Topological Considerations in Isosurface Generation

UCSC-CRL-94-31

Allen Van Gelder and Jane Wilhelms
Baskin Center for Computer Engineering and Information Sciences
University of California, Santa Cruz

June 11, 1994

Abstract

A popular technique for rendition of isosurfaces in sampled data is to consider cells with sample points as corners and approximate the isosurface in each cell by one or more polygons whose vertices are obtained by interpolation of the sample data. That is, each polygon vertex is a point on a cell edge, between two adjacent sample points, where the function is estimated to equal the desired threshold value. The two sample points have values on opposite sides of the threshold, and the interpolated point is called an *intersection point*.

When one cell face has an intersection point in each of its four edges, then the correct connection among intersection points becomes ambiguous. An incorrect connection can lead to erroneous topology in the rendered surface, and possible discontinuities. We show that disambiguation methods, to be at all accurate, need to consider sample values in the neighborhood outside the cell. This paper studies the problems of disambiguation, reports on some solutions, and presents some statistics on the occurrence of such ambiguities.

A natural way to incorporate neighborhood information is through the use of calculated gradients at cell corners. They provide insight into the behavior of a function in well-understood ways. We introduce two *gradient-consistency heuristics* that use calculated gradients at the corners of ambiguous faces, as well as the function values at those corners, to disambiguate at a reasonable computational cost. These methods give the correct topology on several examples that caused problems for other methods we examined.

Categories and Subject Descriptors: I.3.3 [**Computer Graphics**]: Picture/Image Generation — display algorithms; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling — boundary representations; curve, surface, solid, and object representations; geometric algorithms

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: Surface topology, ambiguity, surface fitting, scientific visualization, isosurface extraction

This report will appear in *ACM Transactions on Graphics*. Some images have low resolution to facilitate electronic distribution.

1 Introduction

The sampled volumetric data common to many scientific applications has been visualized using a variety of approaches. One is to generate approximate *isosurfaces* that correspond to specific threshold values within the data [FKU77, CS78, AFH80, KDK86, WMW86, LC87, CDL+87, Blo88, CLL+88, Win88, GN89, KB89, UA90, Kal91, KDHB92]. An isosurface clearly indicates where in space the volumetric data values at a given threshold value lie.

Polygonized isosurfaces can be used with common graphics rendering algorithms available in hardware and software, and provide the possibility of realtime interaction with the data. However, the clarity of the isosurface may itself be misleading. For many applications, the underlying function that the sampled data represents is not known, and the isosurface produced is at best a good guess. While the exact spatial location of the isosurface may not be critical, its topological features, such as connected components and the presence of holes (tunnels) or cavities, are normally important.

This paper explores the problems of generating topologically correct approximate isosurfaces from sample data where the underlying function is not available for resampling. Sections 2 and 3 characterize situations in which topological ambiguities can arise in cell-at-a-time processing (a cell normally consisting of eight data values at the corners of a cube). Section 4 discusses several approaches to disambiguation. We demonstrate that correct disambiguation on any reasonably broad class of volumes *must* consider sample data values besides those at the eight cell corners, and propose several techniques. Sections 5 and 6 describe experimental results on scientific data. Section 7 draws conclusions, and some technical matters are covered in appendices.

2 Design Objectives of Isosurface Algorithms

Many of the isosurface techniques in the literature were designed for specific applications. As a result, they may have implicit assumptions about the nature of the data that would not hold in another application. As applications proliferate it becomes important to have a general-purpose method that is free of application dependencies. Toward this end we identified a number of desirable features of a general-purpose polygonal isosurface algorithm:

1. The algorithm should yield a continuous surface: Each polygonal edge should be shared by exactly two polygons, or lie in an external face of the entire volume [Sri81, USH82].

2. The isosurface should be a continuous function of the input data: A small change in the threshold value or some data value should produce a small change in the isosurface.

3. The isosurface should be topologically correct when the underlying function is "smooth enough".

4. The isosurface produced should be neutral with respect to positive and negative sample data values (relative to the threshold): Multiplying the samples (and threshold) by -1 should not alter the surface. In this criterion we depart from earlier studies [LV80, Sri81, USH82] in which the "object" (usually represented by positive values) and its complement are assigned different degrees of connectivity.

5. The algorithm should not create artifacts not implied by the data, such as bumps and holes.

6. Preferably, the algorithm should be efficient enough for real-time interactive use.

Some of these criteria may not seem important when the resolution is fine enough that the eye does not notice an occasional "glitch". However, visualization systems will inevitably provide a zoom ability for close-up examination of "interesting" features of a scene. Incorrect topology can easily become visible and lead to misleading or at least confusing images under close-up examination, regardless of the original resolution.

Of the above criteria, only objective 4 is likely to be controversial. This issue is discussed further in Section 5.1.

3 Isosurface Generation

Two approaches have been taken to represent regions of constant value within the volume data. The first, the *cuberille* method, considers the subdivisions of space created by the sample points and represents constant value regions as a set of polyhedra (generally cubes) that include the values of interest [HL79, AFH81, HU83, CHRU85, Udu89]. The second approach, explored in this paper, is to generate a two-dimensional isosurface in three dimensions by interpolating the function values between data points. We shall call this the *beveled-surface* method, following Kalvin's terminology. Each "method" is actually a group of methods described by various researchers.

Two advantages of the cuberille method are that it is fast, and it does not attempt to generate a surface in more detail than the data unambiguously specifies. One disadvantage is that the image produced is naturally blocky, because all surface patches are orthogonal to one of the coordinate axes. However, it can be made to appear smoother by the deft use of surface shading and filtering [CHRU85, Udu89]. A second problem is that the surface changes if positive and negative are inverted; the changes may be topological as well as spatial. These changes result from the difference between 1-connectivity and 3-connectivity [USH82] (see Definition 3.2). See Section 5.1 for further discussion.

Beveled surfaces permit patches with any orientation, so have the potential of providing a closer approximation to the underlying surface. However, this greater generality may carry with it greater costs in both processing times and storage space. Many of the methods described in the literature have the same problem that inversion of positive and negative can cause both spatial and topological changes to the surface. Efficiency problems are addressed elsewhere [Kal91, WVG92]. This paper addresses the topological problems in beveled surface construction.

Early isosurface approaches first generated two-dimensional contour lines for parallel planes running through the data, and then connected the contour lines into three-dimensional isosurfaces [FKU77, CS78, WH79, CIBL83]. More recent approaches create the isosurface by examining the three dimensions at once.

Wyvill *et al.* [WMW86] developed a method that represents the isosurface as a polygon mesh. They compute samples in a three-dimensional rectangular lattice and analyze cells in this lattice individually. Although they were primarily interested in an application for which they knew the underlying function, their method of constructing the isosurface within each cell does not require such information, so can be used in a more general setting. It is summarized in Section 3.2, and serves as a starting point for our work. They recognize ambiguities, and disambiguate by estimating the function value at the center of an ambiguous face; the corners that agree with the central estimate are considered connected. In the interest of efficiency, they estimate the center value by averaging the four corner values (the *facial average* method, discussed later).

Koide *et al.* reported a more complex method based on decomposition of the cell into tetrahedra, and polygonizing the isosurface within each tetrahedron [KDK86]. Ambiguities are implicitly resolved by the decomposition.

Lorensen *et al.* reported a simpler approach [LC87, CLL$^+$88], where the data samples were essentially the only information available about the underlying function. Their method, called "marching cubes", achieves performance by polygonizing each cell based on a precomputed table of 15 topologically distinct plus-minus patterns of cell corners. In the original implementation, "marching cubes" did not recognize ambiguities. As pointed out by Dürst [Dür88], that method could yield a discontinuity between cells. Later researchers [Bak89, Nat91, Kal91], as well as the original authors, described modifications that ensured continuity. Some variants are discussed in Section 4.1.

Gallagher and Nagtegaal [GN89] generalized the above approaches to irregular lattices that frequently occur in finite element analysis, and considered higher-degree surfaces instead of triangulation. They also do not address ambiguities. Winget also briefly discusses isosurfaces resulting from finite element analysis

[Win88].

Subsequent to the presentation of a preliminary version of this work at a workshop [WVG90], other researchers have developed additional methods to resolve ambiguities using an assumption of trilinearity in each cell [Nat91], or bilinearity in each face [NH91].

3.1 Nature of the Data

Isosurfaces may be used to represent scalar data values that are distributed in a three-dimensional coordinate system, such as density values generated by a CT-scan or temperature values from computational fluid dynamics. For this paper, we assume scalar data distributed in a regular three-dimensional rectangular lattice, though the issues raised are equally important to irregular and sparse data. The algorithms generalize straightforwardly to "warped" lattices, in which each cell still has eight corners.

Definition 3.1: A data sample is referred to as a *voxel*. The cubical region bounded by eight neighboring voxels is called a computational *cell*, and the cell's corners called *cell vertices*. □

Strictly speaking, a cell is a rectangular parallelepiped, but by rescaling Δx, Δy, and Δz to 1 we can simplify the nomenclature and presentation.

There are several notions of neighborhood (adjacency) in a rectangular lattice. Unfortunately, the terminology is not uniform in the literature [LV80, Sri81, USH82]. The next definition gives the terminology we shall use, and the alternates seen. (See also Section 4.1.)

Definition 3.2: Two cell vertices are said to be *k-adjacent* if they differ in at most k coordinates, and differ by at most 1 in any coordinate. Thus, they are:

1-adjacent if they are connected by a cell edge; this type of adjacency has also been called $O(6)$-adjacency because a cell vertex has 6 neighbors with this type of adjacency.

2-adjacent if they have a cell face in common (also called $O(18)$-adjacent).

3-adjacent if they have a cell in common (called $O(26)$-adjacent).

We abbreviate *1-adjacent* to *adjacent* when no confusion is likely.

A set of cell vertices is *k-connected* if, by defining an edge between every k-adjacent pair in the set, the result is a connected graph. □

3.2 Nature of the Isosurface

An isosurface is implicitly specified by an underlying scalar function of three variables and a threshold value. Following Wyvill *et al.* [WMW86], who call the underlying function the "field function", generation of an isosurface involves determining, for each cell, whether the underlying function takes on the threshold value within the cell, and if so, approximately where the isosurface lies.

We shall call a cell vertex *positive* if its value is greater than the threshold, and *negative* if not; thus our terminology translates the threshold to the origin. (The case is which the sample value is exactly at the threshold introduces technical problems. We elide these problems in practice by choosing the threshold to be distinct from all sample values.) When some vertices of a given cell are positive and some are negative, the isosurface must pass through the cell and the problem becomes finding where it does so.

Definition 3.3: An *intersection point* is the point at which the isosurface is estimated to cross the edge connecting two adjacent cell vertices that have different signs with respect to the threshold, usually using linear interpolation.

Such intersection points become vertices of one or more *topological polygons*, whose edges lie in the faces of the cell. We call them such because they specify the topology of the isosurface within the cell. Polygons joining more than three intersection points are normally nonplanar. □

Unless the underlying function that was sampled to produce the voxel values is known and can be resampled, the data are inherently incomplete, and the isosurface produced is at best a good approximation of the true one. Certain "smoothness" assumptions are implicit in all interpolation-based methods:

1. If two adjacent cell vertices both have values on the same side of the threshold, the isosurface is assumed not to pass between them, although the surface may actually pass between them any even number of times.

2. If two adjacent cell vertices have values on opposite sides of the threshold, the isosurface is assumed to pass between them just once, although the surface may actually pass between them any odd number of times.

The above two assumptions are made by linear interpolation. Of course, unsupported smoothness assumptions may be incorrect. As pointed out by Winget, isosurfaces in fluid dynamics problems can have discontinuities in the gradient, while the surface itself is continuous [Win88]. Furthermore, very small connected components may be entirely missed, for example, in the case where the surface lies entirely within one cell, and the voxel values of all cell vertices are positive. These problems are inherent to sampling, and only finer sampling or further knowledge of the function will deal with them correctly.

To quantify smoothness relative to the sampling interval in another way, we use the following definitions.

Definition 3.4: We say that a function is *locally linear* in a cell (relative to a given threshold) if there is a linear function (of three variables) whose values at the cell vertices are on the same side of the threshold as the sample values. That is, the positive and negative cell vertices can be separated by a plane.

Similarly, we say that a function is *locally quadratic* in a cell (relative to a given threshold) if there is a quadratic function that induces the same set of positive and negative vertices as do the sample values. □

When one considers the eight vertices defining a cell as being positive or negative, as defined above, there are 256 possible combinations of "positive" and "negative" vertex values. Using symmetries of the cube, these can be grouped into 22 cases [LV80, Sri81]. Eight of these cases are inverses of another case, in which the positive and negative values are reversed. Treating inverses as the same case, there are 14 cases, as shown in Figure 1, which we refer to as the *major cases*. Lorensen and Cline [LC87] give 15 cases because they do not consider the reflection of case 11 to be the same case. Our case numbering follows theirs with the exception of this omitted case.

It is easy to see that cases 1, 2, 5, 8 and 9 are locally linear; in these cases the black vertices are 1-connected, and so are the white. Case 11 also has only one connected component of each color, but cannot be locally linear, as a separating surface needs to slope one way as it cuts the top face, then slope a different way as it cuts the bottom face.

All of the cases except 13 are locally quadratic; an hyperboloid is always able to separate the colors. However, for case 13 it turns out that any quadratic function satisfies the identity that the sum of the values at the black vertices (of case 13) equals the sum of the values at the white vertices; but the former sum must be positive and the latter negative to agree with the sample values on sign (relative to the threshold).

3.3 Topological Ambiguities

We now examine the topologies that occur among the cases of Figure 1, and identify those that are ambiguous.

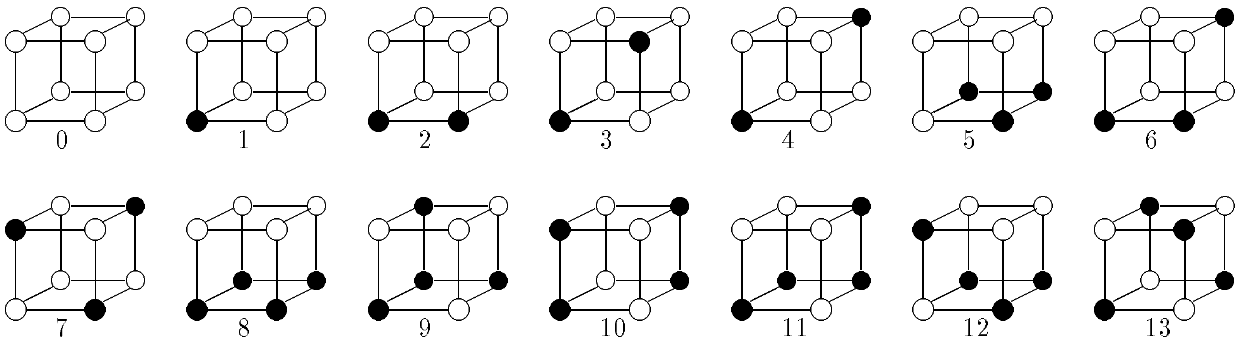


Figure 1: Fourteen topologically distinct major cases.

Definition 3.5: A *positive* cell edge is one that connects two positive cell vertices; a *negative* cell edge is one that connects two negative cell vertices.

A cell is *topologically ambiguous* if either the positive vertices of that cell cannot be connected together via *that cell's* positive edges into a single connected component, or the negative vertices cannot be connected by negative edges. In other words, either the set of positive cell vertices is not 1-connected, or the set of negative cell vertices is not 1-connected (see Definition 3.2).

An *ambiguous face* is a cell face that contains a diagonally opposite pair of positive vertices and a diagonally opposite pair of negative vertices (see Figure 2). □

Since the isosurface separates certain cell vertices, the connectedness of the vertices and the topology of the isosurface go hand in hand.

The principal manifestation of topological ambiguity occurs when two positive and two negative corner values of one cell face are diagonally opposite each other. The isosurface will then intersect all four edges. By examination of the possible cases, which are 3, 6, 7, 10, 12, 13, and their inverses, we observe that there is no configuration of the remaining four vertices of the cell that allows us both to connect the positive pair using positive edges and to connect the negative pair using negative edges. It is unclear from the cell vertex values alone whether to separate the positive vertices, separate the negative vertices, or even to have the contour lines cross and create four separate components (see Figure 2).

Consider the construction of the topological polygons, which are nonplanar in general. Each cell face has four edges, an even number of which will contain intersection points. If two edges of a face contain intersection points, then the only choice is to connect them with an edge, which will eventually belong to a topological polygon in each of the two cells that share that face. However, if a cell face has intersection points in all four of its edges, there are choices of how to connect up pairs to produce polygonal edges, as shown in Figure 2, which justifies the name *ambiguous face*.

Notice that case 4 in Figure 1 is also ambiguous as a *cell*, but has no ambiguous *face*. The ambiguity is whether the isosurface within this cell consists of two triangles or one "tube". Having no ambiguous face, there is no possibility of producing discontinuities. Therefore the treatment of this case is an independent issue, which we do not address in this paper. As seen in Table 1, this case rarely occurs. For simplicity, our implementation produces two triangles, thus separating the black vertices of case 4. Natarajan studies case 4 (as well as more complicated versions of the "long diagonal", which occur in cases 6, 10, and 12) under the assumption that the underlying function is trilinear within the cell [Nat91].

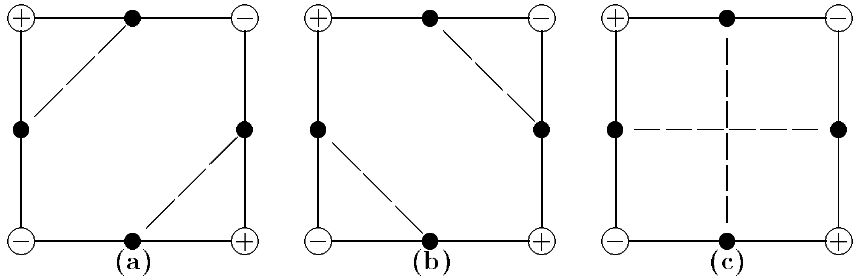


Figure 2: Possibilities for connecting four intersection points. Choice (a) puts the negative corners in the same connected component; choice (b) puts the positive corners in the same connected component; and choice (c) is ambiguous.

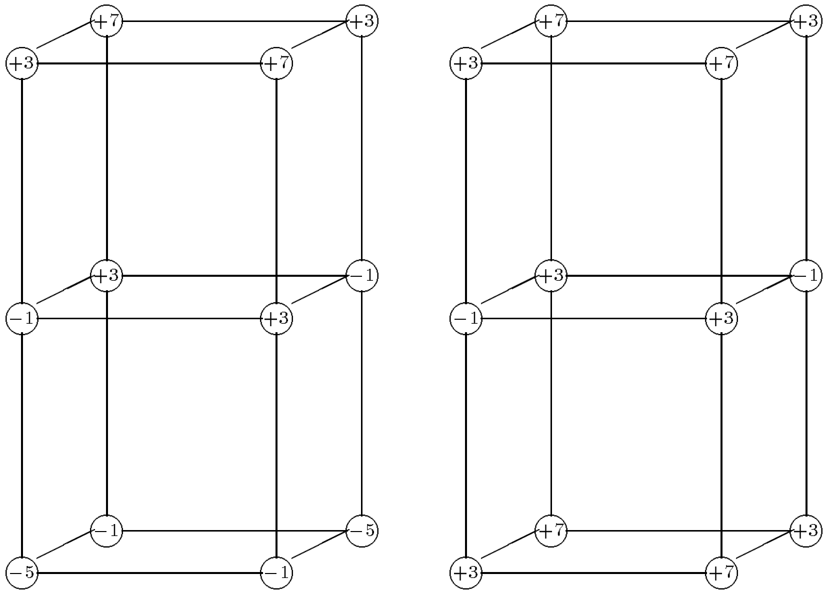


Figure 3: Center and Center-Lower Cell Values of F_1 and F_2

3.4 The Problem of Disambiguation

A number of approaches can be taken to pick a particular topology in ambiguous cases. If the underlying function is known, it may be possible to resample the data until the ambiguity is resolved [Blo88, KB89]. However, we are concerned with problems where resampling is impossible or impractical. This is frequently the case when the data originate from physical measurements, as in medical applications, or from complex computations, such as simulations and partial differential equations.

In cases of sampled data, it is often not possible to determine whether the choice of topology was correct: though the underlying function may be continuous, its exact form is generally not known. However, the polygonal representation of the isosurface *can* be checked for continuity. The surface is C^0 (positionally) continuous if and only if each edge is shared by exactly two polygons, except for edges that lie in the boundary of the entire volume, which must occur in exactly one polygon.

The topology is certainly incorrect if the polygonal isosurface is not continuous. However, even if the polygonal isosurface is continuous, its topology is not necessarily in agreement with the corresponding isosurface of the underlying function.

In any cell with an ambiguous face the underlying function cannot be locally linear (see Definition 3.4). To see why, observe that the intersection of any linear isosurface with a cell face is one straight line, which cannot induce an ambiguous configuration of positive and negative cell vertices. Thus we have focused on treating locally quadratic functions correctly, as this is essentially the best smoothness that can be hoped for. (Actually, case 13, in which *every* face is ambiguous, is not even locally quadratic.)

We have devised a number of functions that prove useful in testing the ability of isosurface algorithms to determine correct topology. Two of these functions are of particular interest:

$$F_1(x, y, z) = 4y + 4(x - z)^2 - 5 \tag{1}$$

$$F_2(x, y, z) = 4(y - 1)^2 + 2(x - z)^2 - 2(x + z - 3)^2 + 1 \tag{2}$$

Their actual isosurfaces for real numbers ranging from 0 to 3 in the three dimensions are shown in Figure 4.a. (In our examples, the threshold is zero unless otherwise specified.) We considered the lattices that result from sampling these functions at the integers 0 through 3, that is, a 4x4x4 array. Figure 3 shows the values of the center cell and the cell below the center for F_1 and F_2. Figure 5.a show the isosurfaces of each function for the center cell in isolation.

The salient point about these two functions is that the central cell of the 4x4x4 sample array *looks exactly the same for both functions*. However, one function represents a single connected surface, while the other represents two lobes of an hyperboloid. This pair of functions leads us to one of our principal findings:

Proposition 3.1: When the underlying function is not locally linear, it is impossible (in general) to determine the correct surface topology in a cell solely by examination of the voxel values at the vertices of that cell.

Proof: Functions F_1 and F_2 demonstrate the proposition. Sampled as discussed above, they cannot be distinguished in the central cell, yet have different topologies there. ∎

This proposition explains why anomalies are bound to occur in unfortunate cases in any approach which works only with values of one cell at a time.

3.5 Assurance of Continuity

This section addresses the problem of ensuring that the representation of the isosurface is continuous. It is known that a polygon mesh defines a continuous surface if and only if each edge occurs exactly twice, unless it is on the boundary of the entire data grid [Sri81, USH82]. Whereas correct topology is not achievable without knowledge of the underlying function, we shall see that continuity can be guaranteed by a variety of methods.

Definition 3.6: An edge of a polygon mesh is called *anomalous* if it is interior to the volume and does not occur exactly twice, or if it is on the volume boundary and does not occur exactly once. Such an edge is either *disconnected*, meaning that it occurs only once (leaving a hole, or void), or it is *multiple-branched*, meaning that it occurs too often. □

When cells are polygonized independently, some discipline is needed to assure that compatible decisions are made in the two cells that share an ambiguous face. A simple and practical principle may be employed to assure that this is always the case:

Proposition 3.2: (Facial Plane Principle) If the method of disambiguation for ambiguous faces employs only values in the plane of the face, and is invariant under rotations and reflections, then the isosurface as defined by topological polygons will be continuous.

Proof: Each nonboundary face is shared by two cells. If the face is ambiguous, by hypothesis the same edges are defined in that face for each cell. Thus these edges occur exactly twice in the polygon mesh, once in a topological polygon of each cell. Similarly an edge in an unambiguous face occurs once in a polygon of each cell. ∎

Aside from ambiguous faces, whenever an edge that lies in a cell face belongs to two polygons in one cell, there is a danger that multiple-branched edges may occur: that edge will normally occur also in the adjacent cell, and the edge is now in three or more polygons. This discontinuity may occur if many-sided topological polygons are tessellated by edges (chords) within a cell face rather than through the interior of the cell. A special case of in-face chords occurs when facial polygons are created, which has been done in some tables in an attempt to avoid holes in the surface (see Section 4.1). If facial polygons from adjacent cells coincide, a membrane-like appearance results, as can be seen in Figure 4, (b, right). See Section 4.8 for further discussion.

4 Approaches to Disambiguation of Sampled Data

Methods for disambiguation can be broadly classified according to two properties.

1. Is the data treated as *boolean* or *metric*?

   Boolean methods consider only whether the data is above or below the threshold. *Metric* methods also take into account how far the data is above or below the threshold. (Here we are classifying only how the connectivity is decided, not how the surface is represented.)

2. Is the region that affects the disambiguation decision *simple* or *extended*?

   Simple methods consider only the sample values in the cell being processed, which contains the ambiguous face. *Extended* methods consider additional sample values, in the original data, but outside the cell.

We have investigated numerous approaches to disambiguation that cover the combinations of these properties. They are briefly summarized here and described in more detail in subsequent sections. Computational experience is reported in Section 5.2.

Simple Boolean

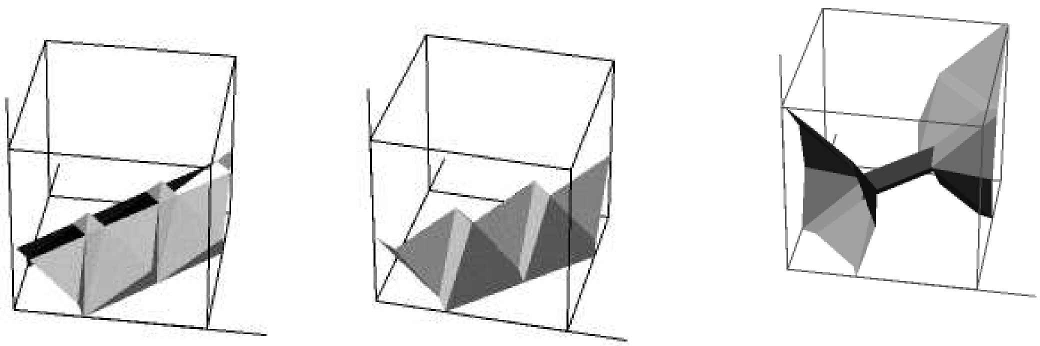
This category includes the simplest policies, such as "always connect the positive diagonal," which is implicit in early boundary tracking methods [AFH81].

The *marching cubes* algorithm [LC87, CLL$^+$88, GN89], uses the signs of the eight cell vertices (relative to the threshold) to index a table of 15 cases containing a polygonization of the isosurface. The table covers cells with up to 4 positive vertices; a cell with 5 or more positive vertices is inverted before lookup. This method may lead to discontinuities because an ambiguous face can be resolved differently, according to the values in the opposite face of the cell.

A table of 22-23 cases may be constructed in such a way that the resolution of an ambiguous face depends only on that face. This approach guarantees continuity. Several variants have appeared. Some were designed to correct the problem of discontinuities in marching cubes.

Since triangular faces cannot be ambiguous, another method to disambiguate implicitly is to decompose each cell into tetrahedra, and interpolate linearly on each tetrahedral edge [KDK86, Blo88, DK91]. The

a. Isosurface 0 of quadratic functions F_1 (left) and F_2 (right).



b. Various incorrect topologies. Marching cubes on F_1 (left);
Facial Average, Bilinear, and UA-R2 on F_1 (center); UA-R1 on F_2 (right).

c. Correct topologies. Gradient heuristic methods are correct on both.
Others are correct on only one: UA-R1 (F_1 only);
Marching Cubes, Facial Average, Bilinear, and UA-R2 (F_2 only).

Figure 4: Renditions of $F_1(x,y,z) = 4y+4(x-z)^2-5$, and $F_2(x,y,z) = 4(y-1)^2+2(x-z)^2-2(x+z-3)^2+1$, over the range [0,3] in X, Y, Z (a 4x4x4 sample volume).

a. Isosurface 0 of quadratic functions F_1 (left) and F_2 (right).

b. Polygonized.

Figure 5: Center cell comparisons for F_1 and F_2. Simple methods choose the same topology for both functions. UA-R1 chooses (b, left); Marching cubes, UA-R2, Facial Average, and Bilinear choose (b, right). Gradient heuristic methods correctly choose (b, left) for F_1 and (b, right) for F_2.

---

decomposition is not isotropic, and requires an arbitrary choice. The resulting topology depends on this choice.

Extended Boolean

We are not aware of any work in this category.

Simple Metric

Wyvill *et al.* introduced the first isosurface method that constructed beveled surfaces [WMW86]. They used the *Facial average value* to choose which corners to connect in an ambiguous face. This is the average value of the cell vertices at the corners of that face.

A newer proposal is the *Bilinear model*. A trilinear function is often used for interpolating sampled data. The isosurface of the trilinear function that fits the cell corners may be used for disambiguation [Nat91]. In a cell face the model simplifies to bilinear [NH91].

Another possibility that has never been implemented is *Ambiguous representation*: Ambiguities can be rendered *ambiguously* as a compromise between topologies. The main idea is to partition an ambiguous face into four regions that meet at a single point, and are alternately positive and negative.

Extended Metric

A new class of methods, *Gradient consistency heuristics*, is introduced here. In these methods, the choice among ambiguous subcases is made using estimated gradient information. Since we estimate gradients using the central difference method, they contain information about function values outside the cell. The heuristic is to choose a topology that is "most consistent" with the gradient information, in some sense. We have implemented two such heuristics, which we call the *center-pointing gradient* method, and the *quadratic fit* method.

Tricubic interpolation is a more expensive interpolation method which also considers values outside the cell in generating the isosurface.

4.1 Simple Boolean

The first approach examined was the use of a single table. We studied several variations of "marching cubes" tables, which we call "major case" tables. These variants determine topology based only on the signs of the eight corner values relative to the threshold, and pick one topology for each case. The table entry for each case specifies the edge intersections that should be connected to create triangles representing the isosurface within the cell [LC87, CLL+88]. The marching-cubes table lookup method is devised for speed, at the occasional expense of correct topology.

Part (b, left) of Figure 4 shows how the original table [LC87, CLL+88] polygonizes function F_1. This table polygonizes cells with 5 or more positive vertices by using the inverse major case; e.g., "+++++-+-" is treated the same as "-----+-+". Note that the method leaves a discontinuous hole in function F_1, a problem pointed out by Dürst [Dür88]. This occurs because, when two cell vertices diagonally opposite across a face are of one sign and the other six cell vertices are of the other, the table always chooses to treat the two as two disconnected components and separates them using two triangles. The center cell of F_1 contains two diagonal negatives on the lower face and the rest positives, whereas the cell beneath has two diagonal positives on the upper face and the rest negatives, hence a hole. This approach does correctly separate the two lobes of the hyperboloid in function F_2, where the center cell is as in F_1 and the lower cell is the same case upside down. The picture is similar to Part (c, right) of Figure 4.

Udupa and Ajjanagadde have described three simple boolean connectivity policies (their terminology differs) [UA90]. Positive values are considered to represent the "objects". (In the cited paper "$R_u = R_1$" corresponds to our label "UA-R1", etc.; the "UA" gives the authors' initials.)

UA-R1 Always connect negative diagonals (often associated with corrected versions of marching cubes [LC87, Bak89, Kal91]).

UA-R2 Always connect positive diagonals (often associated with cuberilles [AFH81, HU83, CHRU85]).

UA-R3 Connect negative diagonals if the face is parallel to the xy-plane; otherwise, connect positive diagonals.[1]

Policy UA-R3 originates in the cited paper. Typically, CT volumes have a greater spacing between sample values in the z direction. However, the paper does not discuss pros and cons of the various alternatives in terms of representing the underlying physiology. Rather, they motivate UA-R3 in terms of computational efficiency for boundary tracking.

A proposal to correct the discontinuity problems of marching cubes was made independently by several researchers [Bak89, Kal91, Nat91, NH91]. It treats cells with 5 or more positive vertices independently of their inverses, by always connecting the negative corners of any ambiguous face. Therefore, it corresponds

[1] Of course, one can choose any one of the coordinate planes to be treated differently, but in practice Udupa and Ajjanagadde chose the xy-plane for their datasets.

to the UA-R1 connectivity policy. This method also may be implemented by a single major case table, this time with 23 cases. It chooses the correct topology for function F_1, as shown in Figure 4, part (c, left).

A problem with the UA-R1 policy is illustrated by function F_2. Both the center cell and the one below it are the same case; the lower cell is an upside down boolean version of the center cell. The result is an incorrect "tube" connecting the two lobes of the hyperboloid, as shown in Figure 4, part (b, right).

If the positive and negative regions of F_1 and F_2 were reversed, the table based on UA-R1 would correctly separate the lobes of the hyperboloid in F_2 (Figure 4, part (c, right)). However, for F_1 this method would produce the incorrect result seen in Figure 4, part (b, center). This indicates the problem with using different approaches for inverse cases, and motivates criterion 4 in Section 2.

Note that policy UA-R2 produces the same results on the original functions that policy UA-R1 produces on the inverted functions. On this example with a single ambiguous face, policy UA-R3 will correspond to either UA-R1 or UA-R2, depending on the choice of coordinate axes. In general, whether a decision by UA-R3 agrees with UA-R1 or with UA-R2 is governed by the orientation of the ambiguous face.

4.2 Simple Metric

Methods in this category include the facial average method, bilinear models, and ambiguous representation.

Facial Average Values

The facial average value method is employed by Wyvill *et al.* [WMW86]. The *facial average value* is the average of the four corner values of an ambiguous face. It is also equal to the center value of a bilinear interpolation across the cell face.

The facial average method *does* differentiate between ambiguous topologies, but not in a particularly satisfactory manner. Because the choice is based only on the face values, a continuous isosurface is guaranteed by the facial plane principle, Proposition 3.2. However, the method will sometimes choose incorrectly on underlying functions as simple as quadratics.

Part (b, center) of Figure 4 shows how the facial average method interprets the location of the isosurface for F_1. Note that, although the surface is continuous overall, the topology of the center cell is incorrect. In particular, the facial average value (the estimated center value) is positive, while the actual function value there is negative. Facial averaging does correctly leave the two lobes of the hyperboloid in function F_2 separate (Figure 4, Part (c, right)).

The treatment of the center cell is shown in isolation in Part (b, left) of Figure 5. Notice that the choices of topology in the center and lower center cells of function F_1 are incorrect, but consistent with each other. The positive corner values of the shared face are further from the threshold than the negative corner values, so the method interprets the negative voxels in the shared face as topologically separate, although they are part of one component in the actual function.

In the lower center cell, these edges in the shared face are part of the same topological polygon. The result is a surface with a jagged saw-tooth look.

All of the metric methods have been implemented using a supplementary "subcase table" when the major case is ambiguous, as described in Section 4.7. With this implementation, the facial average method has performance speed close to that of the simple boolean methods.

Bilinear Model

Trilinear interpolation among the eight cell vertices is a method often used to estimate the behavior of the sampled data between sample points, when extracting isosurfaces [CLL+88], for direct volume rendering [Lev88, UK88], and for producing more sample points from those already provided [HB86]. It is attractive because it is simple to calculate.

The trilinear model of the cell may be used for connectivity decisions, as described by Natarajan [Nat91]. Nielson and Hamann considered bilinear models in each face [NH91]. In each cell face, the trilinear function reduces to a bilinear function that depends only on the corners of that face. Thus, these methods obey the facial plane principle of Proposition 3.2. It follows that C^0 (positional) continuity is achieved in the shared faces between neighboring cells, for regular grids.

The iso-contour lines of the bilinear function form an hyperbola. If the product of the positive values (based on threshold 0) at the corners of the ambiguous face exceeds the product of the negatives, the positives are joined by this hyperbola; if the opposite is true, the negatives are joined. The model is used only for disambiguation decisions. Polygonization is based on linear interpolations, as with other methods.

Like the facial average method, this method will sometimes choose incorrectly on underlying functions as simple as quadratics. Part (b, center) of Figure 4 shows the incorrect surface produced by the bilinear model of F_1. The explanation is similar to that given for the facial average method. Part (c, right) shows its treatment of F_2, which is correct.

The trilinear model can also be used to subdivide the cells, possibly producing a smoother isosurface than that produced by simply approximating the surface with a table of polygons [CLL+88]. However, this has no effect on the topology, and F_1 will still emerge with the saw-tooth image.

Ambiguous Representation

An approach that was briefly explored is to render ambiguities *ambiguously*. It is possible to produce a single ambiguous representation (see part (c) of Figure 2) for each of the base cases and use a single major case table (as in Section 4.1) without producing grossly incorrect topologies. This "noncommittal" method deserves further exploration, and may be useful for degenerate cases.

4.3 Tricubic Interpolation

We now investigate extended metric methods. These methods use sample values external to the cell, to attempt to find the topology more accurately.

A more expensive approach is to fit a tricubic polynomial to the cell, using the 4x4x4 region of voxels surrounding the cell to specify the function. Tricubic interpolation has the advantage of taking into account the behavior of the neighborhood beyond the cell of interest.

The tricubic function used is determined by the 64 voxel values surrounding the cell of interest. The function is the 3-dimensional volumetric version of the Catmull-Rom spline [CR74, FDFH90]. Details are given in Appendix A. This method produces a tricubic function within the cell that is C^1 continuous with tricubic functions in neighboring cells at faces.

Parts (a) of Figure 4 show the surfaces produced when the resolution is increased by a factor of 5, the volume is filled in by tricubic interpolation, and regular isosurface software is used on the results, which contain no ambiguous cells.

Tricubic interpolation is the first method that we have discussed that correctly picks the cell topology for both functions F_1 and F_2, as shown in Figure 5.a, where the center cell is isolated. Also, it can be proven to yield a continuous isosurface, using Equation 18 and the facial plane principle, Proposition 3.2.

However, the expense of tricubic interpolation makes it impractical as merely a disambiguation method. (It may still be of interest as a higher order method for the entire volume, but that is beyond the scope of this paper.) One computation of $F(x, y, z)$ requires over a thousand arithmetic operations, and the method just described requires over 100 such computations per cell! The decisive difference from other methods is that it must be applied to *all* cells, not just the ambiguous cells, because cubic interpolation is not necessarily consistent with the smoothness assumptions stated in Section 3.2. That is, discontinuities will result from performing linear interpolation in a nonambiguous cell and cubic interpolation in an adjacent ambiguous cell.

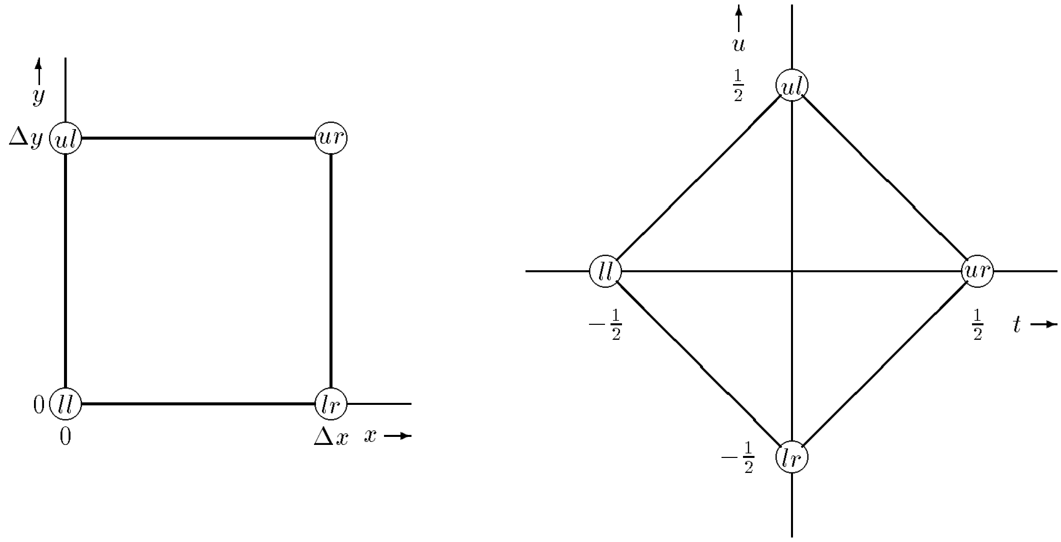


Figure 6: Change of coordinate system for gradient heuristic methods.

Later sections describe significantly more efficient methods that use neighborhood information in the form of computed gradients to disambiguate.

4.4 Gradient Consistency Heuristics

Gradient consistency heuristics use the estimated gradients at the cell vertices to pick topologies in ambiguous cases, in this way using information from beyond the cell. The gradient direction is normal to the isosurface, and its magnitude indicates how rapidly the function is changing. Gradient consistency heuristics encompass a class of methods that use gradient information to disambiguate; we shall describe two methods that we have investigated and implemented, called the *center-pointing gradient* method, and the *quadratic fit* method. In some cases no additional computational cost would be incurred to calculate gradients, because they might also be needed for the shading model. In the interests of modularity, our implementation calculates gradient components separately, as needed, for disambiguation.

The gradient gives an indication of the behavior of the function across the interior of the face. For our heuristics, only the component of the gradient in the plane of the face is needed; e.g., if the face is parallel to the x-y plane (z is constant) only the x and y components of the gradient are used. No direct gradient information is given in the data, so wherever a computation using gradients is mentioned, it should be understood that the gradients must be estimated from the data samples. Moreover, we nondimensionalize by implicitly scaling $\Delta x = \Delta y = 1$ before computing gradients. The computed gradients use the 8 points in the plane of the face that are 1-connected neighbors of the corners of the face.

Recall that no linear function of two variables can fit an ambiguous face (Section 3.2); thus a quadratic is in a sense the simplest possibility. Both of our methods model the function in the cell face as a bivariate quadratic that fits the corner values of the face exactly, and fits the (nondimensionalized, computed) gradients at the corners with minimum squared error. (It is easy to see that this function is also the quadratic that fits the corner values of the face exactly and fits the 1-connected neighbors with minimum squared error.) Fitting the corner values of an ambiguous face exactly ensures that the same edges have intersection points in the quadratic model as in the data.

Let the corners of the face in the local (x, y) coordinate system be $(0,0)$, $(\Delta x, 0)$, $(0, \Delta y)$, and $(\Delta x, \Delta y)$; let us label them as ll, lr, ul, and ur, respectively. We use the nomenclature *even* to refer to points ll and ur, and *odd* to refer to points lr and ul. We denote the x-component of the estimated gradient by $\bigtriangledown f_x$ and the y-component by $\bigtriangledown f_y$.

Our gradient heuristics are most easily explained in a transformed coordinate system (t, u), as shown in Figure 6, where

$$\begin{aligned} x &= (t - u + \tfrac{1}{2})\Delta x \\ y &= (t + u + \tfrac{1}{2})\Delta y \end{aligned} \tag{3}$$

With some abuse of notation, we use $f(t, u)$ to denote $f(x(t, u), y(t, u))$. Gradients in the (t, u) system are given by

$$\begin{pmatrix} \bigtriangledown f_t \\ \bigtriangledown f_u \end{pmatrix} = \begin{pmatrix} \Delta x & \Delta y \\ -\Delta x & \Delta y \end{pmatrix} \begin{pmatrix} \bigtriangledown f_x \\ \bigtriangledown f_y \end{pmatrix} \tag{4}$$

Let us use the notation $\bigtriangledown f_{x,ll}$ and $\bigtriangledown f_{y,ll}$ for the estimated gradient components in the x and y directions at point ll, with corresponding notations for $\bigtriangledown f_{t,ll}$, $\bigtriangledown f_{u,ll}$, and for other corners. Notice also that $\bigtriangledown f_t$ and $\bigtriangledown f_u$ are independent of Δx and Δy.

A quadratic function of two variables has 6 coefficients. We shall represent it in the (t, u) system described by Equations 4 and 4.4 (see Figure 6) as:

$$Q(t, u) = c + b_1 t + b_2 u + a_{11} t^2 + 2a_{12} tu + a_{22} u^2 \tag{5}$$

For notation, let b denote the column vector of (b_1, b_2), let b^T denote the corresponding row vector, and let A be the 2x2 symmetric matrix of elements a_{ij}.

Since we have four sample values and 8 computed gradient components to fit, we cannot expect to satisfy all constraints with 6 degrees of freedom. As mentioned, we shall choose the coefficients to fit the sample values exactly; Thus we require $Q(-\frac{1}{2}, 0) = f_{ll}$, $Q(0, -\frac{1}{2}) = f_{lr}$, etc. We easily get the following equations from the exact fit requirements:

$$\begin{aligned} f_{ll} + f_{lr} + f_{ul} + f_{ur} &= 4c + \tfrac{1}{2}(a_{11} + a_{22}) \\ f_{ur} - f_{ll} &= b_1 \\ f_{ul} - f_{lr} &= b_2 \\ f_{ll} - f_{lr} - f_{ul} + f_{ur} &= \tfrac{1}{2}(a_{11} - a_{22}) \end{aligned} \tag{6}$$

With these equations we can eliminate all unknown coefficients from the expression for $Q(t, u)$ except for a_{12} and c. In particular,

$$\begin{aligned} a_{11} &= 2(f_{ll} + f_{ur}) - 4c \\ a_{22} &= 2(f_{lr} + f_{ul}) - 4c \end{aligned} \tag{7}$$

Then we require the gradient components fit with least square error. That is, let

$$\begin{aligned} E(t, u) &= (b_1 + 2a_{11} t + 2a_{12} u - \bigtriangledown f_t(t, u))^2 \\ &+ (b_2 + 2a_{22} u + 2a_{12} t - \bigtriangledown f_u(t, u))^2 \end{aligned} \tag{8}$$

which defines the squared error at each corner (t, u). To minimize the sum over four corners, we must satisfy the constraints:

$$\left(\frac{\partial E(-\frac{1}{2},0)}{\partial a_{11}}+\frac{\partial E(0,-\frac{1}{2})}{\partial a_{11}}+\frac{\partial E(0,\frac{1}{2})}{\partial a_{11}}+\frac{\partial E(\frac{1}{2},0)}{\partial a_{11}}\right)\frac{da_{11}}{dc} \quad + \tag{9}$$
$$\left(\frac{\partial E(-\frac{1}{2},0)}{\partial a_{22}}+\frac{\partial E(0,-\frac{1}{2})}{\partial a_{22}}+\frac{\partial E(0,\frac{1}{2})}{\partial a_{22}}+\frac{\partial E(\frac{1}{2},0)}{\partial a_{22}}\right)\frac{da_{22}}{dc} \quad = \quad 0$$
$$\frac{\partial E(-\frac{1}{2},0)}{\partial a_{12}}+\frac{\partial E(0,-\frac{1}{2})}{\partial a_{12}}+\frac{\partial E(0,\frac{1}{2})}{\partial a_{12}}+\frac{\partial E(\frac{1}{2},0)}{\partial a_{12}} \quad = \quad 0$$

The symmetries cause b_1 and b_2 to drop out, leading to a reasonably simple solution:

$$\begin{aligned} c &= \tfrac{1}{4}(f_{ll}+f_{lr}+f_{ul}+f_{ur}) \\ &\quad +\tfrac{1}{16}(\nabla f_{t,ll}+\nabla f_{u,lr}-\nabla f_{u,ul}-\nabla f_{t,ur}) \\ a_{12} &= \tfrac{1}{4}(-\nabla f_{u,ll}-\nabla f_{t,lr}+\nabla f_{t,ul}+\nabla f_{u,ur}) \end{aligned} \tag{10}$$

4.5 Center-Pointing Gradient Method

We observe that $f_C = Q(0,0)$, the estimate of the function at the center of the ambiguous face, is simply c in Eq. 11. In this equation, we see that f_C can be thought of as providing a "correction term" to the facial average method. Based on this improved estimate, the center-pointing gradient method disambiguates by using the subcase table described in Section 4.7 (which is also used by the facial average method). That is, if f_C is above the threshold, this method decides to connect the positive pair of corners, otherwise the negative pair.

This approach successfully finds the correct topology for both F_1 and F_2, as shown in Parts (d) of Figure 4 and 5. However, it *can* fail for quadratic functions, motivating the *quadratic fit* refinement discussed next.

4.6 Quadratic Fit Method

The quadratic fit method is a more sophisticated – and more expensive – gradient consistency heuristic. Assuming that we believe a quadratic function provides a satisfactory local representation of the underlying function in an ambiguous face and its immediate neighborhood, then the curves along which that quadratic function is zero describe a conic section in the plane of the face, and the conic section determines the topology in the face. This conic section is usually an hyperbola, but might be an ellipse, parabola, or, in a very degenerate case, a pair of straight lines. Based on the topology induced by the conic section in each ambiguous cell face, the quadratic fit method disambiguates by using the subcase table described in Section 4.7 (which is also used by the facial average and center-pointing gradient methods).

A bilinear function, considered earlier in Section 4.2, is really a special case of a bivariate quadratic. Besides greater generality, considering a general quadratic has the advantage that it is insensitive to a change of coordinates.

Consider the quadratic function $Q(t,u)$ of Eq. 5, where the coefficients given in Section 4.4. We assume the threshold is 0 in this discussion. If the curve implicitly defined by $Q(t,u) = 0$ describes a *convex* conic section (an ellipse, parabola, or pair of parallel lines), the facial center (f_C, used by the center-pointing gradient method) must have the same sign as the diagonal pair of corners that are connected within the conic section. However, in the more common case of an hyperbola, it is possible that the sign of the facial center is opposite that of the connected diagonal corners. Geometrically, one lobe of the hyperbola "cuts off" the facial center, so that it does not lie between the two lobes.

In hyperbolic cases, the sign of the facial center value (as estimated by the quadratic fit or center-pointing gradients) is not necessarily indicative of which diagonal pair of corners to connect. However, the sign of the *saddle point* does indicate which diagonal pair to connect, as the saddle point is always between the lobes of the hyperbola. This follows from the fact that the major axis of the hyperbola, which runs between the two lobes, goes through the cell face, and $Q(t, u)$ has the same sign everywhere on the major axis as it does at the saddle point.

Even in an hyperbolic case, it may be unnecessary to compute the saddle point. For example, suppose $Q(0, 0)$ has the same sign as $Q(-\frac{1}{2}, 0)$ and $Q(\frac{1}{2}, 0)$. Then if $Q(t, 0)$ has no roots in the interval $(-\frac{1}{2}, \frac{1}{2})$, the quadratic fit connects the corners on the t axis, ll and ur. If $Q(t, 0)$ does have roots in this interval, then the saddle point must be consulted. Similarly, when $Q(0, 0)$ has the same sign as $Q(0, -\frac{1}{2})$ and $Q(0, \frac{1}{2})$, then $Q(0, u)$ is tested for roots.

The root test just described fails only when A is nonsingular $Q(t, u)$ has a unique *saddle point* at

$$\begin{pmatrix} t_S \\ u_S \end{pmatrix} = -\tfrac{1}{2}A^{-1}b \tag{11}$$

This point may be outside of the cell face. Nevertheless, the remarks above concerning the major axis show why the quadratic connects the cell corners that agree in sign with the saddle point.

It is most instructive to consider the change in value between the facial center and the stationary point:

$$Q(t_S, u_S) - Q(0, 0) = -\tfrac{1}{4}b^T A^{-1} b \tag{12}$$

where the values of b and A were given in Section 4.4. Just as the center-pointing gradient method could be viewed as giving a correction term for the facial average value, so can the quadratic fit method be viewed as providing a correction term for the center-pointing gradient.

The sign of $Q(t_S, u_S)$ determines which diagonal pair to connect. However, if this value is zero, then the conic section must be a pair of intersecting lines, a degenerate hyperbola, and we do not know which pair to connect. This occurrence is very unlikely, as Q must factor into the product of two linear forms; nevertheless, it points up the fact that any small value of $Q(t_S, u_S)$ computed from noisy data is unreliable for determination of topology.

As would be expected, the quadratic fit method always chooses topology correctly when the underlying function is indeed quadratic, as are F_1 and F_2 in our running example. The renditions are shown in parts (d) of Figure 4, and 5. Of course, it can choose incorrectly if the underlying function is not quadratic.

4.7 Subcase Tables

Several of the methods described earlier provide a procedure to decide, in each ambiguous face of an ambiguous cell, whether to connect the positive pair of corners, or the negative pair, in the sense of section 3.3. This section describes how our implementation translates these decisions into an actual polygonization for the cell. The idea is to supplement the major table, which is sufficient for the unambiguous cases, by a subcase table to handle the ambiguous cases in greater detail.

The signs of the eight cell vertex values are used as an index into a major case table; these indices range from 0 to 255. If the major case is unambiguous, the major table simply contains a list of edges for the polygons representing the isosurface in the cell.

However, if the major case is ambiguous, the major table contains a list of the ambiguous faces, where a decision is needed whether to connect the positive corners or the negative corners, and an index into a second table that describes the subcases of this ambiguous case. Each subcase specifies the polygonization desired for a different possible topology for the case.

Figure 7 shows the subcase polygonizations for ambiguous major case 12 of Figure 1. The major table entry for this case contains a list specifying that the front and left faces of the cell are ambiguous. The four possible decisions are encoded as follows:

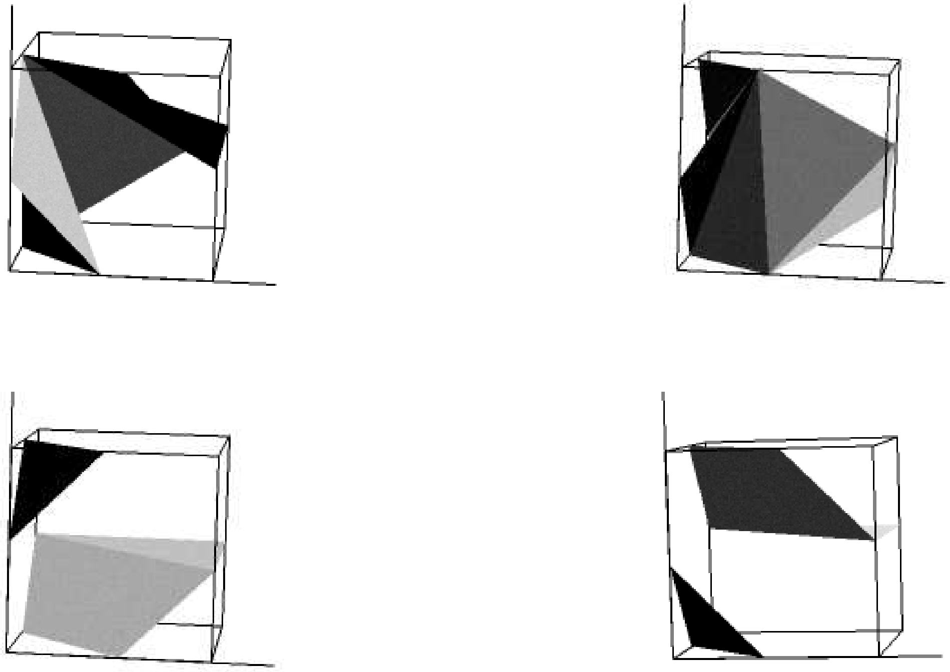


Figure 7: Subcase Representations for Ambiguous Case 12.

00 means connect the negatives in both faces (as shown in lower left picture);

10 means connect the positives in the front face and the negatives in the left face (upper left);

01 means connect the negatives in the front face and the positives in the left face (upper right);

11 means connect the positives in both faces (lower right).

This two-bit integer is treated as an offset from the index that is also in the major table, to specify a precise subcase table entry. The subcase table entry specifies the polygonization.

The subcase table was created by manually designing the polygons that should represent the ambiguous cases for the major cases. A program then automatically generated the table for all 128 ambiguous cases that can occur, by considering the mapping from the major cases to these cases.

4.8 From Topological Polygons to Surface Patches

As shown in Section 4.7, a topological polygon can have as many as 12 vertices. It remains to define the surface patch of which it is the boundary. The usual way to do so is to subdivide such polygons into simpler figures, such as triangles, which define a planar patch, and quadrilaterals, which define an hyperboloid patch. For lack of a better word, we call this process "tessellation". Tessellation is a topic in its own right. This

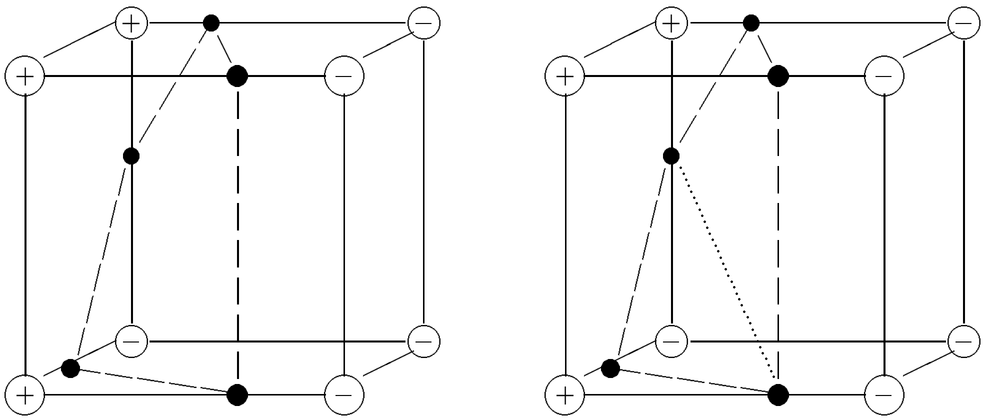


Figure 8: Topological and “Tessellated” Polygons

paper, which is primarily concerned with *deciding upon* the topological polygons, will not treat this topic in depth.

Our implementation tessellates by introducing chords through the cell interior. Figure 8 shows a topological polygon defining the isosurface for a cell with three positive vertices forming one connected component, and five negative vertices forming another. The topological polygon is pentagonal, with five edges in the faces of the cell; it can be further “tessellated” for rendering into a triangle and a quadrilateral, as shown. The quadrilateral could also be divided into two triangles by another interior chord.

Kalvin points out that it is important that the final surfaces not intersect themselves or each other, so that they describe physically realizable objects [Kal91]. We shall call this the *non-intersection property*. He proves that the “cuberille method” of disambiguation (always connect positive diagonals in ambiguous faces) has this property.

There are certain topological polygons, involving multiple ambiguous faces, that cannot be triangulated without introducing an in-face chord. (Nielson and Hamann have spelled out these cases in detail [NH91].) As mentioned earlier, in-face chords are undesirable because they create discontinuities. Our implementation uses a combination of triangles and *quadrilaterals* to ensure that all chords are interior to the cell. This ensures continuity, but does not ensure the non-intersection property. As the quadrilaterals are generally hyperboloidal, further analysis of the surface (such as segmentation, area and volume) is difficult. (We thank one of the referees for pointing out these issues.)

For completeness, we briefly describe a modification to our implementation that is sufficient to achieve the non-intersection property. When necessary, a topological polygon is supplemented with an *interior vertex*, which is the centroid, or “center of gravity”, of the vertices of its topological polygon. In this event, the tessellation consists of rays from the centroid to the vertices of the topological polygon.

1. Let us call an ambiguous face *overworked* if both of its topological edges belong to the same topological polygon. Interior vertices are needed only if a cell has two or more overworked faces (by an easy case analysis). In this event (which we believe to be quite pathological), each topological polygon involved with an overworked face requires an interior vertex.

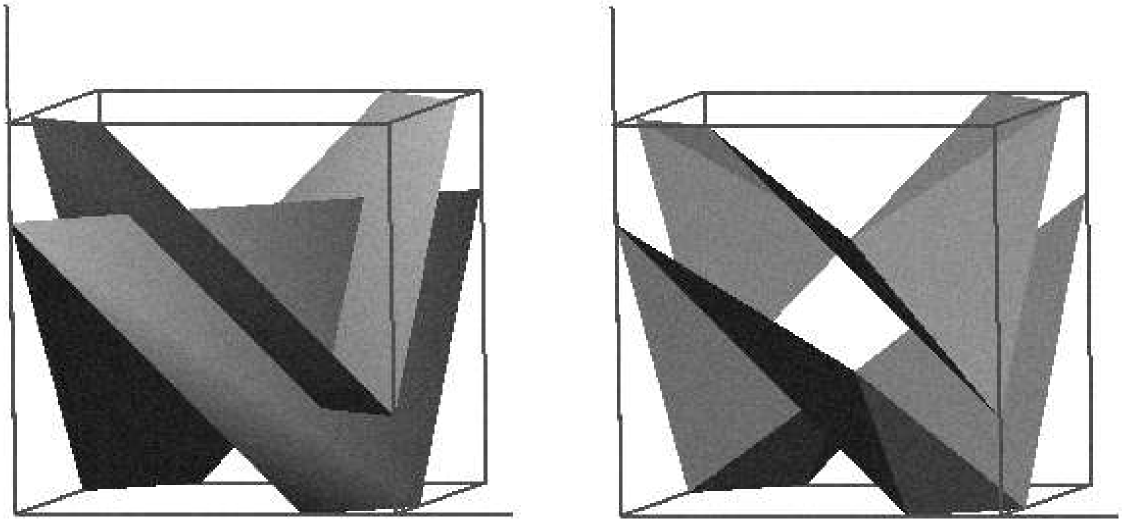


Figure 9: Intersecting surfaces within a cell can result from triangulation of certain topological polygons (left). Introducing centroids as additional vertices eliminates the problem (right).

2. There is one situation in which two topological polygons involve overworked faces, as shown in Figure 9. This is a subcase of our "case 13". As shown on the left of that figure, chordal tessellation can produce intersecting surfaces. On the right, centroids serve as interior vertices, and the surfaces are disjoint.

   To prove that the resulting surfaces must be disjoint, we observe that the centroid of the lower polygon is at most 1/3 from the bottom face of the cell, while the upper centroid is at most 1/3 from the top face (where cell sides are unit length). Also, each centroid is at least 1/3 from each side face. The conclusion follows by analytical geometry.

3. In other cases, of which there are several, one topological polygon is incident upon two or more overworked faces. The only question is whether the rays from its centroid produce a self-intersecting surface. So, imagine a sphere outside the cell whose center is the centroid. Project the edges of the topological polygon radially onto this sphere. Clearly, the surface is self-intersecting only if some of the resulting spherical arc segments intersect. But this is impossible, because then the corresponding topological edges would also intersect.

Nielson and Hamann described a different way to define interior vertices, which is sufficient for their purposes, but which does not handle the case shown in Figure 9 [NH91]. Non-intersection is not mentioned, but an argument similar to that of case 3 above shows it to hold.

5 Results on Scientific Data

5.1 Connectivity Comparison on Medical Data

An earlier section discussed behavior of disambiguation methods on artificial functions, because in those cases we *know* the correct behavior. With data from an application such as medical imaging, a clear-cut determination of the correct behavior in all cases is not possible. One can only look at examples and form a judgement about whether the underlying physiology is correctly represented. This section presents some examples that are typical of our own observations.

As stated earlier, disambiguation is essentially the process of deciding which pair of diagonal nodes to consider connected in an ambiguous face. Recall the definitions of the simple boolean policies UA-R1, UA-R2, and UA-R3 from Section 4.1. From the earlier discussion, we know that none of these methods can be correct on both F_1 and F_2. Now we consider their correctness on medical image volumes, particularly in comparison to the gradient heuristics.

We would expect UA-R1 to err on the side of disconnecting thin objects that should be connected. Rusinek *et al.* have reported difficulty in visualizing thin bones with Kalvin's "Alligator" method, which uses UA-R1 connectivity [RNMK91]. Their experiment used a CT-scan of a dry skull. On similar data, we have also noticed that small tunnels sometimes appear in thin bones using UA-R1 connectivity, but they disappear using others. However, without access to the original specimen we cannot be sure whether these are *pseudo-foramena* (reconstruction artifacts), or actual *foramena* (small holes in the bone through which nerves and blood vessels often pass).

On the other hand, UA-R2 would tend to connect distinct objects that are close to each, and to connect nearby parts of an object that have no actual connection. Connectivity UA-R3, being a mixture, would have one tendency or the other, depending on the orientation of the ambiguous face.

Figure 10 shows isosurface images based on magnetic resonance imaging (MRI) of a brain. The dataset was distributed by University of North Carolina, Chapel Hill. The threshold was 650.5. The top image was based on connectivity UA-R2; UA-R3 was very similar. The bottom image was based on the Quadratic Fit method; the Center-Pointing Gradient method was very similar.

In the top image, separate convolutions are connected by "bridges". Arrows show sites of numerous bridges in the *central sulcus* (A, B), *pre-central sulcus* (C, D), and elsewhere (E, F). The sites (A–D) are in major *sulci* (deep fissures between convolutions). At these sites, UA-R2 and UA-R3 created 11–12 incorrect bridges, the facial average method created 4, while UA-R1 and the gradient heuristic methods created none.

In the narrower fissures at (E, F) all methods that we studied created 5–7 incorrect bridges. These counts are approximate, based on close-ups and cut-aways (see Figure 11); bridges may also be found at unlabeled sites.

These data indicate that, when ambiguities do occur, the methods that use the most information tend to be the most accurate. The simple boolean methods UA-R1–3, using only one bit, fare the worst. Notice that simple boolean methods cannot be neutral with respect to sign inversion. The Facial Average method, using information in the cell face only, is intermediate. The Gradient Heuristics, using information beyond the cell face, fare the best. Of course, the large majority of cells are not ambiguous and all methods produce the same surface elements in such cells.

5.2 Comparative Performance Measurements

The table-driven methods described above were evaluated on sampled data volumes derived from CT-scans, a finite element analysis in computational fluid dynamics (CFD), and a molecular data set. Their performance was compared with respect to number of discontinuities, number of polygons produced, and CPU time.

The CT-scan volumes and the molecular data set consisted of scalar data arranged in a three-dimensional rectilinear lattice. The CFD study was on a geometrically warped grid, but each cell had eight corners. Statistics here refer to three sample volumes:

1. A CT-scan of a dolphin head with density values ranging from -1024 to 2563, provided by Ted Cranford of UCSC. Thresholds 120.2 and 1125.2 on the dolphin detect the skin and bone surfaces respectively.

2. CFD data of an aircraft fin with density ranging from 0.1926 to 4.9775, provided by NAS at NASA-Ames Research Center. Threshold 1.0 on the aircraft fin finds an isopressure surface.

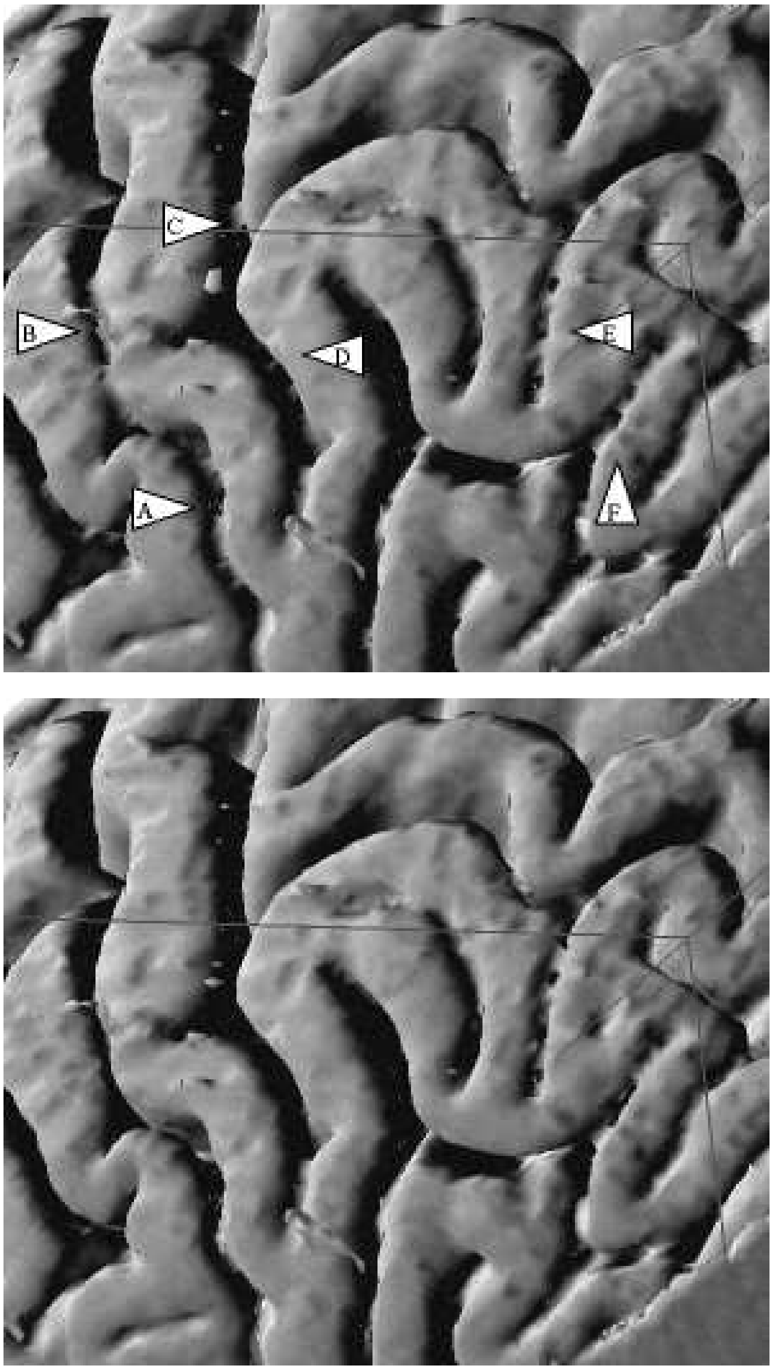


Figure 10: MRI isosurfaces of brain convolutions, threshold 650.5. Top image used UA-R2 connectivity. Arrows indicates sites of some of the incorrect “bridges”. In the bottom image, which used Quadratic Fit connectivity, most, but not all, bridges are absent.

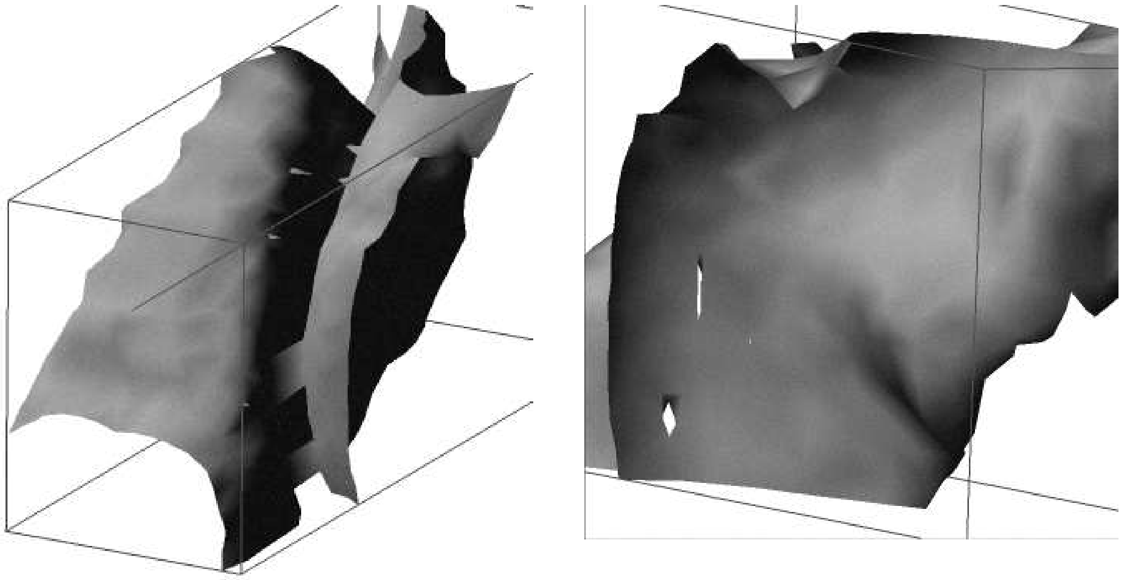


Figure 11: (Left) Detail view at site C of the previous figure, showing two “bridges” across the pre-central sulcus, present in UA-R2 and UA-R3 images. (Right) Cut-away side view shows that “bridges” have full dimensionality; the surface is continuous.

---

3. A quantum mechanics calculation of a high potential iron protein (“hipip”) generated by L. Noodleman and D. Case of Scripps Institute. Threshold 0.0 on the protein detects the “nodal surface”.

Table 1 provides some statistics on the nature of the volumes and the frequency of the 14 major cases, which are numbered as in Figure 1; we follow Lorensen and Cline [LC87], except that their cases 11 and 14 have been combined into our case 11. Inverse cases are included in the count of the non-inverted case. The number in square brackets after the ambiguous cases indicates how many topological subcases there are for the major case shown.

Table 2 compares four table-based methods in terms of number of polygons, number of disconnected edges, and running times. Only the marching cubes table produced disconnected edges.

Running times indicated are the user CPU time plus the system CPU time to process the volume after it has been read in through the generation of polygons but before display; the code included statistics gathering, so time is not indicative of “production” performance. Statistics were done on a Sun Sparcstation 1 workstation. Note that running times for disambiguation methods are only slightly greater than those using the original single table method (1a) without disambiguation. As mentioned before, these times *include* the time for gradient calculations for the gradient-consistency disambiguation methods.

We have drawn the following tentative conclusions from the timing results, and confirmed their reasonableness by inspection of the program:

1. Processing a cell with isosurface takes about 13 times as long as processing an empty cell. Nevertheless, the overall cost (CPU time) of processing empty cells was 30 to 70 percent of the total cost.

2. Processing an ambiguous cell with the quadratic fit method takes about three to four times as long as processing an unambiguous cell.

Data Set Threshold	Dolphin 120.2	Dolphin 1125.2	Blunt Fin 1.0	Protein 0.0
Number of Cells	3,718,093	3,718,093	31,117	226,981
Cells with Isosurface	106,612	64,908	4,036	28,828
Ambiguous	3.1%	3.3%	5.6%	0.67%
Case 1 :	19.7%	29.1%	21.4%	25.6%
Case 2 :	33.5%	29.3%	32.7%	37.0%
Case 3 : [2]	1.6%	1.4%	1.9%	0.3%
Case 4 :	0.2%	0.4%	0.3%	0.03%
Case 5 :	13.9%	19.9%	15.9%	17.5%
Case 6 : [2]	0.8%	1.3%	1.4%	0.3%
Case 7 : [8]	0.4%	0.1%	1.0%	0.01%
Case 8 :	27.9%	14.2%	22.1%	14.9%
Case 9 :	1.3%	2.8%	1.6%	4.1%
Case 10 : [4]	0.1%	0.1%	0.6%	0.04%
Case 11 :	0.5%	1.0%	0.5%	0.18%
Case 12 : [4]	0.4%	0.3%	0.6%	0.1%
Case 13 : [64]	0.02%	0.0%	0.05%	0.0%

Table 1: Case Frequency in Scientific Data. Ambiguous cases are followed by number of subcases in brackets. Percentages are based on cells intersecting the isosurface.

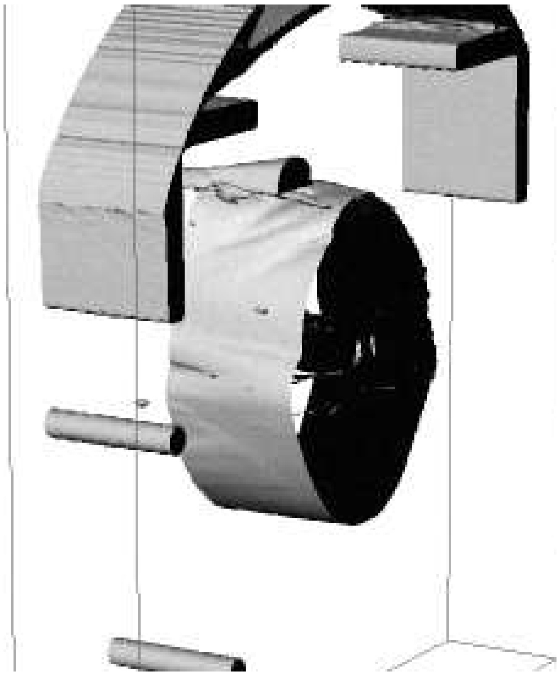
3. The cost of disambiguation is about five to ten percent of the overall cost of processing cells with isosurface, and an even smaller percentage of total time.

This study also showed that the facial average disagreed with the gradient heuristic methods on how to disambiguate 26 to 42 percent of the ambiguous cells. The two gradient heuristic methods disagreed with each other about ten percent of the time.

Examples of the type of discontinuities possible using the original marching cubes table are shown in Figure 13. The image is of the protein described above. Figure 13 shows a volume rendered with the original marching cubes table. Holes are obvious using this approach; discontinuities of this type are not uncommon on volumes in our experience. They are made more obvious by zooming in on the volume so that the contributions of individual cells show details.

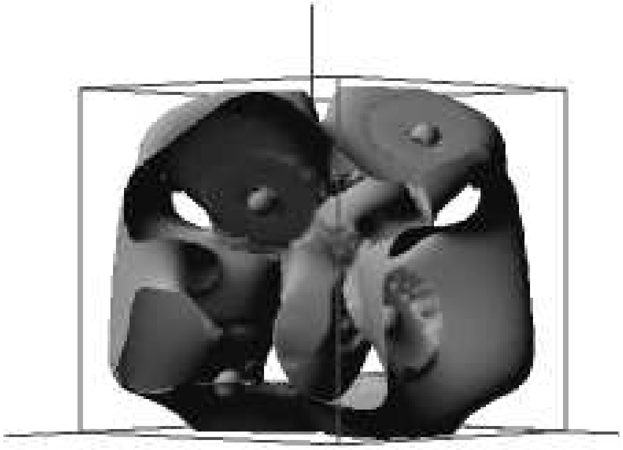
6 Topological Accuracy of the Heuristics

Without knowing the underlying function that produced the samples, or at least some properties of that function, no definite statements can be made about the topological accuracy of any disambiguation method. Unfortunately, such information seems to be rarely available to researchers on volume visualization for widely distributed volume data sets. Absent such information, we took two approaches toward evaluating accuracy. One was to create test volumes with known underlying functions, so true resampling could be performed on ambiguous cell faces. The other was to evaluate the smoothness of volumes with the aid of Fourier transforms.

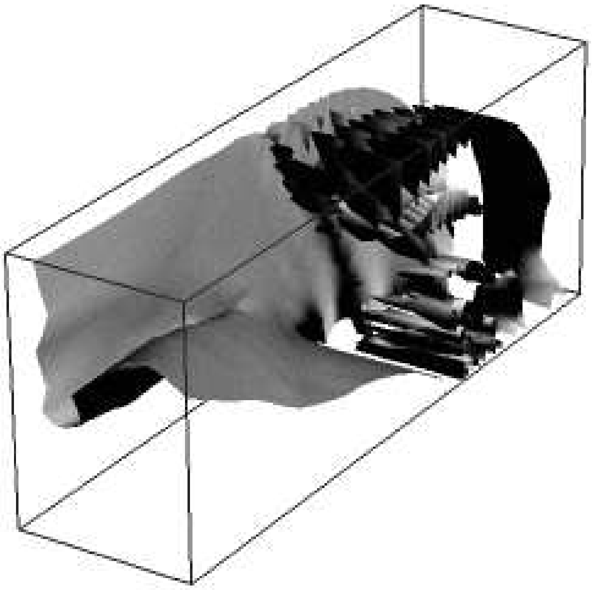


a. Dolphin threshold 120.2

b. Dolphin threshold 1125.2



c. Hipip threshold 0.0



d. Aircraft fin threshold 1.0

Figure 12: Isosurfaces analyzed:

Method	Data Set and Threshold			
	Dolphin (120.2)	Dolphin (1125.2)	Blunt Fin (1.0)	Protein (0.0)
Marching Cubes				
Polygons Generated	127,624	82,988	5069	35,327
Disconnected Edges	4300	2552	284	192
CPU Seconds	222	209	3.2	19.5
Facial Average				
Polygons Generated	128,835	83,720	5154	35,429
Disconnected Edges	0	0	0	0
CPU Seconds	221	210	3.2	19.5
Center-Pointing Gradient				
Polygons Generated	128,839	83,639	5196	35,405
Disconnected Edges	0	0	0	0
CPU Seconds	223	211	3.5	19.6
Quadratic-Fit Gradient				
Polygons Generated	128,804	83,633	5196	35,403
Disconnected Edges	0	0	0	0
CPU Seconds	223	211	4.1	19.7

Table 2: Comparison of Methods on Scientific Data.

6.1 Tests on Known Underlying Functions

To evaluate the topological accuracy of metric disambiguation heuristics, we generated 40 distributions of electric charges. For each distribution we produced one volume based on sampling the magnitude of the electric field, and one volume based on sampling the electric potential.[2] These kinds of functions were chosen because they do represent physical phenomena, and are similar to "influence functions" used in implicit surface modeling. Experimental design details of 400 sample runs are described in Appendix B.

For each sample run, we ran four metric disambiguation heuristics on the ambiguous faces: facial average, bilinear interpolation, center-pointing gradient, and quadratic fit. In addition, we ran a procedure that used resampling to discover the "correct" disambiguation. "Correct" is placed in quotes because this is still a numerical procedure that is not guaranteed analytically to produce the correct choice (see Appendix B). For the experiment, an heuristic was considered correct in an ambiguous cell only if all ambiguous faces of that cell were decided "correctly". The overall probability of deciding a cell correctly at random is less than one half because some cells have multiple ambiguous faces.

The correctness statistics for volumes representing electric field magnitude are shown in Table 3; here, 68 out of 200 sample runs contained some ambiguous cell. All differences between methods were highly significant, statistically, as explained further in Appendix B. The worst performing method was still significantly better than chance, by 4.5 standard deviations. but clearly there is room for much improvement.

Correctness statistics for volumes representing electric potential are shown in Table 4; in this case, 156 out of 200 sample runs contained some ambiguous cell. Only the largest difference between methods was statistically significant. All methods were significantly better than chance, but clearly there is room for much

[2] Recall that the electric potential (a scalar) due to one charge q at a distance r is q/r, and its gradient is the vector $(-x, -y, -z)q/r^3$, which is the electric field. The magnitude of the electric field is found by summing the gradients for all charges, *then* taking the magnitude of the result. This is quite different from summing the magnitudes of individual gradients, which would have no physical significance.

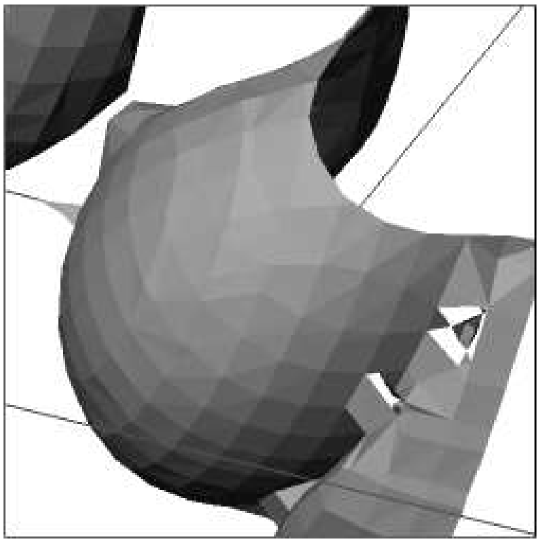


Figure 13: Discontinuities due to original marching cubes table appear as holes at lower right. Volume is a detail of "hipip".

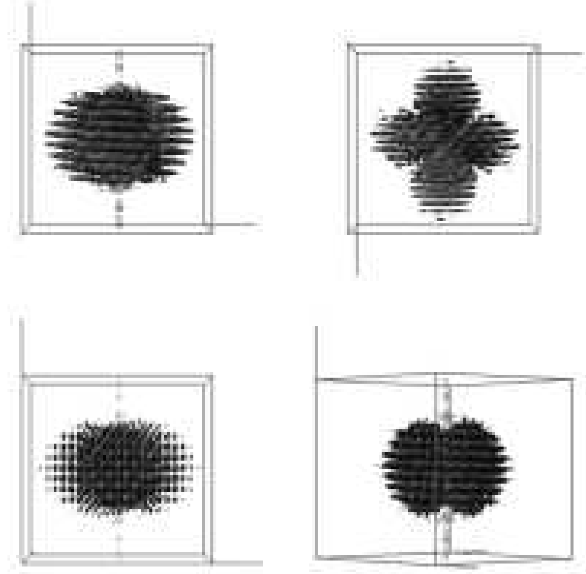


Figure 14: Isosurfaces in the 3D Fourier transform of the "hipip" volume show concentration of the spectral energy at low frequencies; red is at plus 1% of the maximum component and blue is at minus 1%.

Method	Correct Num.	Correct Pct.
(random)		45.2
Fac.Avg.	771	48.1
Bilinear	820	51.2
C.P.Grad.	941	58.7
Quad.Fit	1018	63.5

Table 3: Topological accuracy comparison on electric field magnitude volumes, covering a total of 1602 ambiguous cells. All differences were significant at level .001. (Small numbers indicate high significance.)

Method	Correct Num.	Correct Pct.
(random)		47.3
Fac.Avg.	617	66.6
Bilinear	625	67.4
C.P.Grad.	599	64.6
Quad.Fit	630	68.0

Table 4: Topological accuracy comparison on electric potential volumes, covering a total of 927 ambiguous cells. The difference between Quad. Fit and C.P.Grad is significant at level .001; other differences were insignificant even at .05.

improvement.

6.2 Fourier Analysis of Volumes

Fourier analysis can be used to determine the apparent smoothness of an unknown underlying function. We shall illustrate this with a discussion of the so-called *hipip* (High Potential Iron Protein) volume studied in Section 5.2. Assume this 64x64x64 volume is scaled as $\Delta x = \Delta y = \Delta z = 1/64$. Then the 3D discrete Fourier transform will compute frequencies (positive and negative) through 32 cycles, in increments of 1. We can ask the question: what radius must a sphere centered at the origin have in the 3D frequency domain in order to enclose 99% of the spectral energy of the volume (excluding the zero-frequency DC component)?[3]

For the hipip volume, the answer is that 99% of the non-DC spectral energy is contained within a sphere of radius 22.5 cycles, which is well below the maximum "observable" frequency of 32. Figure 14 shows this concentration qualitatively; the rendered isosurfaces are at plus and minus 1% of the maximum non-DC frequency component. For other percentages, we found that 98% of the non-DC spectral energy is within a sphere of radius 19.2, and 99.5% is within 26.1 cycles.

When almost all of the spectral energy of a volume appears at frequencies well below the maximum observable frequency, there is a reasonable presumption that the underlying function has been adequately sampled—otherwise there would be a large gap in the underlying function's spectrum. By "adequately sampled" we mean that this frequency information is representative of the underlying function itself, and not just of the sampled volume, that going to a higher resolution would produce little or no new spectral information.

Less decisive behavior was found in the blunt fin's spectrum, where frequencies up to 16 cycles were observable. Here 98% of the non-DC spectral energy was found within a radius of 11.0 cycles, 99% within 13.1 cycles, and 99.5% within 15.0 cycles. These computations were done in "computational space", i.e., as though the grid were regular, and cells were 1/32 on each side.

Finally, analysis of the dolphin data set showed that a presumption of adequate sampling was very doubtful. Scaling the cells to be 1/32 on each side permitted frequencies up to 16 cycles (plus and minus) in each coordinate direction to be observed. However, a sphere of radius 16 contained only 95% of the non-DC spectral energy; a radius of 19 was needed to capture 99%.

The point of this analysis is that, if the volume can be presumed to be adequately sampled, then a Fourier-based interpolation should be very accurate. Such interpolation essentially evaluates the inverse Fourier transform at non-grid points. If correct disambiguation is critical, the expense of Fourier-based interpolation, combined with searching for a "conclusive path", as described in Appendix B, may be justified.

Another implication of adequate sampling is that the possible error in a quadratic fit within a cell face can be bounded—essentially this error arises from third and higher derivatives. This subject is well-studied in signal processing. For example, if an error bound of .01 is known, and the quadratic fit $Q(t, 0)$ of Section 4.6 has a minimum value greater than .01 on the diagonal from ll to ur, it is conclusive that these corners are in the same connected component.

Fourier analysis holds much promise for volume visualization. These remarks indicate some directions to be pursued further.

7 Discussion and Conclusions

Correct isosurface generation is important for scientists to correctly interpret their data. Our experiments showed that methods that use information from neighboring cells are more likely on balance to pick correct isosurfaces, at least on the functions we tested.

It is equally important for users of isosurface algorithms to be aware of the possibility of incorrect topologies, so that they can adjust their interpretation of the data to take into account these possible errors.

In many graphics problems there is a trade-off of speed for accuracy. We attempted to get the best of both worlds by using a major case table for speed on the unambiguous cases, similarly to Lorensen and Cline [LC87], but with a careful disambiguation for accuracy when ambiguities did arise. The extra cost of careful disambiguation does not seem to be a serious burden in practice, due to the relatively low statistical

[3] Recall that spectral energy is defined as $\sum |a_{ijk}|^2$, where a_{ijk} are 3D frequency components, appropriately normalized.

frequency of ambiguous cases. As Albert Einstein said, "Things should be made as simple as possible, but no simpler." We believe this treatment helps to correct the oversimplifications of previous methods.

In summary, we expanded the case table to allow a choice of possible topologies on ambiguous cells, introduced two new methods for picking among these choices, and experimentally studied those and several other methods.

All metric disambiguation methods (Section 4) guarantee continuity between cells. Of the three methods studied on scientific data, the simplest is the "facial average" method which uses only the four corners values for the ambiguous face to decide upon topology. While simplest to compute, we believe the true topology is better estimated by extended metric methods, which can provide further insight into the underlying function. The two "gradient consistency heuristic" methods consider an extended neighborhood. To guarantee consistency across shared faces, the extra data points used all lie in the grid plane of the shared face. While the two heuristic methods rarely differ and their differences are subtle, the more expensive "quadratic fit" method has been shown to reflect more accurately the behavior of a quadratic underlying function than does the "center-pointing gradient" method.

Acknowledgements

We wish to thank Judy Challinger and Orion Wilson for their programming, Ted Cranford for his CT-scan data of dolphins, and NAS/NASA-Ames Research Center for the aircraft fin data. The "hipip" data set was developed by Louis Noodleman and David Case of the Scripps Clinic, and Michael Pique of the Scripps Institute supplied us with additional background on it. Tom Malzbender of Hewlett-Packard assisted us with Fast Fourier Transform code. Nelson Max of UC Davis made some helpful suggestions on the presentation. David Dean of NYU helped us to understand CT-scan data. We also thank Silicon Graphics Incorporated, Sun Microsystems, and Digital Equipment Corporation for their generous gifts of equipment, without which this research would not have been possible. We wish to thank the reviewers for their many helpful suggestions. This research was supported in part by a State of California Micro-Electronics Grant, a UCSC Committee on Research Grant, NSF grants CCR-8958590 and ASC-9102497, and NASA-Ames Research Center Cooperative Agreement Interchange No. NCA2-430.

Appendix A Tricubic Interpolation Equations

A tricubic polynomial may be fit to the cell data, using the 4x4x4 region of voxels surrounding the cell to specify the function. The function is the 3-dimensional volumetric version of the Catmull-Rom spline [CR74, FDFH90].

In a Catmull-Rom spline *curve*, the curve interpolates the interior two of the four points defining the curve and uses the outer two points to calculate the gradient of the curve at the interior two. Thus, the curve formula can be stated either in terms of four points, or in terms of two points and two gradients at those points (the Hermite formulation). A Catmull-Rom spline *surface* would use 16 points in a 2-dimensional 4x4 grid and interpolate the interior four. A Catmull-Rom spline *volume*, which we use here, considers 64 points in a 3-dimensional 4x4x4 grid and interpolates the interior eight defining the middle cell of the grid. This method produces a tricubic function within the cell that is C^1 continuous with tricubic functions in neighboring cells at faces.

The gradients for the interior points are found using standard central differences:

$$\bigtriangledown f(x,y,z) = (\bigtriangledown f_x(x,y,z), \bigtriangledown f_y(x,y,z), \bigtriangledown f_z(x,y,z)) \qquad \text{where} \tag{13}$$

$$\bigtriangledown f_x(x,y,z) = \frac{f(x+\Delta x,y,z) - f(x-\Delta x,y,z)}{2\Delta x} \tag{14}$$

$$
\begin{aligned}
\nabla f_y(x,y,z) &= \frac{f(x,y{+}\Delta y,z)-f(x,y{-}\Delta y,z)}{2\Delta y} \\
\nabla f_z(x,y,z) &= \frac{f(x,y,z{+}\Delta z)-f(x,y,z{-}\Delta z)}{2\Delta z}
\end{aligned}
$$

where Δx, Δy, and Δz are the distances between two neighboring sample points in (x,y,z). This assumes a regular sampling, though the interval may vary among the three directions. For the remainder of this section we assume the coordinates have been rescaled and nondimensionalized to make Δx, Δy, and Δz unity; this simplifies the presentation without any loss of generality.

To derive the tricubic fit, first consider four sample values f_{-1}, f_0, f_1, and f_2 in one dimension, and a univariate cubic curve $F(u)$ obeying the constraints $F(0)=f_0$, $F'(0)=f'_0$, $F(1)=f_1$, and $F'(1)=f'_1$, where, by central differences, $f'_i=\frac{1}{2}(f_{i+1}-f_{i-1})$. We shall use $F(u)$ as the interpolated sample value in the interval $0\le u\le 1$. The desired $F(u)$ can be expressed in terms of the four sample values as

$$
F(u) = \sum_{-1}^{2} f_i\ B_i(u) \tag{15}
$$

with the appropriate blending functions, $B_i(u)$. The $B_i(u)$ are found by routine algebra to be:

$$
\begin{aligned}
B_{-1}(u) &= \tfrac{1}{2}(-u^3+2u^2-u) \\
B_0(u) &= \tfrac{1}{2}(3u^3-5u^2+2) \\
B_1(u) &= \tfrac{1}{2}(-3u^3+4u^2+u) \\
B_2(u) &= \tfrac{1}{2}(u^3-u^2)
\end{aligned} \tag{16}
$$

Using a corresponding indexing scheme (from -1 to 2) for the cell in three dimensions, the tricubic function is given by

$$
F(x,y,z) = \sum_{k=-1}^{2}\sum_{j=-1}^{2}\sum_{i=-1}^{2} f_{i,j,k} B_i(x)B_j(y)B_k(z) \tag{17}
$$

We can easily verify that this function fits both the sample values and the computed gradients at the 8 cell vertices by observing that the blending functions' values and first derivatives are zero at 0 and 1, except for the cases:

$$
B_0(0)=B_1(1)=1, \qquad B'_1(0)=B'_2(1)=\tfrac{1}{2}, \qquad B'_{-1}(0)=B'_0(1)=-\tfrac{1}{2} \tag{18}
$$

Appendix B Test Details on Known Underlying Functions

Experimental Design

To evaluate the topological accuracy of various disambiguation heuristics, as reported in Section 6.1, we generated 40 distributions of electric charges. Each distribution consisted of 600 charges placed uniformly at random in a sphere of radius 10, surrounding a cube of side 8. Charges were ± 1 with equal probability. For each distribution we produced one volume based on sampling the magnitude of the electric field, and one volume based on sampling the electric potential.

Thus a total of 80 volumes were tested, each volume being a cube of 10 cells on a side (11x11x11 samples). On each volume 5 isosurfaces were extracted. An isosurface was extracted only in the interior 8^3 cells to avoid edge effects with gradients. A *sample run* consists of one combination of volume and threshold.

For each sample run, we ran 4 disambiguation heuristics on the ambiguous faces: facial average, bilinear interpolation, center-pointing gradient, and quadratic fit. In addition, we ran a procedure that used

Methods	Plur-ality	Observ-ations	Num. of Std.Devs.	Signif. Level
Bilinear vs. Fac.Avg.	+49	153	3.96	.001
C.P.Grad. vs. Fac.Avg.	+170	372	8.81	.001
C.P.Grad. vs. Bilinear	+121	487	5.48	.001
Quad.Fit vs. Fac.Avg.	+247	555	10.48	.001
Quad.Fit vs. Bilinear	+198	514	8.73	.001
Quad.Fit vs. C.P.Grad.	+77	131	6.73	.001

Table 5: Topological accuracy comparison on electric field magnitude volumes, covering a total of 1602 ambiguous cells.

Methods	Plur-ality	Observ-ations	Num. of Std.Devs.	Signif. Level
Bilinear vs. Fac.Avg.	+8	134	0.69	
C.P.Grad. vs. Fac.Avg.	-18	206	-1.25	
C.P.Grad. vs. Bilinear	-26	276	-1.57	
Quad.Fit vs. Fac.Avg.	+13	227	0.86	
Quad.Fit vs. Bilinear	+5	249	0.32	
Quad.Fit vs. C.P.Grad.	+31	55	4.18	.001

Table 6: Topological accuracy comparison on electric potential volumes, covering a total of 927 ambiguous cells.

resampling to discover the "correct" disambiguation. "Correct" is placed in quotes because this is still a numerical procedure that is not guaranteed analytically to produce the correct choice. The procedure searches for what is called a *conclusive path*, defined below, connecting corners of the same sign in an ambiguous cell face. It begins with an 11x11 resampled grid covering the cell face, i.e., 100 subfaces. If no conclusive path is found at this resolution, it tries a 21x21 resampled grid. If it fails again it goes to 41x41, and finally, if necessary to 81x81. After this, it gives up and called the face "indecisive".

A *conclusive path* is defined as a series of line segments joining two diagonally opposite corners of the ambiguous cell face, such that:

1. Each segment proceeds from a resampled point to another resampled point that is closer to the opposite corner and is adjacent in the resampled grid either horizontally, vertically, or diagonally. Except for the two corners joined, all points must be interior to the cell face.

2. On each segment, except the first and last, a cubic spline function maintains the correct sign relative to the threshold value, that is, the same sign as the corners. The cubic is fit to the end-points of the segment and the resampled points "in front of" and "behind" the segment, and equi-distant.

3. No ambiguous subfaces (faces in the resampled grid) are touched or traversed by the path.

The last condition ensures that a conclusive path cannot exist (at the same resolution) to connect the other pair of cell corners. In our tests conclusive paths were usually found at the coarsest resolution, and no indecisive faces were found.

Statistical Significance

To obtain a sharp evaluation of statistical significance, a pairs test was used. Essentially, an "observation" is the *difference* in performance between two methods (the pair) on the same sample. For the experiment, an heuristic was considered correct in an ambiguous cell only if all ambiguous faces of that cell were decided "correctly", as described above. The performance value is 1 for each "correct" ambiguous cell, and 0 for each "incorrect". Cells in which the two methods had no difference in performance are *not* counted as observations. A positive performance difference is called a "win" and a negative one is a "loss".

Statistical significance computations attempt to reject the "null hypothesis" concerning the pair of methods being compared. For our test the null hypothesis is that, given that the two methods have a difference in performance on some sample, each has an equal probability of "winning".

The statistical significance level of a particular plurality represents (an upper bound on) the probability of getting a sample this biased under the null hypothesis. For example, line 1 of Table 5 can be read as, "If the Bilinear and Facial Average methods had an equal probability of "winning" against each other, then the probability is less than .001 that Bilinear would achieve a plurality of 49 or more in 153 trials; this is 3.96 standard deviations from the mean." Thus small numbers indicate high significance.

References

[AFH80] E. Artzy, G. Frieder, and G. Herman. The theory, design, implementation, and evaluation of a three-dimensional surface generation program. *Computer Graphics (ACM Siggraph Proceedings)*, 14(3):2–9, July 1980.

[AFH81] E. Artzy, G. Frieder, and G. Herman. The theory, design, implementation, and evaluation of a three-dimensional surface detection algorithm. *Computer Graphics and Image Processing*, 15(1):1–24, January 1981.

[Bak89] H. H. Baker. Building surfaces of evolution: The weaving wall. *International Journal of Computer Vision*, 3:51–71, 1989.

[Blo88] Jules Bloomenthal. Polygonization of implicit surfaces. *Computer-Aided Geometric Design*, 5:341–355, 1988.

[CDL+87] H. E. Cline, C. L. Dumoulin, W. E. Lorensen, H. R. Hart, Jr., and S. Ludke. 3D reconstruction of the brain from magnetic resonance images. *Magnetic Resonance Imaging*, July 1987.

[CHRU85] Lih-Shyang Chen, Gabor T. Herman, Anthony Reynolds, and Jayaram K. Udupa. Surface shading in a cuberille environment. *IEEE Computer Graphics and Applications*, 5(12):33–43, December 1985.

[CIBL83] L. T. Cook, S. J. Dwyer III, S. Batnitzky, and K. R. Lee. A three-dimensional display system for diagnostic imaging applications. *IEEE Computer Graphics and Applications*, 3(5):13–19, August 1983.

[CLL+88] Harvey E. Cline, William E. Lorensen, Sigwalt Ludke, Carl R. Crawford, and Bruce C. Teeter. Two algorithms for the reconstruction of surfaces from tomograms. *Medical Physics*, June 1988.

[CR74] E. Catmull and R. Rom. A class of local interpolating splines. In R. Barnhill and R. Riesenfeld, editors, *Computer Aided Geometric Design*, pages 317–326. Academic Press, San Francisco, 1974.

[CS78] H. N. Christiansen and T. W. Sederberg. Conversion of complex contour line definitions into polygonal element mosaics. *Computer Graphics (ACM Siggraph Proceedings)*, 12(3):187–192, August 1978.

[DK91] A. Doi and A. Koide. An efficient method of triangulating equi-valued surfaces by using tetrahedral cells. *IEICE Transactions on Communications Electronics Information and Systems*, E-74:214–224, 1991.

[Dür88] Martin J. Dürst. Letters: Additional reference to "marching cubes". *Computer Graphics (ACM Siggraph Quarterly)*, 22(2), April 1988.

[FDFH90] James D. Foley, Andies Van Dam, Steven Feiner, and John Hughes. *Computer Graphics: Principles and Practice.* Addison-Wesley Publishing Company, Reading, Mass., 2 edition, 1990.

[FKU77] H. Fuchs, Z. M. Kedem, and S. P. Uselton. Optimal surface reconstruction from planar contours. *Communications of the ACM*, 10:693–702, October 1977. Also in Siggraph '77.

[GN89] Richard S. Gallagher and Joop C. Nagtegaal. An efficient 3-D visualization technique for finite element models. *Computer Graphics*, 23(3):185–194, 1989.

[HB86] K. H. Höhne and R. Bernstein. Shading 3D-images from ct using gray-level gradients. *IEEE Trans. Medical Imaging*, pages 45–57, 1986.

[HL79] G. T. Herman and H. K. Liu. Three-dimensional display of human organs from computer tomography. *Computer Graphics and Image Processing*, 9(1), 1979.

[HU83] G. T. Herman and J. K. Udupa. Display of 3-D digital images: Computational foundations and medical applications. *IEEE Computer Graphics and Applications*, 3(5):39–46, August 1983.

[Kal91] A. D. Kalvin. *Segmentation and Surface-Based Modeling of Objects in Three-Dimensional Biomedical Images.* PhD thesis, New York University, 1991.

[KB89] Devendra Kalra and Alan H. Barr. Guaranteed ray intersections with implicit surfaces. *Computer Graphics (ACM Siggraph Proceedings)*, 23(3):297–306, July 1989.

[KDHB92] A. D. Kalvin, D. Dean, J-J. Hublin, and M. Braum. Visualization in anthropology: Reconstruction of human fossils from multiple pieces. In *Visualization 92*, pages 404–410. IEEE, 1992.

[KDK86] A. Koide, A. Doi, and K. Kajioka. Polyhedral approximation approach to molecular orbit graphics. *Journal of Molecular Graphics*, 4:149–156, 1986.

[LC87] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4):163–169, July 1987.

[Lev88] Marc Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(3):29–37, March 1988.

[LV80] S. Lobregt and P. W. Verbeck. Three-dimensional skeletonization. *IEEE Transactions on Pattern Matching and Machine Intelligence*, PAMI-2(1):75–77, January 1980.

[Nat91] B. K. Natarajan. On generating topologically correct isosurfaces from uniform samples. Technical Report HPL-91-76, Software and Systems Laboratory, Hewlett-Packard Company, Page Mill Road, Palo Alto, Ca, 1991. (To appear in *Visual Computer*).

[NH91] Gregory M. Nielson and Bernd Hamann. The asymptotic decider: Resolving the ambiguity in marching cubes. In *Proceedings of Visualization '91*, pages 83–91, San Diego, Ca., October 1991.

[RNMK91] H. Rusinek, M. E. Noz, G. Q. Maguire, and A. D. Kalvin. Quantitative and qualitative comparison of volumetric and surface rendering techniques. *IEEE Transactions on Nuclear Science*, 38(2):659–662, 1991.

[Sri81] S. N. Srihari. Representation of three-dimensional digital images. *Computing Surveys*, 13(4):399–424, 1981.

[UA90] J. K. Udupa and V. G. Ajjanagadde. Boundary and object labelling in three-dimensional images. *Computer Vision, Graphics, and Image Processing*, 51:355–369, 1990.

[Udu89] Jayaram K. Udupa. Display of medical objects and their interactive manipulation. In *Proceedings of Graphics Interface '89*, pages 40–43, London, Ontario, June 1989.

[UK88] Craig Upson and Michael Keeler. The v-buffer: Visible volume rendering. *Computer Graphics*, 22(4):59–64, July 1988.

[USH82] Jayaram K. Udupa, Sargur H. Srihari, and Gabor T. Herman. Boundary detection in multidimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-4(1):41–50, January 1982.

[WH79] Thomas Wright and John Humbrecht. Isosurf – an algorithm for plotting iso-valued surfaces of a function of three variables. *Computer Graphics (ACM Siggraph Proceedings)*, 13(2):182–189, August 1979.

[Win88] James M. Winget. Advanced graphics hardware for finite element results display. In *Advanced Topics in Finite Element Analysis*, New York, 1988. American Society of Mechanical Engineers. Presented at Pressure, Vessels, and Piping Conf.

[WMW86] G. Wyvill, C. McPheeters, and B. Wyvill. Data structures for soft objects. *The Visual Computer*, 2(4):227–234, August 1986.

[WVG90] Jane Wilhelms and Allen Van Gelder. Topological considerations in isosurface generation, extended abstract. *Computer Graphics*, 24(5):79–86, 1990. Special Issue on San Diego Workshop on Volume Visualization; also UCSC technical report UCSC-CRL-90-14.

[WVG92] Jane Wilhelms and Allen Van Gelder. Octrees for faster isosurface generation. *ACM Transactions on Graphics*, 11(3):201–227, July 1992. Extended abstract in ACM Computer Graphics 24(5) 57–62; also UCSC technical report UCSC-CRL-90-28.