

Exploring Geovisualization
J. Dykes, A.M. MacEachren, M.-J. Kraak (Editors)
© 2005 Elsevier Ltd. All rights reserved.

Chapter 17

Interactive Approaches to Contouring and Isosurfacing for Geovisualization

Adriano Lopes, Department of Informatics of the Faculty of Science and Technology/CITI, New University of Lisbon, Lisbon, Portugal
Ken Brodlie, School of Computing, University of Leeds, Leeds, UK

Keywords: interactive, contour, isosurface, correctness, robustness, accuracy, Marching Cubes, bilinear/trilinear interpolation

Abstract

This chapter describes ways in which contouring and isosurfacing techniques for 3D computer graphics have evolved. In particular, we show how they are strongly intertwined and how they have moved from being non-interactive to interactive processes, making them appropriate for the kinds of dynamic representation required for geovisualization. The analysis relies upon data defined on a regular, rectilinear grid, and a bilinear or trilinear data model within each 2D or 3D cell.

17.1 Introduction

Geospatial data is inherently structured into 2, 3 or 4 dimensions. For geovisualization, when this data is a scalar field, a very common approach is to show the set of points where this scalar field has a certain value, or threshold. For 2D data this is contouring, whilst for 3D data, this is isosurfacing. The two techniques are strongly related: both involve extracting geometry from data, by taking cross-sections through the space of the dependent variable. In the case of contouring, lines (isolines) of equal value are extracted from 2D data; in the case of isosurfacing, we extract surfaces (isosurfaces) of equal value from 3D data. Likewise, the development of techniques for generating isolines and isosurfaces are strongly intertwined – new approaches for one can lead to new approaches for the other.

Traditionally, both contouring and isosurfacing have been seen as non-interactive processes in an interactive environment to support visualization in its broadest sense and geovisualization in particular (MacEachren and Kraak, 2001). In the early days of contouring, packages such as SYMAP (Fisher, 1963) would generate detailed maps that were plotted on paper for subsequent, off-line examination. Today, contouring remains a useful technique, but the style of working has changed and we need techniques that perform

well in an interactive environment to support the process of geovisualization. We need to be able to change contour levels and see the effect immediately; we need to be able to zoom in and out; and we may want to keep the contour levels constant, but vary the data to form an animation. In each of these scenarios, we need the contouring technique to be fast, robust, and smooth with respect to changes of data and contour levels.

For 3D data, in the early days of geovisualization, the most common approach was to slice through the dataspace, so that we reduce the visualization problem to 2D, and contouring can be applied. A sequence of slices can be generated, at intervals through the 3D space. However, as technology has developed over the years, it has become possible to treat the 3D data as a single entity, and use a 3D visual representation – in this case, isosurfacing to view a surface of common value. There are significant advantages to be gained from a 3D approach (see Wood et al., this volume (Chapter 14)).

Isosurfacing naturally has a much shorter history than contouring, but in many ways a very similar one. In the early days, for isosurfacing this was the late 1980s, it was seen as a relatively slow operation, particularly for large data sets, and this parallels the early days of contouring. Today, advances in computing hardware allow the surface extraction to be achieved much more quickly, and advances in graphics hardware boards allow fast rendering of the extracted surfaces; see Wood et al., this volume (Chapter 14) and Döllner, this volume (Chapter 16). Thus isosurfacing has now become an interactive technique, and we can set out very similar requirements to those for contouring.

In this chapter, we describe how methods for contouring and isosurfacing have evolved, and in order to show how strongly related the two topics are, our description will intertwine the two. We will show how improvements to the methods have been achieved as a result of a clearer definition of a model underlying the data, i.e., a model of the entity from which we suppose the data has been sampled. Ultimately, we show that for high-quality visualization in an interactive environment, we need to base our approach on a visualization of that underlying model. Only then can we adjust thresholds, apply zooming or vary the data values, and maintain a consistent visualization.

17.2 The Evolution of Contouring and Isosurfacing

This chapter traces the evolution of contouring and isosurfacing in a number of stages. We assume throughout that the data is defined on a regular, rectilinear grid. We are interested in estimating the behaviour throughout the region, not just at the grid points, and we build a bilinear, or trilinear, model within each 2D or 3D cell. We make the fundamental assumption that this order of interpolation is an appropriate compromise between accuracy and speed.

In §17.2, we describe the essential problem to be solved in generating contours or isosurfaces, and then sketch the outline of the remainder of the chapter.

17.2.1 Contouring

Contouring aims to represent scalar data of constant value on a 2D plane, by lines joining those points of equal value (isolines or contour lines). Though an old technique it is still

very useful, as shown in many examples: isobars and isotherms in weather forecasting, isolines of digital elevation data (contours) in Cartography, etc.

The data is defined point wise on a 2D rectangular grid and it is common practice to assume it is modelled as a bilinear interpolant on each cell of the grid (linear along the edges). Therefore, a contouring method should construct an approximation to the contours of the bilinear interpolant. Since the fundamental 2D drawing element is a line, we seek a piecewise linear approximation to the contours. A common strategy is to scan and examine each cell in sequence, independently. When there are grid values higher and lower at the corners of the cell then part of the contour lies within the cell.

Formally, the problem is to compute and display isolines of a threshold value giving the values of a bilinear function F^{2D} at the vertices of the rectangular cell. For convenience and without loss of generality the cell domain is transformed into a unit square $[0, 1] \times [0, 1]$ and the threshold value is zero.

$$F^{2D}(x, y) = ax + by + cxy + d, \quad (17.1)$$

with $a = F_{10} - F_{00}$, $b = F_{01} - F_{00}$, $c = F_{00} + F_{11} - F_{01} - F_{10}$ and $d = F_{00}$, where F_{00} , F_{01} , F_{10} and F_{11} are function values at the vertices of the cell. Assuming $c \neq 0$ the contour curve within the cell is a hyperbola (with $c = 0$ it is a line).

17.2.2 Isosurfacing

Isosurfacing aims to represent a surface of a given threshold value drawn in the interior of volumetric scalar data. It is the natural extension of contouring; the boundary surface separates points with values greater or equal to threshold from those with values less.

Data is defined pointwise on a 3D rectangular grid and the model within each cell is the trilinear interpolant – linear along edges, bilinear across the faces of the cell. As in contouring, a common strategy is to scan and examine each cell in sequence, independently. If grid values spanning the threshold are found, then part of the isosurface is within the cell.

Without loss of generality the cell domain is assumed to be a unit cube with corner at origin, and the threshold value is considered as zero. The interpolant F^{3D} can be written in terms of the vertex values F_{ijk} , $i, j, k = 0, 1$ or in a equivalent way:

$$F^{3D}(x, y, z) = a + ex + cy + bz + gxy + fxz = dyz + hxyz \quad (17.2)$$

with $a = F_{000}$, $b = F_{001} - F_{000}$, $c = F_{010} - F_{000}$ and so on.

Any representation should provide an approximation to the surface of the trilinear interpolant. Since the triangle is the basic drawing element in 3D, we seek a piecewise linear approximation forming a triangulation in 3D.

17.2.3 From passive to interactive

We trace the evolution in a series of four steps – with each step we improve the representation of the underlying bilinear, or trilinear, model. By the final step, we have a representation that is suitable for interactive work where we wish to explore rather than just present.

In the first step, we use the fact that the model is linear along the edges in order to accurately determine the intersections of the interpolant with the grid lines. However, in this first evolutionary step, no attempt was made to properly follow the curved nature of the interpolant within a cell, because this is computationally expensive. Instead, an *ad hoc* strategy was typically used in order to connect the intersection points with lines in the case of contouring, or triangular pieces in the case of isosurfacing. This was the normal practice in the early days of contouring (1960s) and isosurfacing (late 1980s).

As explained in §17.3, it was almost immediately realized that there are cases where a number of alternative ways of joining up the points are possible – the so-called ambiguous cases. An *ad hoc* strategy here is fraught with danger.

Hence, in the second evolutionary step, described in §17.4, the solution was to decompose the cell into triangular, or tetrahedral, parts. The advantage is that the contours are now exactly straight lines, the isosurface exactly flat planes, and so very fast to draw. This represents a major improvement, but the model being piecewise linear within the grid cell, is not continuous in first derivative across the cell interior.

Thus in a third evolutionary step, described in §17.5, the strategy returns to consideration of the bilinear, or trilinear, interpolant. An attempt is made to approximate the interior topology of the interpolant. This turns out to be non-trivial particularly for the isosurface case, but we reach a stage where the correct topology is generated in a piecewise linear representation (lines or triangles). Notice the contrast between the approaches: in the second evolutionary step, the cell is decomposed into pieces and we fit linear models in each piece (with exactly linear isolines or isosurfaces); in the third step, we fit a bilinear, or trilinear, model in the cell (with non-linear isolines or isosurfaces) and construct a piecewise linear approximation to the isolines/isosurfaces.

In the final step, described in §17.6, attention turns to improving the representation of the previous step. We not only want the correct topology but also want the representation to work well in interactive exploration of the data.

17.3 Correctness of Boundary Points

17.3.1 Contouring

Consider the case of a function defined at the corners of a unit square as described in §17.2.1, where we are interested in drawing the zero contour and where we assume a bilinear interpolant as in equation (17.1). The intersection points of the hyperbolae with the cell edges are easily and accurately calculated by inverse linear interpolation. The basic method is then to approximate the hyperbolae by straight lines connecting those intersection points.

There are 16 different vertex cases depending on whether the values at the corners are positive or negative. The 16 cases can be reduced to just four canonical cases as shown in [Figure 17.1](#): no contour at all; a single segment cutting off a corner; a segment

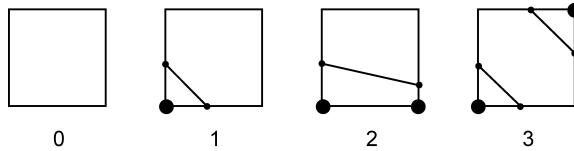


Figure 17.1. Canonical configurations for rectangular cells when data varies linearly between adjacent grid points. Positive vertices are marked.

cutting off two corners; or two segments cutting off opposite corners. However, this final case requires an arbitrary decision to be taken as regards which corners are cut off: it occurs when one pair of opposite vertices are of different sign from the other pair. This is discussed as a difficulty in many early papers on contouring (Sutcliffe, 1980). *Ad hoc* strategies, such as always “keeping high ground on the right” emerged to provide some consistency, if not correctness.

So the main features in this basic method are:

- correctness of boundary points so there is consistency from one cell to the next;
- no guarantee of topological correctness.

17.3.2 Isosurfacing

The classical and simple approach to isosurfacing is the Marching Cubes (MC) algorithm, proposed by Lorensen and Cline (1987), with a similar suggestion from Wyvill et al. (1986). Each vertex of a cube can be either greater than or less than the threshold value, giving 256 different configurations. The intersections of the isosurface with the edges of the cube are easily and accurately calculated by inverse linear interpolation. Then the set of intersection points are triangulated to yield an approximation to the isosurface within the cube. Lorensen and Cline argued that for reasons of symmetry and complementarity there are only 15 canonical configurations, and proposed corresponding triangulations of the isosurface. For a given configuration (from the set of 256), they provide a look up table to give the corresponding canonical configuration and hence its triangulation. But a further case can be removed by reflectional symmetry, leaving a canonical set of 14 cases. These are shown in Figure 17.2. Unfortunately, the efficient code of the small look up table caused inconsistent matching of surfaces between adjacent cells, so that “holes” could appear as first reported by Durst (1988), within a short time of the publication of Lorensen and Cline’s paper.

The behaviour of the interpolant F^{3D} inside the cube is non-trivial and is a cubic surface. The difficulties identified by Durst were exactly the problems identified by the early contouring pioneers, when using *ad hoc* strategies to represent the contours of a bilinear interpolant simply by joining up intersection points. As we see later the problem is deeper in 3D since we need to understand the correct behaviour both on the faces, and in the interior.

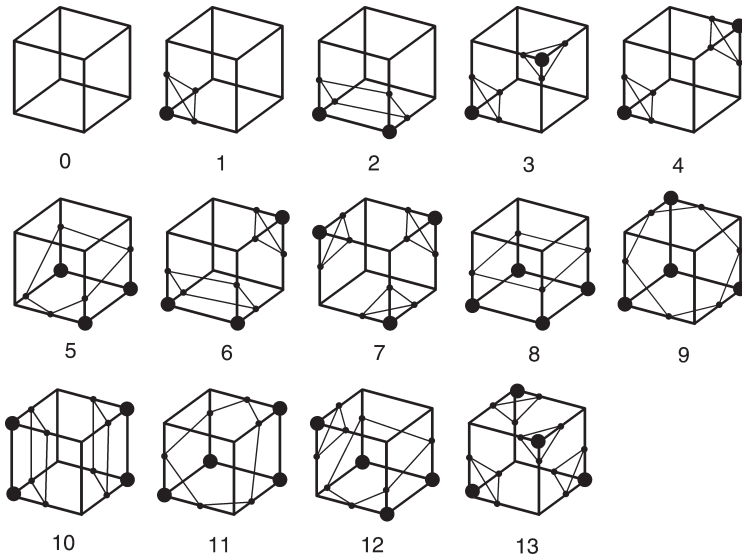


Figure 17.2. Basic cases in the Marching Cubes algorithm, and examples of how the points of intersection between the isosurface and the edges of the cube can be connected. Positive vertices are marked.

The main features of this basic MC algorithm are:

- correctness along the edges of the cube;
- potential flaws as a consequence of mismatched patterns between adjacent cells. Therefore, no topological correctness on the boundary faces, and so no consistency of solution from cell to cell;
- topologically is incorrect in the cell.

17.4 Grid Decomposition

In §17.4, we see how decomposition of the basic rectangular shaped cell into triangular (or tetrahedral) shaped pieces addresses some of the problems discussed in §17.3.

17.4.1 Contouring

As an attempt to solve the ambiguity in the basic contouring method, several authors proposed decomposing the basic square cell into four triangles (Dayhoff, 1963; Heap, 1972). The value at the centre of the cell is taken as the average of the four corner points. Then two extra intersection points lying on the diagonals of the cell are obtained by using inverse linear interpolation (Figure 17.3). Notice that the averaging process at the centre corresponds to bilinear interpolation, but is then followed by a linear interpolation within each triangle.

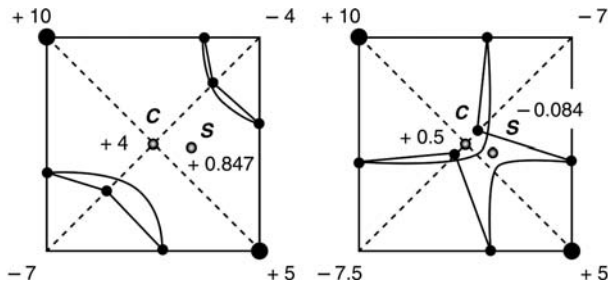


Figure 17.3. Four-triangles method, with two extra points lying on the diagonals of the basic square cell (Lopes, 1999). The solution on the left is topologically correct, but that on the right is topologically incorrect. It is only correct when both the centre point C and the saddle point S have the same sign.

The main features of this method are:

- correctness of boundary points so consistency of solution from cell to cell;
- no guarantee of correct topology as [Figure 17.3](#) shows. It is only correct when both the saddle point and the centre lie in the same region of the hyperbolae. In other words, when both have the function value with the same sign;
- discontinuity of derivative of the contour as the diagonal is crossed.

17.4.2 Isosurfacing

Similar to the four-triangles method for contouring, the Marching Tetrahedra (MT) method ([Doi and Koide, 1991](#)) emerged largely as the result of trying to solve the problem of ambiguity in the MC algorithm. In this case, the cube is decomposed into tetrahedra and then the MT method is applied to each tetrahedron. Any additional vertex values that are needed can be obtained by trilinear interpolation. The advantage of this decomposition is similar to the four-triangle contouring method: a linear interpolant is fitted in each tetrahedron, based upon the four data values at the vertices:

$$F^T(x, y, z) = a + bx + cy + dz \quad (17.3)$$

The exact isosurface within a tetrahedron is a plane (one or two triangles).

The main features are as follows:

- it is simple to triangulate the surface due to the linear interpolant;
- it is unclear how best to decompose the cube into tetrahedra;
- it does not always provide topological correctness within the cube (when compared with the isosurface of the trilinear interpolant);
- there is discontinuity of the model across the faces of the tetrahedra.

17.5 Topological Correctness

We discuss the next stage in §17.5. Rather than decomposing the cell, we aim to work with the bilinear, or trilinear, interpolant within the whole cell, and try to

understand its behaviour so we can represent it correctly from a topological viewpoint.

17.5.1 Contouring

To the best of our knowledge, this stage emerged first as an improvement to isosurfacing, and by implication, becomes an improvement to contouring. It was developed by [Nielson and Hamann \(1991\)](#), who were trying to solve ambiguity in MC, looked at the behaviour of the bilinear interpolant on the cell faces – this is exactly a contouring problem (the intersection of an isosurface with a cell face is a contour line). [Nielson and Hamann \(1991\)](#) looked at the asymptotes of the hyperbolic arcs and the behaviour at their intersection, the saddle point. When one pair of opposite vertices have data values of different sign from the other pair, the pairwise connection is established so as to “cut off” the vertices of opposite sign to the saddle point value. [Figure 17.4](#) depicts this method – the previous basic method with asymptotic decider to resolve the ambiguity.

However, the decision on which connection to make “flips” as the contour level moves through the saddle value. This makes it unsatisfactory for an interactive application where we are smoothly changing the contour levels.

The main features are:

- correctness of the boundary points so consistency of solution from cell to cell;
- correctness of topology;
- discontinuity of visual representation with respect to changes in contour level (or changes in data).

17.5.2 Isosurfacing

As mentioned earlier, the classical MC algorithm has a naive approach to approximating the interpolant F^{3D} . Not only can holes between cells appear when two adjacent cells have certain configurations but also the interior of the cell is not represented in a topologically correct way.

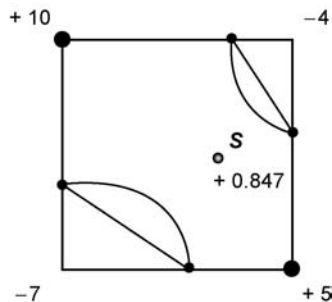


Figure 17.4. The asymptotic decider to solve ambiguity. F^{2D} of equation (17.1) in the region between the two contour sections is positive, which includes the saddle point S – the intersection of the two asymptotes $F_x^{2D} = 0$ and $F_y^{2D} = 0$, i.e., the point $(-b/c, -a/c)$.

The asymptotic decider described earlier for contouring is able to give positional continuity and correct topology on the faces of the cell. Of the 14 cases in [Figure 17.2](#), six involve at least one ambiguous face.

The problem now is to deal with the exact shape of the interior surface. [Natarajan \(1994\)](#) finds that further ambiguities can occur. For example, consider a cube where one pair of opposite vertices are positive, while the remaining six vertices are negative (this is configuration 4 in [Figure 17.2](#)). A simple triangulation will cut off the positive vertices with single triangular pieces, but these approximate a curved surface that bows out towards the centre of the cell. Imagine the data values all increasing uniformly so the zero isosurface pieces move towards each other – the simple-minded triangular approximation remains separated as two pieces, but the true trilinear surface pieces will come into contact with each other, and as that happens they merge into a single surface with a tunnel. This is an internal ambiguity, rather than the face ambiguity treated by the asymptotic decider. This is illustrated in [Figure 17.5](#) where we see the two situations (a) and (b) that can occur internally for a single vertex configuration. [Natarajan \(1994\)](#) shows that a key to identifying tunnels is the value at the body saddle point, which is the 3D equivalent of the saddle point whose value was exploited by [Nielson and Hamann \(1991\)](#) to determine face ambiguities in the asymptotic decider. The body saddle is located where the transition from one to two pieces occurs. [Figure 17.5a](#) shows the situation with two pieces whereas in [Figure 17.5b](#) the pieces have merged with a tunnel appearing.

A treatment of topological correctness in isosurfacing on rectilinear grids is presented by [Chernyaev \(1995\)](#). He identifies some 33 canonical configurations, covering both face and interior ambiguities. More recently, [Lopes and Brodlie \(2003\)](#) indicated that two cases can be removed as they were not canonical. [Figure 17.6](#) shows the 31 cases. The first number denotes the original MC case number as in [Figure 17.2](#); the second indicates the resolution of a face ambiguity; and the third the resolution of an interior ambiguity.

Other contributions to solve internal ambiguities in the MC algorithm are worthy of mention. In particular, [Cignoni et al. \(2000\)](#) extended the 256 cases to some 798 different cases, but only 88 of these were distinct configurations. They proposed an “extended look

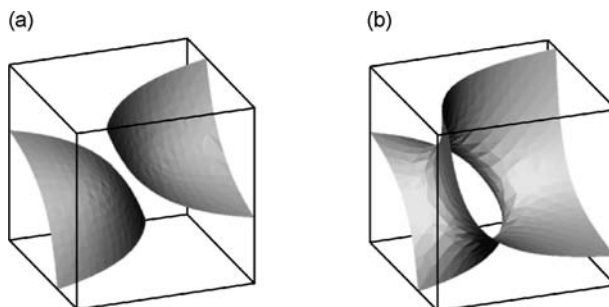


Figure 17.5. The two possible topologies (a) and (b) in MC configuration 4 (see [Figure 17.2](#)). The value of the body saddle point can discriminate the two situations.

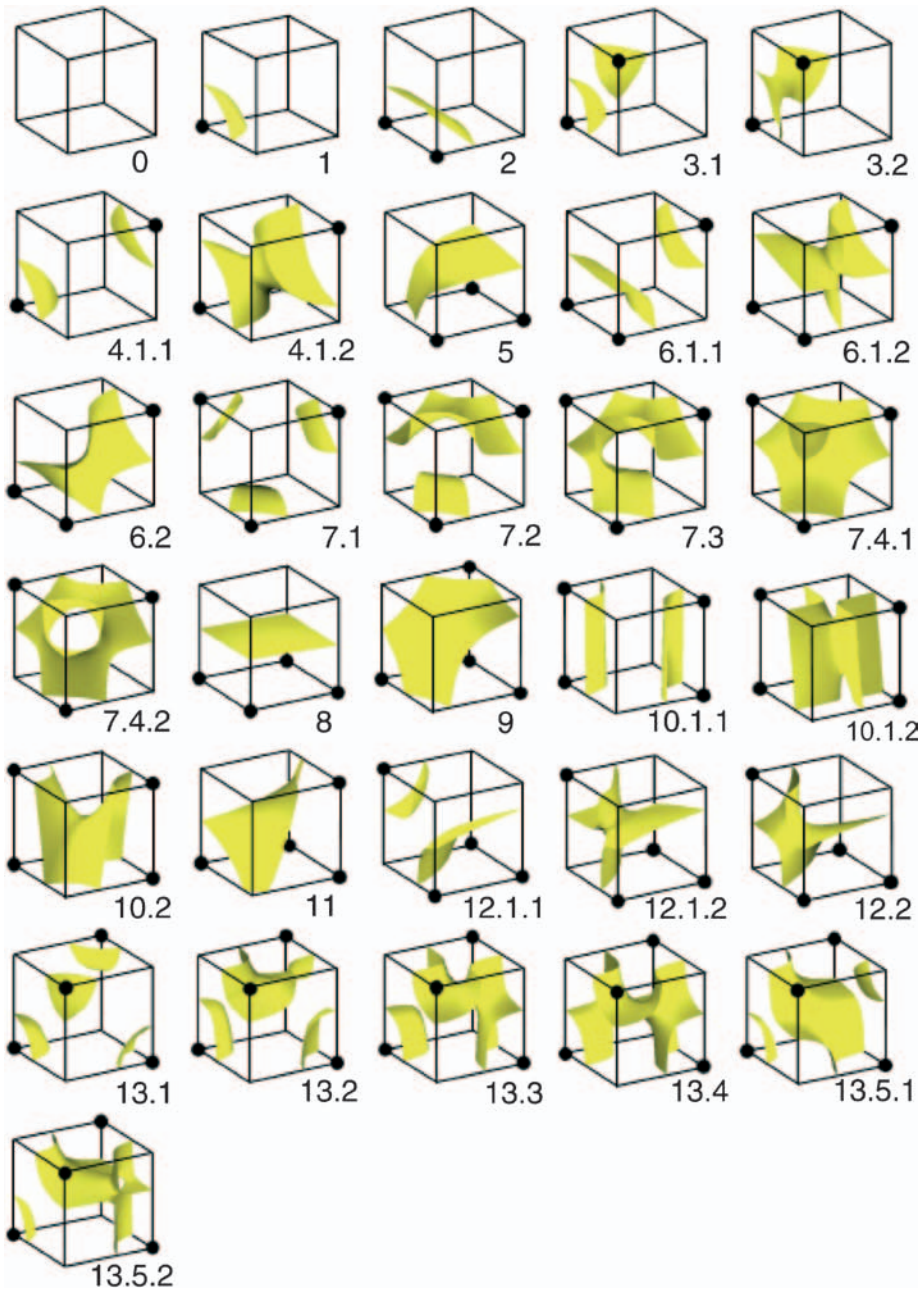


Figure 17.6. The 31 cases in the Marching Cubes algorithm. The first label number denotes the original MC case number as in [Figure 17.2](#); the second indicates the resolution of a face ambiguity (six ambiguous faces in the group of case 13, three in the 7, two in the 10 and 12, one in the 3 and 6, none in the others); and the third the resolution of an interior ambiguity.

up table" (ELUT), which aims to provide explicit triangulations for a large number of cases based on face saddle, and body saddle, values. Also, [Matveyev \(1994\)](#) has considered the behaviour of the interpolant along the cell diagonals in order to solve ambiguities.

Many of these efforts attempt to provide explicit triangulations of the surface. However, it is important to note that many fail to do this correctly, because they propose triangulations that involve triangles in the face of a cell, which should not be allowed. [van Gelder and Wilhelms \(1994\)](#) identified this important principle: in order to obtain a continuous triangulated surface (C_0) each polygon edge must belong to no more than two triangles. This forbids any triangles from lying in the face of a cell, since these will have an edge belonging to two triangles from that cell, and to at least one triangle from the adjoining cell. Generally, the choice of additional interior points needed to complete the triangulation is made in an *ad hoc* manner; in particular, they are typically not chosen so as to lie on the exact surface.

As with the asymptotic decider for contouring, these methods take decisions on how to represent the isosurface that are not continuous with respect to the threshold value. This makes them unsuitable for interactive work.

In conclusion, the main features of the MC algorithm with asymptotic decider and body saddle point test are:

- correctness of the boundary points so consistency of solution from cell to cell;
- correctness of topology both on the boundary and in the interior of the cell;
- the triangulation methodology is not very clear as additional points in the interior are sometimes needed;
- a discontinuity of visual representation with respect to changes in contour level (or changes in data).

17.6 Interactive Working

Cartographic maps are no longer seen as static entities, but rather as dynamic interfaces, responding to interaction by an investigator. This engagement between map and investigator is key to improving the thinking process, as geospatial data is explored ([MacEachren and Kraak, 2001](#)). The challenge for the geovisualization community is to provide techniques that are fast enough to be responsive (to allow this interaction), yet are of high integrity in the representation of the data so that correct inferences are drawn.

The methods introduced earlier indicate that we are now able to achieve topological correctness, with respect to the exact isolines of a bilinear, or exact isosurface of a trilinear. However, there remains a quality problem, in that the visual representation is discontinuous as the data changes. This is exactly the situation we would have in an interactive setting when the investigator alters the threshold value. Thus, we need to further develop methods that are robust when used in a setting where any of the following operations are carried out:

- the contour levels are altered, or the isosurface threshold is modified;
- the data values are continuously changed;
- the viewpoint is continuously changed, for example to zoom in on the contours or isosurface.

The solution is to take more care in the representation of the isoline, or isosurface, in the interior of the cell. At the same time, we cannot afford the solution to be inefficient, because speed is essential for effective interaction. Thus, we pay attention to the way contour lines are approximated as polylines, and isosurfaces as triangular meshes.

17.6.1 Contouring

For contouring, [Lopes and Brodlie \(1998\)](#) have modified the basic method with asymptotic decider by adding an extra point for each pair of intersection points, the so-called shoulder points. Each contour section is now drawn as a polyline from one intersection point to the shoulder point and then to the other intersection point. This gives a two-piece linear approximation to the conic arc.

The shoulder point is the point on the conic parallel to the chord joining the end-points. It is an optimal point to choose in forming this approximation, as it is the furthest point on the arc from the chord. This is shown in [Figure 17.7](#): P and Q are the end-points of the hyperbolic arc, and R is the shoulder point of the arc. R is quite easy to calculate as it lies on the line joining M , the mid-point of the chord PQ , and S , the saddle point of the bilinear interpolant.

The significant advantage of this simple extension is that the visual representation now moves smoothly as the contour level moves smoothly through the saddle value, compared with the visual “jump” that occurs in the unmodified method. [Figure 17.8](#) depicts this property. Moreover, the process of adding an extra point can be applied recursively as we zoom in on a contour line. Thus, we have an approach that is robust enough to be used in an interactive setting.

17.6.2 Isosurfacing

This work has been extended from contouring to isosurfacing ([Lopes and Brodlie, 2003](#); [Lopes, 1999](#)) with the objective of achieving a high quality internal representation by adding a minimal number of carefully chosen extra points, so that a minimal number of extra triangles are created. First, as in the classical approach, the intersection points of the isosurface with cell edges are calculated to form an initial polygonal outline of

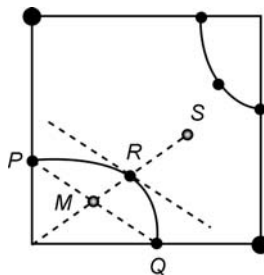


Figure 17.7. Shoulder point R as an extra point on the hyperbolic arc. It lies on the line MS as well as on the line parallel to PQ .

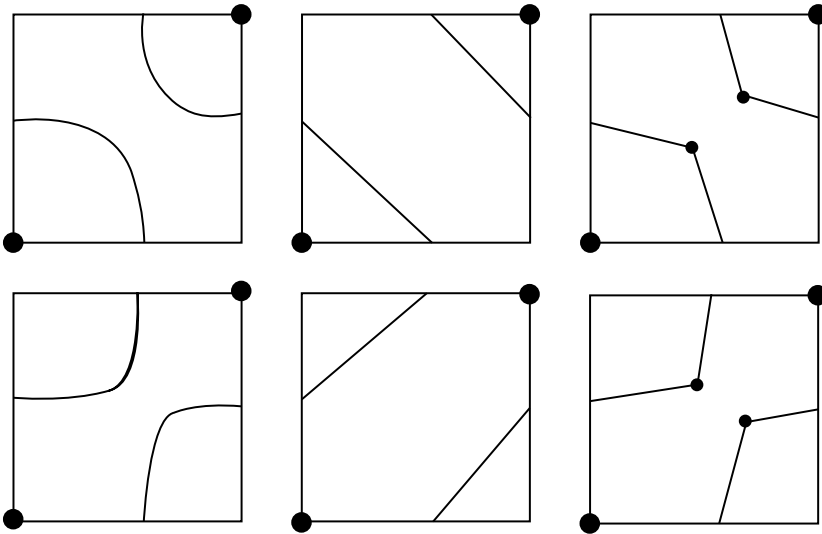


Figure 17.8. Shoulder points (right) help to move smoothly from top to bottom row in comparison to the basic method with asymptotic decider (middle) described in §17.5.

the isosurface. The edges of the polygon lie on the cell faces, and are approximations to the isocontour lines on the faces. This polygon is then extended by adding shoulder points, exactly as in the contouring. This improves the accuracy on the faces. In the interior there are two special classes of points, which help to define the internal behaviour of the interpolant. One class is called *bi-shoulder points*, which are 3D analogues of shoulder points; the other is called *inflection points*, which are generalizations of the body saddle point. With these points it is possible to generate efficient triangulations that correctly represent the interior topology, as well as increasing the accuracy. Figure 17.9 shows examples depicting how these classes of points help to nicely delineate the shape of the surface.

As with the contouring extension, we get a smooth transition of the visual representation with changes to the threshold level and to the data.

In conclusion, the main features of this method are:

- The solution is topologically correct and accurate.
- It is robust in the sense of being continuous with respect to changes in the data or isosurface level. As the threshold changes so that a different choice is made to resolve an ambiguity, so the representation does not display a visual discontinuity – the transition between the different states in the ambiguous cases is smooth.
- While achieving correctness and robustness, care has been taken to add as few additional points to the triangulation as possible (compared with the standard MC algorithm). The cost of computation of this superior representation is of

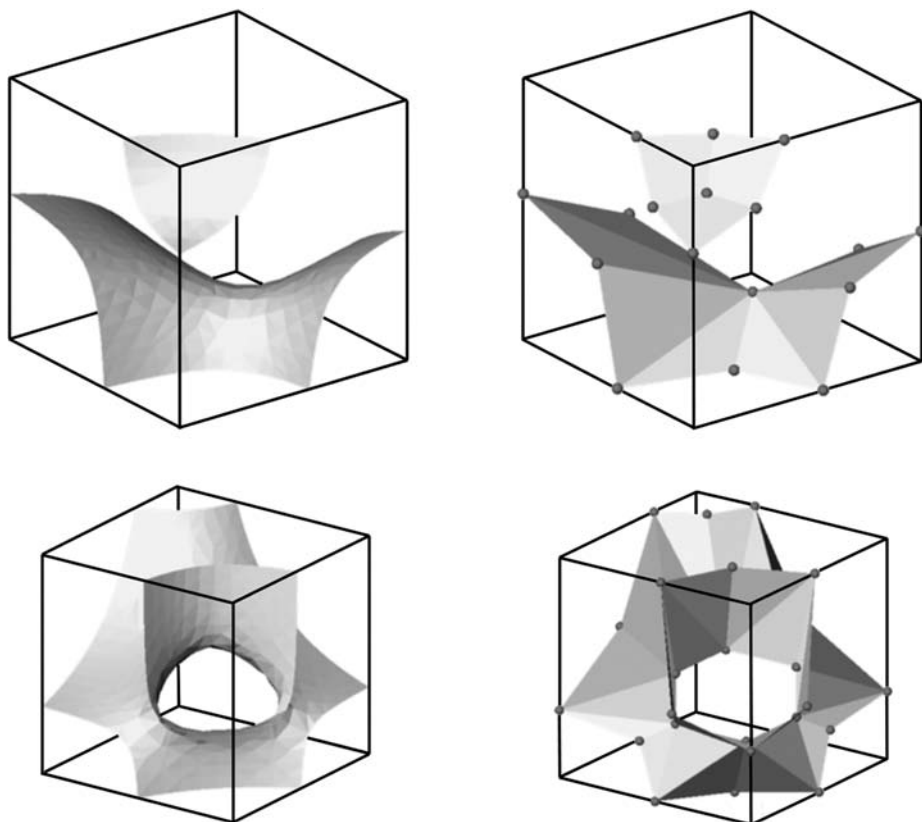


Figure 17.9. With the help of special classes of points – *bi-shoulder points* and/or *inflection points* – it is possible to generate efficient triangulations (right) to nicely delineate the shape of the surface (left) (Lopes, 1999).

course higher. But the major extra cost is indeed the increased number of triangles as computation of those points is quite simple.

- The extra points that are added, are chosen to lie on the isosurface of the trilinear interpolant, and to be optimal in terms of accuracy of polygonal representation.
- A straightforward set of triangulations is provided in contrast to look-up tables. The classification is ultimately based on the type of boundary polygon, i.e., the number of intersection points, number of faces intersected twice, and internal points. The four cases with no faces intersected twice have two sub-cases: either with or without an interior hole. This gives 14 canonical cases in total.

Figure 17.10 shows a set of isosurfaces distributed over a 3D grid of volatile organic compounds. An isosurface is by definition only a subset of the data, and it is a natural part of the visualization process to modify thresholds in order to sweep out areas of interest in the data (Shneiderman, 1996).

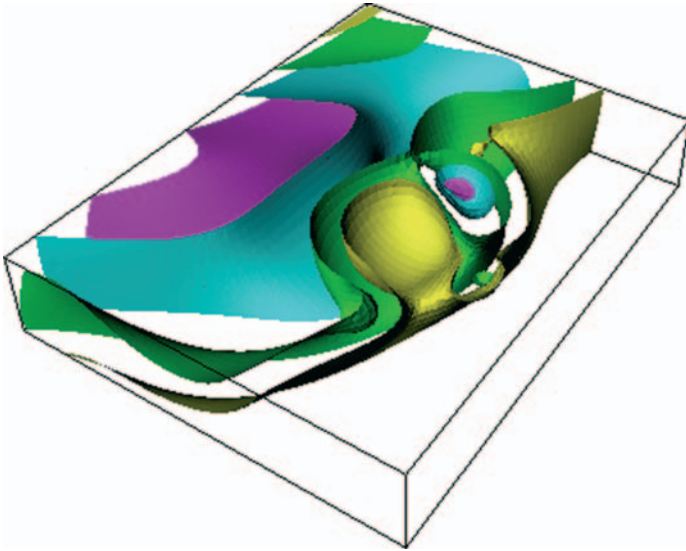


Figure 17.10. Set of isosurfaces depicting concentrations of volatile organic compounds in a 3D grid of size $60 \times 60 \times 50$. The data was kindly supplied by City University, using EarthVision(R) software by Dynamic Graphics, Inc., Alameda, California, USA. The visualization was created by ourselves, using the IRIS Explorer visualization system from NAG Ltd and in particular a module developed by one of the authors (Lopes) and now available through the IRIS Explorer Centre of Excellence (<http://www.comp.leeds.ac.uk/iecoe>).

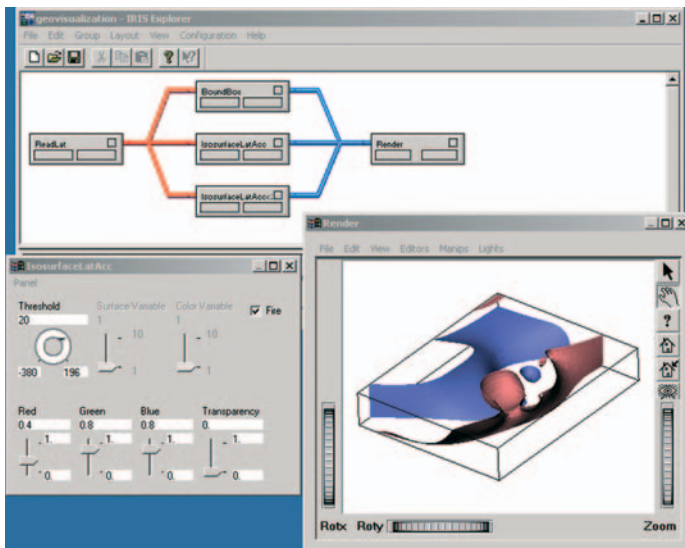


Figure 17.11. Interactive environment to depict concentrations of volatile organic compounds using the IRIS Explorer visualization system.

Figure 17.11 shows the sort of interactivity that is possible. In the interface panel shown, the user can alter the isosurface threshold by means of the dial. In a dataflow visualization system such as IRIS Explorer (as shown in the figure), a change of threshold triggers a recalculation of the isosurface and the resulting geometry is passed automatically to the rendering module for display. Thus, the picture changes immediately to reflect the dial change. This dynamic representation adds significant value to the geovisualization process. Notice that any algorithm that was not robust to threshold changes would be quite unsatisfactory in such an interactive environment.

17.7 Conclusions

This chapter has traced the history of contouring and isosurfacing techniques, showing how their evolution has been intertwined over the years. The evolution has taken us from initial approaches which aimed to satisfy a passive style of computing, where the graphic depiction is used as a presentation medium – to the modern era where interactivity allows the opportunity to explore the data in order to gain deeper insight through visualization.

Our final stage of the evolution has yielded an approach that offers efficiency, through economizing on either straight lines for contours or triangular pieces for isosurfacing, the latter mapping well on to fast triangle rendering hardware such as PC graphics boards supporting OpenGL. More than efficiency, the approach offers a visual continuity as the data is explored through variations in threshold level.

Acknowledgements

Adriano Lopes is grateful to the Department of Informatics of the Faculty of Science and Technology, New University of Lisbon, Portugal, and its associated research centre CITI, for their travel support. Figures 17.10 and 17.11 were produced in collaboration with City University using the EarthVision (R) software and data from Dynamic Graphics.

References

- Chernyaev, E., (1995) Marching cubes 33: construction of topologically correct isosurfaces, (CN/95-17). CERN.
- Cignoni, P., Ganovelli, F., Montani, C., and Scopigno, R., (2000) “Reconstruction of topologically correct and adaptive trilinear surfaces”, *Computers and Graphics*, 24(3), 399-418.
- Dayhoff, M., (1963) “A contour-map program for X-ray crystallography”, *Communications of the ACM*, 6(10), 620-622.
- Doi, A., and Koide, A., (1991) “An efficient method for triangulating equi-valued surfaces by using tetrahedral cells”, *IEICE Transactions on Communication Electronic Information Systems*, E-74(1), 214-224.
- Durst, M., (1988) “Letters: additional reference to ‘Marching Cubes’”, *Computer Graphics*, 22(2), 72-73.

- Fisher, H., (1963) *SYMAP*. USA: University of Harvard.
- Heap, B., (1972) *Algorithms for the Production of Contour Maps*, Report NAC-10, National Physics Laboratory, Teddington, UK.
- Lopes, A., and Brodlie, K., (1998) Accuracy in contour drawing, *Eurographics UK 98*, Leeds, UK, pp. 301–311.
- Lopes, A., and Brodlie, K., (2003) “Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing”, *IEEE Transactions on Visualization and Computer Graphics*, 9(1), 16-29.
- Lorensen, W., and Cline, H., (1987) “Marching Cubes: a high resolution 3D surface reconstruction algorithm”, *Computer Graphics*, 21(4), 163-169.
- MacEachren, A. M., and Kraak, M. J., (2001) “Research Challenges in geovisualization, cartography and geographic information science”, *Special Issue on Geovisualization*, 28(1), 3-12.
- Matveyev, S., (1994) “Approximation of isosurface in the marching cube: ambiguity problem”, In: Bergeron, R. D., and Kaufman, A., (eds.), *IEEE Visualization 94*. Washington, DC: IEEE Computer Society Press, pp. 288–292.
- Natarajan, B., (1994) “On generating topologically consistent isosurfaces from uniform samples”, *The Visual Computer*, 11, 52-62.
- Nielson, G., and Hamann, B., (1991) “The asymptotic decider: resolving the ambiguity in Marching Cubes”, In: Nielson, G., and Roseblum, L., (eds.), *IEEE Visualization 91*. San Diego, CA: IEEE Computer Society Press, pp. 83-90.
- Shneiderman, B. (1996), “The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations”, In *Proceedings IEEE Symposium on Visual Languages*, Washington: IEEE Computer Society Press, pp. 336-343.
- Sutcliffe, D., (1980) “Contouring over rectangular grids and skewed rectangular grids”, In: Brodlie, K., (ed.), *Mathematical Methods in Computer Graphics and Design*. London: Academic Press, pp. 39-62.
- Wyvill, G., McPheeters, C., and Wyvill, B., (1986) “Data structure for soft objects”, *The Visual Computer*, 2, 227-234.
- van Gelder, A., and Wilhelms, J., (1994) “Topological considerations in isosurface generation”, *ACM Transactions on Graphics*, 13(4), 337-375.
- Lopes, A., (1999). *Accuracy in Scientific Visualization*, School of Computer Studies, University of Leeds, p. 168.