorithms with $k < 3$ will miss these small portions, and will not be able to offer a complete reconstruction unless ℓ is chosen sufficiently small. However G_3 may also miss some of these small portions if they do not guarantee a 3-reconstruction.

The differences between other sets in the grey hierarchy cannot be so easily characterized. Cells in $G_2 \setminus G_1$ are cells that contain a portion of the volume that enters and exits through faces of cells, without disturbing any edges. Therefore they are *thin* portions of V. Tipically these voxels are populated by pipe-like portions of the volume, but this need not be so. Given a certain rectangular grid R, consider the volume obtained for filling it up completely (taking the union of all the cells in R), and then subtracting arbitrarily small cylinders with the edges for axes. The result is a solid V whose volume is arbitrarily close to that of R, yet all the cells of R are in G_2 but not in G_1.

A similar case is $G_1 \setminus G_0$, where we can consider $V = \cup_{c \in c(R)} c \setminus \cup v \in v(R) B_v$, where B_v denotes a small open ball centered at the vertex v.

These are extreme but unreal examples in practice. We will show however with examples to which extent the problem is not so distant from practical reconstruction problems.

Finally, notice that 3-reconstructions guarantee a bound on the Hausdorff distance between $S = \partial V$ and the result of the reconstruction:

Property 3 *Given a volume V, and a 3-reconstruction V' using a grid with*

cell-size ℓ, then

$$Haus(V, V') \leq \sqrt{(3)}\ell$$

Notice that V' intersects all the cells in G_3, and there is a portion of S, and hence of V, in each cell of G_3. Furthermore each point on V is either in a black cell or in some cell in G_3, and a 3-reconstruction must produce a volume that intersects that cell. Since two points in a cell cannot be further than the stated bound, the property follows immediately.□

This desirable property, however, is not guaranteed for any other reconstruction level, as we have seen that in general $G_3 \setminus G_k \neq \emptyset$ for $k < 3$, and portions on cells in $G_3 \setminus G_k$ may lie arbitrarily far from the algorithm's output surface.

4 Topological properties of isosurfaces

The objective of this section is to characterize the surface extraction algorithms reviewed in Section 2 by using the properties defined in the previous Section. A special remark will be done on discussing the topological properties of the surfaces that they reconstruct, without considering the final application described in the papers where they are presented.

4.1 0-reconstruction algorithms

Most of the reviewed algorithms [6,7,15,17–19,28,32,35] are variations of the original Marching Cubes algorithm [1]. They have a set of common characteristics: The input information is the classification of the grid vertices with respect to the volume (they only detect G_0 cells), the output surface is a triangle mesh reconstructed locally for each G_0 cell. Each triangle of the final mesh belongs to a unique cell, and the output surface S' intersects only once each black-white edge of the grid. These properties guarantee that the output surface stabs all G_0 cells, so all these algorithms are $0-$reconstructions.

The main difference among the former algorithms lies in the method they use to perform the local reconstruction of the triangle mesh M. There are two decisions to consider: the selection of the local topology of M in a cell and the triangulation of the resulting connected components (sheets) in such a way that it has no impact on the selected topology. [35] shows that the only MC configurations giving choices to control the local topology are those that correspond to cells having ambiguity faces (also called *X-faces*, see Figure 1),

or having only two diagonally-opposite *black* vertices (also called *X-cubes*, see Figure 2).

We can group the Marching Cubes based algorithms in three families according to how they resolve the local ambiguities: those using a tetrahedral subdivision, those using a Single-entry LUT, and those using (implicitly or explicitly) a Multiple-entry LUT. Single-entry LUTs only use the black/white classification of the grid vertices, while Multiple-entry LUTs use the scalar values at grid vertices.

Tetrahedral subdivision approaches [17–19] perform a conformal subdivision of the G_0 cells. Since the intersection of a piecewise-linear surface with the black-white edges of a tetrahedron is univocally defined, the surface reconstruction based on this subdivision yields directly a valid two-manifold surface. However, the topology inside each cube is fixed by the tetrahedral subdivision and there is no control on the global topology of the reconstructed surface. When the grid vertices store a scalar value, the location of the vertices of M can be interpolated on a black-white tetrahedron edge, resulting in a smoother surface.

In Single-entry LUT approaches, the local topology of a G_0 cell is univocally defined by the classification of its vertices. The LUT is indexed by this classification of the vertices [15]. Some proposals do not compute the LUT explicitly but they take an equivalent decision algorithmically [6]. All of these methods can achieve consistent topologies on ambiguous faces. The triangulation of a configuration can be done by always locating the mesh vertices in the black-white grid edges [16] or by inserting additional vertices into the cell for recovering sharp features of the original surfaces [28]. In this case they use Hermite data (position of intersection points of the original surface with grid edges and their normals). The Single-entry algorithms obtain valid and locally consistent surfaces but they do not have control on the global topology of S' which is instead determined by the preferred local reconstruction stored in the LUT (i.e. they cannot decide the number of holes or shells of S').

Multiple-entry LUT approaches try to reproduce in each cell the topology of certain interpolation function. Most of the published papers assume a local bilinear or trilinear interpolation function [13,12,24–27]. From the vertex values, they compute the value of the function in its saddle points in the cell and its faces. The classification of the grid vertices allows to choose a basic *MC configuration* and the computed saddle points allow to choose among all the possible topologies (for a given configuration) the one which corresponds to the preferred function. Recently, Ho et al. [32] have proposed an algorithm that uses the Hermite data to decide the topology in ambiguous faces (X-faces) and in X-cubes. It computes additional points in cell faces from the Hermite information and uses these additional points to disambiguate and reconstruct

the interior of a cell. The algorithm also recovers local sharp features by introducing mesh vertices in G_0 cells. All the algorithms of this group obtain valid, consistent and correct surfaces (according to the preferred function or shape) but do not have a global control on the topology of the final mesh.

The algorithm by Ju et al. [29] is not based on Marching Cubes. The input information is the classification of the grid vertices with respect to the volume. For each G_0 cell, the algorithm inserts a vertex of the final mesh. Based on the hypothesis that each black-white edge of the grid stabs once the final mesh, the algorithm generates a quad by joining the four points located in the edge neighbour cells. This reconstruction method guarantees that the output surface stabs all G_0 cells, so the algorithm is a $0 - reconstruction$. Additionally to the black/white classification of the grid vertices, the input data includes Hermite data which is used to properly locate the interior mesh vertices in the cells trying to preserve the shape of the original surface. This algorithm reconstructs surfaces with non-manifold topology in ambiguous cells (see Figure 5(d)). Note that the reconstruction is also local for each white-black grid edge, so it has no control on the global topology of the resulting surface.

Finally, the algorithm presented in [35] is also based on Marching Cubes. It is the first published approach for selecting amongst all valid two-manifold topologies of the extracted triangle mesh M the one that optimizes a desired topological and combinational complexity mesure, which can be the total triangle count, the number of connected components, or the total genus. In this approach the vertices of the final mesh are located in black-white edges of the original grid. The topological measures are not affected by the placement of these vertices so the information stored at the grid vertices is boolean and indicates whether a specific grid vertex is inside or outside the final surface.

The paper identifies two independent ways to control the topology of the final mesh, by deciding how to slash the $X - faces$ (see Figure 1) and by deciding to have one or two sheets in an $X - cube$ (see Figure 2). The reconstruction of the final mesh is locally performed per each G_0 cell. For non ambiguous cells the reconstruction follows the classical MC algorithm. For the ambiguous cells, in order to take locally and consistent decisions with respect to the user-desired global topology, the paper proposes the use of two new auxiliary data structures: the *X-face propagation graph* and the *Merge tree of equivalence classes of vertices*. The *X-face propagation graph* is a convenient tool for deciding on X-face slashing. The second data structure is related with the *equivalence classes* of vertices. These equivalence classes encode clusters of vertices of the final mesh that belong to the same shell. The final algorithm, after initializing the two mentioned data structures, optimizes the mesh topology by traversing the X-face graph while taking some atomic decisions on how to slash the individual X-faces, and finally by deciding on how to connect (or not) the loops inside X-cubes. The authors identify sixteen possible combined

decisions that can be taken during the traversal of the X-face graph and the visit of the X-cubes. They have developed four of them which allow to optimize some topological properties of the final mesh, for example the number of triangles and the number of connected components (see Figures 6(e) and 6(f)).

4.2 1-reconstruction algorithms

Varadhan [30] uses directed distances at grid vertices to perform an exact edge-intersection test. This edge intersection test is used to detect complex edges, i.e. edges intersected by the surface but not exhibiting a sign change (thus identifying $G_1 \setminus G_0$ cells). The reconstruction algorithm is very similar to the dual contouring algorithm [29] but it considers that an edge can have up to two intersection points and that there can be multiple error-minimizing vertices per cell. Once the edge intersection points have been computed, it separates them into components achieving a two-manifold surface. It also recostructs some sharp features, but does not have a global control on the topology of the final mesh.

4.3 3-reconstruction algorithms

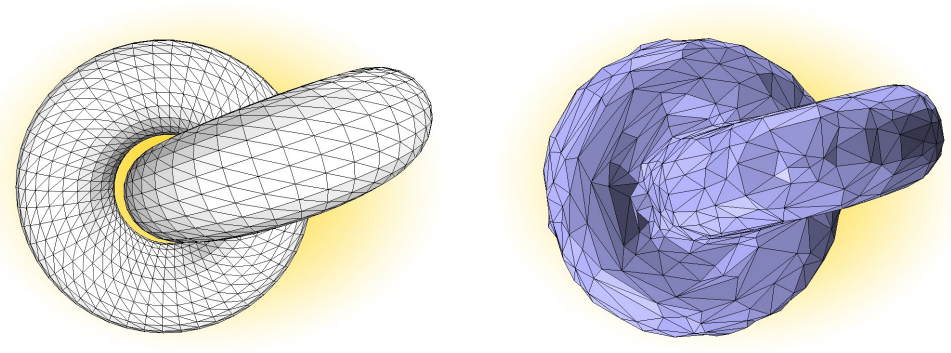
An extension of [30] is presented in [31]. Instead of considering just complex edges (leading to G_1 cells) G_2 and G_3 cells are also identified and recursively subdivided. A cell is complex if it has a complex voxel, face, edge, or an ambiguous sign configuration. A voxel (face) is defined to be complex if it intersects the surface and the grid vertices belonging to the voxel (face) do not exhibit a sign change. Their algorithm subdivides recursively the space using an octree until all cells are not complex and satisfy a star-shaped criterion. The provided subdivision algorithm is guaranteed to terminate if the initial surface is a closed manifold, free from artifacts such as self intersections and if there is no tangential contact between different primitives. The proposed reconstruction algorithm is able to preserve the original topology of the surface but at the expense of an arbitrary number of space subdivisions.

Andujar et al. [33] proposes another G_3 surface extraction algorithm. Given a volume V bounded by a surface S, the intersection between the cells of the grid and S is detected and for each of these G_3 cells the classification of their vertices with respect to the volume is computed and stored. The surface reconstruction algorithm first performs exactly one additional subdivision of the G_3 cells that have at least one white neighbour cell. The classification of the new grid vertices is computed using the type of their neighbour cells. A vertex having a white neighbour cell is classified as white and otherwise it is

classified as black. The central vertex of a subdivided cell is always classified as black. Resulting subcells do not have ambiguous faces and their topology is decided using the LUT of the basic MC algorithm [1].

According to *Definition 6* these two algorithms are *3-reconstruction* as the grey cells used for computing S' are inside of G_3 cells, the surface S' and the output volume V' stab G_3 cells.

Figure 4 shows that [33] is able to reconstruct thin areas of the initial volume, while G_0 algorithms fail on this. In some situations the algorithm tends to connect portions of volume slightly separated (see Figure 3), this is due to the fact that G_3 cells without any white neighbour are not considered by the reconstruction algorithm. However, these cells will wind up inside of V', so the algorithm is not loosing any volume information and fulfills *Property 3* that guarantees the tolerance of the resulting volume V' with respect to V.



(a) Original model (b) 3-reconstruction following [33]

Fig. 3.

5 Results of surface extraction algorithms

In this section we compare some of the algorithms reviewed in Section 2 with respect to their characteristics (see Table 1). We present also some experimental results which highlight the topological differences among several methods.

The input information the algorithms use to decide ambiguous cases (*input info to disamb.*) varies between those that use binary information (black/white) at the grid vertices [16,28,29,33–35] and those that use scalar values in the vertices [18,19,24,27,26,30].

The column on topological control (*Topol. control - local/global*) summarizes how the different algorithms deal with topology. Only [35] can perform global

Paper	Type G_k	Input info. to disamb.	Topol. control local/global	2-manifold surface	Reconst. *thin* parts	Reconst. alg.
Gueziec-95 [18] Pascucci-04 [19]	G_0	binary B/W	Local (alternancy based)	Yes	No	$0-rec.$
Montani-94 [16] Kobbelt-01 [28]	G_0	binary B/W	Local (LUT based)	Yes	No	$0-rec.$
Cignoni-00 [24] Lopes-03 [27] Nielson-03 [26]	G_0	vertex values	Local (trilinear based)	Yes	No	$0-rec.$
Ho-05 [32]	G_0	binary B/W + hermite data	Local (hermite based)	Yes	No	$0-rec.$
Tao-02 [29]	G_0	binary B/W	Local (non-manifold)	No	No	$0-rec.$
Varadhan-03 [30]	G_1	directed distances	Local	Yes	Some	$1-rec.$
Andujar-02 [33]	G_3	binary B/W extra subdiv.	Local (W-surface)	Yes	Yes	$3-rec.$
Varadhan-04 [31]	G_3	directed distances extra subdiv.	N/A	Yes	Yes	$3-rec.$
Andujar-05 [35]	G_0	binary B/W	Global (optimization)	Yes	No	$0-rec.$

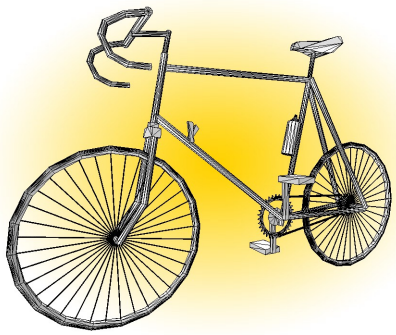
Table 1
Classification of the algorithms with respect to several characteristics

optimizations of the topology. As for [31], it is rated "not applicable" since in this case the algorithm always reproduces the topology of the original surface. However, there is no control available to change it. The other algorithms behave locally. Some [18,19] resort to a subdivision in the tetrahedra of the hexahedral cells, arbitrarily fixing the topology within each cell. Others use extended lookup tables [16,28], or attempt to reproduce the topology of a trilinear interpolant for the scalar values at the vertices [24,27,26]. The approach used in [32] exploits the Hermite data to decide the topology within each cell. In the case of [29], ambiguous cells give rise to unambiguous triangulations which are not two-manifold. Finally, [33] inherits the topology of the surface that separates grey from white cells, except when this surface is not a two-manifold, when the solution is a two-manifold which represents the perturbation of that singularity where the sheets separate receeding into grey cells.

Regarding the remaining three columns (*Type* G_k, *Reconst. thin parts* and *Reconst. alg.*) we can conclude that the only algorithms using G_3 cells, being able to reconstruct *thin* parts of the model and being 3-reconstruction algorithms are [33,31]. The rest of the algorithms use G_0 cells, do not reconstruct *thin* parts and are 0-reconstruction algorithms.

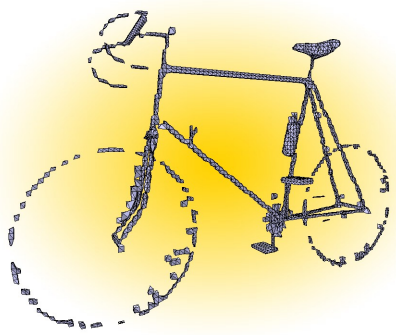
Example models have been chosen specifically to show those charateristics that

differentiate their results. The first model we show is a bicycle (see Figure 4), where we can distinguish many different *thin* parts (e.g. the spokes). This figure clearly illustrates the reconstruction differences between G_0 and G_3 approaches. Algorithms based on G_0 cells are not able to reconstruct the complete surface of the model, unless the number of subdivision levels is high enough.

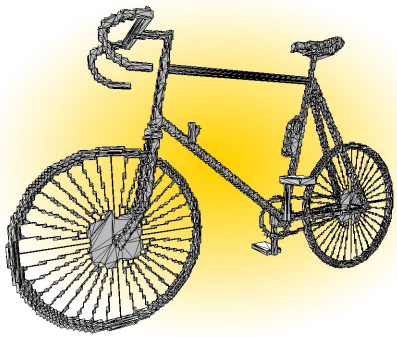


(a) Original model

(b) Trilinear disambiguation (G_0)



(c) Dual Contouring (G_0)



(d) *3-Reconstruction* following [33]

Fig. 4. Different results between G_0 and G_3 algorithms. Model generated with an octree of seven subdivision levels.

The model in Figure 5(a) represents two tori. Its discrete model has several cells with ambiguous MC configurations. The figure shows the comparisons reached by the different algorithms. The algorithms that alternate a tetrahedrization to achieve consistency (Figure 5(b)), sometimes join the ambiguous cubes and sometimes separate them, but the decision adopted has no relation with the topology of the original surface. Algorithms based on a trilinear disambiguation (subfigure 5(c)) are at an advantage with this test case, as the very smooth and regular geometry is approximated very well by a trilinear surface in each cell. Note that algorithms based on dual contouring produce non-manifolds on ambiguous configurations (see Figure 5(d)). The

last two figures are generated by [35] using two diferent optimization strategies. Strategy used for Figure 5(e) minimizes the number of solid components, while Figure 5(f) shows the result of adopting the oposite strategy and it is the only one recovering the initial two shells.

The discretization of the model in Figure 6(a) presents abundant ambiguous faces and cells. This example shows clearly how the alternating methods (Figure 6(b)) arbitrarily join or separate portions of the model. The trilinear approaches (Figure 6(c)) failed to reconstruct diagonals. Figure 6(d) clearly shows how the dual contouring algorithm creates non-manifold vertices and edges. Using the same two strategies as in Figure 5, [35] yields the solutions in Figures 6(e) and 6(f). Note that the topology of Figure 6(e) exactly matches the original model.

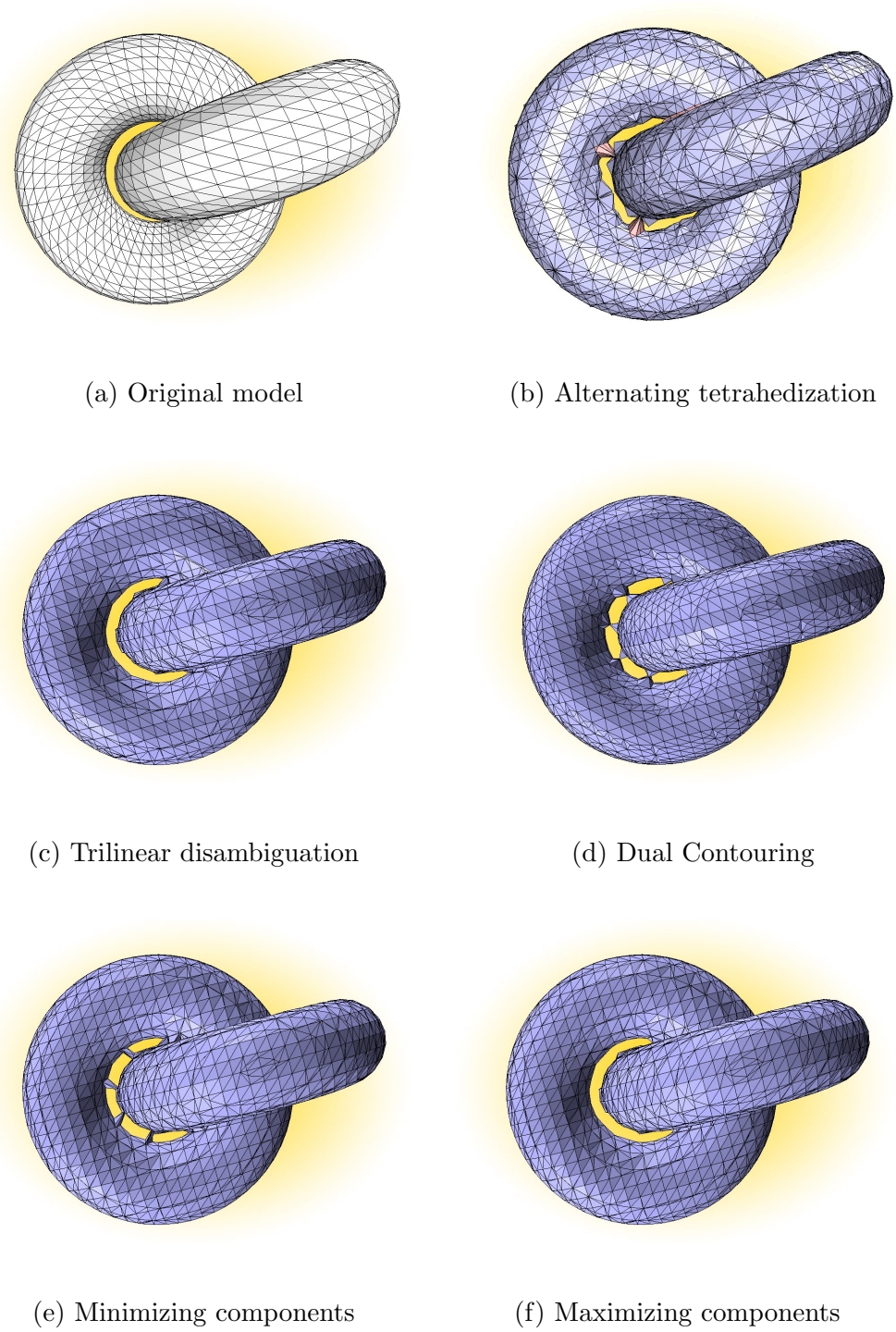
6 Conclusions

Isosurface extraction algorithms differ on how the discrete information in the grid is generated, on what information does the grid store and on the properties of the output surface. Recent algorithms offer different solutions for the disambiguation problem and for controlling the final topology.

Based on a number of properties of the grid grey cells and of the reconstruction algorithms, a characterization of several surface extraction algorithms has been proposed. The classification in Table 1 presents the inherent limitations of the different algorithms concerning global topology control and reconstruction of local features like thin portions of the volume and almost non-manifold regions. These limitations have been observed and discussed in some practical examples.

Most of the algorithms discussed are of type G_0 and only present local topology control. Non-global topology control usually leads to some arbitrary final topology which will not fulfill the user requirements and/or targets. G_k algorithms with $k > 0$ are scarce. In some cases, even if Hermite data are stored in grid edges, the algorithms remain G_0 because they operate by computing G_0 cells and do not consider grid edges with multiple surface intersections (with the sole exception of [30]). As observed in Property 2, G_k algorithms with $k < 3$ will miss small portions of the volume unless the grid edge size is chosen sufficiently small.

A potential line for future work is the development of new two-manifold surface extraction algorithms with the 3-reconstruction property and/or global topology control of the output surface. As stated in Property 3, 3-reconstruction schemes exhibit a bounded Hausdorff distance property that guarantee a sim-



(a) Original model

(b) Alternating tetrahedization

(c) Trilinear disambiguation

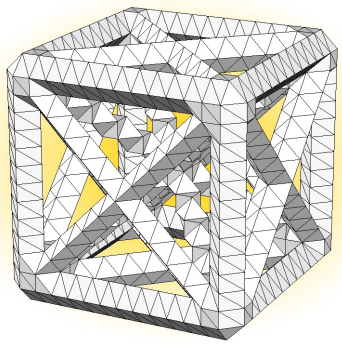
(d) Dual Contouring

(e) Minimizing components

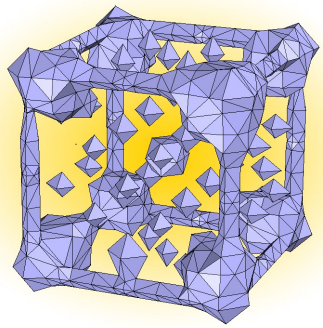
(f) Maximizing components

Fig. 5. Different results from different G_0 algorithms. The model has been generated with five octree subdivision levels.

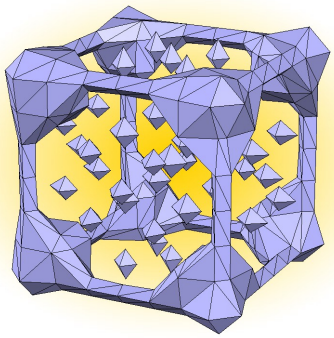
ilarity between the initial volume and the reconstructed one.



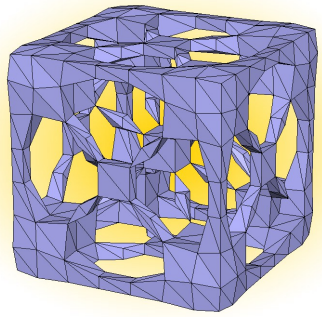
(a) Original model



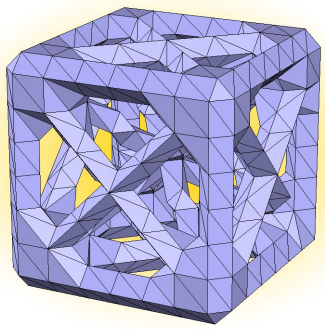
(b) Alternating tetrahedrization



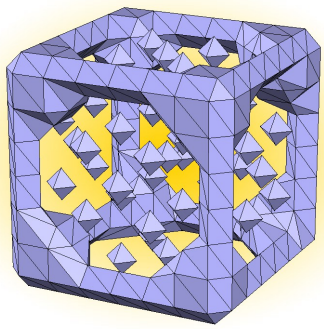
(c) Trilinear disambiguation



(d) Dual Contouring



(e) Minimizing components



(f) Maximizing components

Fig. 6. Different results from different G_0 algorithms. The model has been generated with four octree subdivision levels.

References

[1] W. Lorensen, H. Cline, Marching cubes: A high resolution 3D surface construction algorithm, Computer Graphics 21 (4) (1987) 163–169.

[2] M. J. Dürst, Letters: Additional reference to marching cubes, Computer Graphics 22 (2) (1988) 72–73.

[3] G. Nielson, T. Foley, B. Hamann, D. Lane, Visualizing and modeling scattered multivariate data, IEEE Computer Graphics and Applications 11 (3) (1991) 47–55.

[4] T. He, L. Hong, A. E. Kaufman, A. Varshney, S. W. Wang, Voxel based object simplification, in: IEEE Visualization, 1995, pp. 296–303.

[5] T. He, L. Hong, A. Varshney, S. W. Wang, Controlled topology simplification, IEEE Transactions on Visualization and Computer Graphics 2 (2) (1996) 171–184.

[6] J. Bloomenthal, An implicit surface polygonizer, in: P. S. Heckbert (Ed.), Graphics Gems IV, Academic Press, 1994, pp. 324–349.

[7] J.-O. Lachaud, Topologically defined iso-surfaces, in: Proc. 6th Discrete Geometry for Computer Imagery (DGCI'96), Lyon, France, Springer-Verlag, Berlin, 1996, pp. 245–256.

[8] P. Ning, J. Bloomenthal, An evaluation of implicit surface tilers, IEEE Computer Graphics and Applications 13 (6) (1993) 33–41.

[9] S. Hill, J. C. Roberts, Surface models and the resolution of n-dimensional cell ambiguity, in: A. W. Paeth (Ed.), Graphics Gems V, Academic Press, 1995, pp. 98–106.

[10] A. Wallin, Constructing isosurfaces from ct data, IEEE Computer Graphics and Applications 11 (6) (1991) 28–33.

[11] G. Wyvill, C. McPheeters, B. Wyvill, Data structures for soft objects, The Visual Computer 2 (4) (1986) 227–234.

[12] G. Nielson, B. Hamann, The asymptotic decider : Resolving the ambiguity in marching cubes, in: Proc. of IEEE Visualization 91, 1991, pp. 83–91.

[13] J. Wilhelms, A. V. Gelder, Topological considerations in isosurface generation, Computer Graphics 24 (5) (1990) 79–86.

[14] A. A. Pasko, V. V. Pilyugin, Geometric modelling in the analysis of trivariate functions, Computers & Graphics 12 (3-4).

[15] C. Montani, R. Scateni, R. Scopigno, A modified look-up table for implicit disambiguation of marching cubes, The Visual Computer 10 (6) (1994) 353–355.

[16] C. Montani, R. Scateni, , R. Scopigno, Discretized marching cubes, in: IEEE Visualization, 1994, pp. 281–287.

[17] C. Zahlten, Piecewise linear approximation of isovalued surfaces, in: F. H. Post, A. J. S. Hin (Eds.), Advances in Scientific Visualization, Springer-Verlag, 1992, pp. 105–118.

[18] A. Gueziec, R. Hummel, Exploiting triangulated surface extraction using tetrahedral decomposition, IEEE Transactions on Visualization and Computer Graphics 1 (4) (1995) 328–342.

[19] V. Pascucci, Isosurface computation made simple: Hardware acceleration, adaptive refinement and tetrahedral stripping, in: EG-TVCG Symposium on Visualization, 2004.

[20] B. T. Stander, J. C. Hart, Guaranteeing the topology of an implicit surface polygonization for interactive modeling, Computer Graphics (SIGGRAPH 97 Proceedings) 31 (1) (1997) 279–286.

[21] B. K. Natarajan, On generating topologically consistent isosurfaces from uniform samples, Visual Computer 11 (1) (1994) 52–62.

[22] E. Chernyaev, Marching cubes 33: Construction of topologically correct isosurfaces, Tech. Rep. CN/95-17, CERN, http://wwwinfo.cern.ch/asdoc/psdir/mc.ps.gz (1995).

[23] S. Matveyev, Approximation of isosurface in the marching cube: Ambiguity problem, in: IEEE Visualization, 1994, pp. 288–292.

[24] P. Cignoni, F. Ganovelli, C. Montani, R. Scopigno, Reconstruction of topologically correct and adaptive trilinear isosurfaces, Computers and Graphics 24 (3) (2000) 399–418.

[25] T. Lewiner, H. Lopes, A. W. Vieira, G. Tavares, Efficient implementation of marching cubes' cases with topological guarantees, Journal of Graphics Tools 8 (2).

[26] G. Nielson, On marching cubes, IEEE Transactions on Visualization and Computer Graphics 9 (3) (2003) 283–297.

[27] A. Lopes, K. Brodlie, Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing, IEEE Transactions on Visualization and Computer Graphics 9 (1) (2003) 16–29.

[28] L. P. Kobbelt, M. Botsch, U. Schwanecke, H. P. Seidel, Feature sensitive surface extraction from volume data, ACM Computer Graphics (Siggraph 2001) (2001) 57–66.

[29] T. Ju, F. Losasso, S. Schaefer, J. Warren, Dual countouring of hermite data, ACM Transactions on Graphics 21 (3) (2002) 339–346, proc of Siggraph'02.

[30] G. Varadhan, S. Krishnan, Y. Kim, D. Manocha, Feature-sensitive subdivision and isosurface recostruction, in: IEEE Visualization, 2003, pp. 99–106.

[31] G. Varadhan, S. Krishnan, T. Sriram, D. Manocha, Topology preserving surface extraction using adaptive subdivision, in: Symposium on Geometry Processing, 2004, pp. 241–250.

[32] C.-C. Ho, F.-C. Wu, B.-Y. Chen, Y.-Y. Chuang, M. Ouhyoung, Cubical marching squares: Adaptive feature preserving surface extraction from volume data, Computer Graphics Forum (Eurographics 2005) 24 (3) (2005) 537–546.

[33] C. Andujar, P. Brunet, D. Ayala, Topology-reducing simplification through discrete models, ACM Transactions on Graphics 20 (6) (2002) 88–105.

[34] C. Andujar, M. Fairén, P. Brunet, V. Cebollada, Advances in Multiresolution for Geometric Modelling, Springer-Verlag, 2005, Ch. Simplification of Topologically Complex Assemblies, pp. 339–352, iSBN 3-540-21462-3.

[35] C. Andujar, P. Brunet, A. Chica, J. Rossignac, I. Navazo, A. Vinacua, Optimizing the topological and combinational complexity of isosurfaces, Computer-Aided Design 37 (8) (2005) 847–857.