I.5

COMPACT ISOCONTOURS FROM SAMPLED DATA

Doug Moore and Joe Warren
Rice University
Houston, Texas

Problem

Data in many fields, including medical imaging, seismology and meteorology, arrive as a set of measurements taken over the vertices of a large cubic grid. Techniques for producing a visual representation from a cube of data are important in these fields. Many common visualization techniques treat the data values as sample function values of a continuous function F, and generate, for some c, a piecewise planar approximation to $F(x, y, z) = c$, an isocontour of the function. One of the original *Graphics Gems,* "Defining Surfaces from Sampled Data," surveys several of the best-known techniques for generating isocontours from a data cube (Hall, 1990).

In this gem, we present an enhancement to all techniques of that type. The enhancement reduces the number of elements of any isocontour approximation and improves the shape of the elements as well. The first improvement typically reduces the size of a representation by about 50%, permitting faster redisplay and reducing memory requirements. The second results in better-quality pictures by avoiding the narrow elements that cause undesirable shading artifacts in many lighting models.

Cube-Based Contouring

Several authors have suggested roughly similar methods that create isocontours for visualization from a cubic data grid. These methods process the data separately on each cube, and use linear interpolation along the edges of a cube to compute a collection of points lying on the isocontour. In the Marching Cubes algorithm of Lorenson and Cline (Lorenson and Cline, 1987), these intersections are connected to form edges and triangles using a table lookup based on the signs of the values $F(x, y, z) - c$ at the vertices of the defining cube.

Unfortunately, that method does not guarantee a continuous contour, since adjacent cubes that share a face with mixed signs may be divided differently (Durst, 1988). Others have suggested an alternative method that disambiguates that case by sampling the function at the center of the ambiguous face (Wyvil *et al.*, 1986). We call methods like these, that compute the vertices of the resulting contour using linear interpolation along edges of the cubic mesh, *edge-based interpolation* methods.

Another problem with edge-based interpolation methods is that the surface meshes they produce can be highly irregular, even for simple trivariate data. These irregularities consist of tiny triangles, produced when the contour passes near a vertex of the cubic mesh, and narrow triangles, produced when the contour passes near an edge of the mesh. In our experience, such triangles can account for up to 50% of the triangles in some surface meshes. These badly shaped elements often degrade the performance of rendering algorithms and finite element analysis applied to the mesh while contributing little to the overall accuracy of the approximation.

Compact Cubes

The contribution of this gem is a general technique for eliminating the problem of nearly degenerate triangles from edge-based interpolation. The idea behind the technique is simple: When a vertex of the mesh lies near the surface, “bend” the mesh a little so that the vertex lies on the surface. The small triangles collapse into points, the narrow ones collapse into edges, and only big, well-shaped triangles are left. The rest of the gem outlines an implementation of this idea; a more detailed explanation is available (Moore and Warren, 1991).

Apply any edge-based interpolation algorithm to the data cube, and in the process, record for each vertex generated along an edge of a cube the point of the cubic grid nearer that vertex. We call that vertex a *satellite* of its nearest gridpoint. If the vertex lies at the midpoint of an edge,

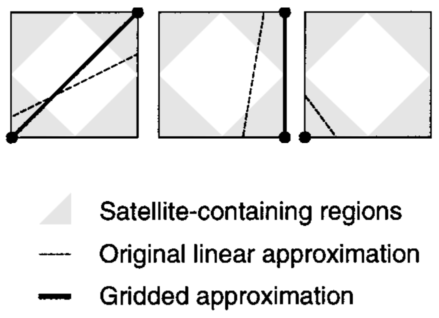

Figure 1. 2D case table for Compact Cubes.

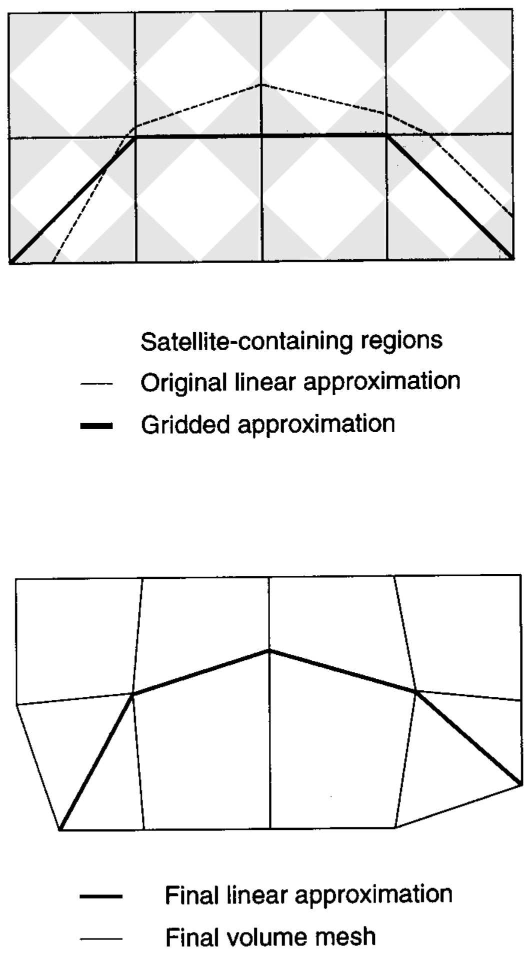
Satellite-containing regions
Original linear approximation
Gridded approximation
Final linear approximation
Final volume mesh

Figure 2. A 2-D example of Compact Cubes.

either endpoint of the edge may be used, as long as all other cubes sharing the edge use the same endpoint. When this phase of the algorithm has completed, you have a triangulation S of the isocontour and a grid point nearest each vertex of the triangulation.

To produce a new, smaller approximation to the isocontour, apply the following procedure:

```
for each triangle T in S do
    if the vertices of T are satellites of distinct gridpoints
        then produce a triangle connecting the gridpoints;
        else T collapses to a vertex or edge so ignore it;
endloop;
for each gridpoint g of the new triangulation do
    displace g to the average position of its satellites;
endloop;
```

The first step of the method defines the topology of a new mesh connecting points of the cubic grid. All the satellites in *S* of a particular

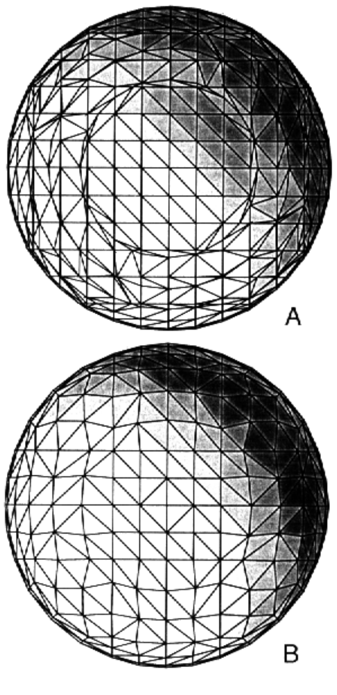

Figure 3. Two approximations to a sphere.

gridpoint are coalesced into a single vertex in the resulting mesh. Thus, small triangles that result when a gridpoint is "chopped off" are collapsed to the gridpoint. Narrow triangles produced when two vertices are very near the same gridpoint are collapsed to make the triangle an edge. Figure 1 illustrates this in two dimensions. This perspective shows that if the original surface mesh is continuous, then the mesh produced in the first step of the algorithm must also be continuous.

In the second step, the vertices of the gridded mesh are displaced to lie on or near the original isocontour. Since each new vertex position is chosen to be at the average position of a small cluster of points lying on the original contour, the new approximation usually diverges only slightly from the original contour.

Figure 2 illustrates this method applied to a two-dimensional mesh. The upper portion illustrates the result of the first step. The lower portion illustrates the output of the second step. The short edges in the upper

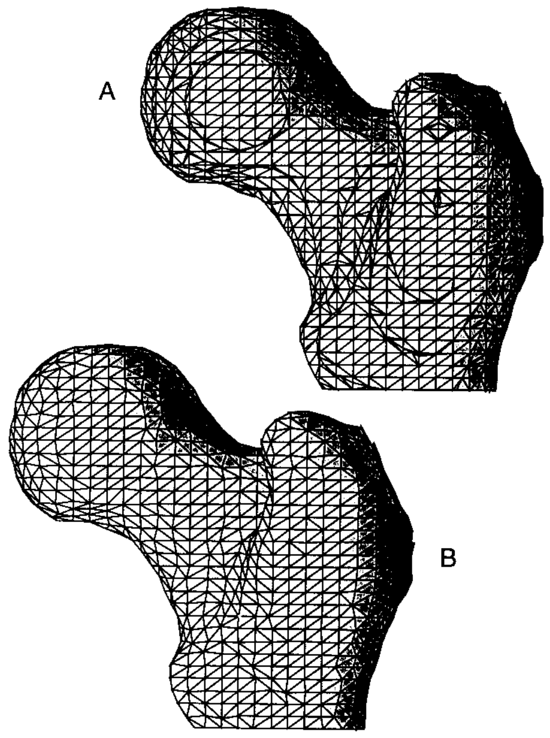

Figure 4. Two approximations to the head of a femur.

portion of the figure have been collapsed to form vertices in the lower portion.

In practice, the method works quite well, reducing the number of triangles by 40% to 60%. Figure 3 shows a sphere generated by Marching Cubes (A) and the same sphere after the application of Compact Cubes (B). Figure 4 shows a human femur, originally presented as CT data, as contoured by Marching Cubes (A) and by Compact Cubes (B). In each example, the number of triangles is reduced by using Compact Cubes, and the shape of the remaining triangles is measurably improved.

As described here, the contours produced by Compact Cubes may have several undesirable features. First, the boundary of the final contour may not lie on the boundary of the defining cubic mesh. Second, two disjoint sheets of the contour passing near a common gridpoint may be fused at that gridpoint. Moore and Warren (1991) describe simple modifications to Compact Cubes that solve each of these problems.

See also G1, 552; G1, 558; G2, 202.