Smooth Isosurface Approximation [*]

Cindy M. Grimm
cmg@cs.brown.edu

John F. Hughes
jfh@cs.brown.edu

March 7, 1996

Abstract

We present a method for approximating an isosurface with a smooth parametric representation. From the isosurface we first produce a *patch mesh*, a description of how many surface patches there are and how they are connected. We then create a smooth surface from the patch mesh.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling, Curve, Surface, Solid, and Object Representations, Splines

Additional Keywords: Isosurface, Arbitrary Topology.

1 Introduction

We present a method for approximating an isosurface with a smooth parametric model. Modeling shapes via an isosurface of a 3 dimensional data set is an efficient and natural method for specifying the topology and basic geometry of an object [BS91] [GH91] [Mur91] [GCB86] but it can be difficult to control fine surface detail. Parametric surfaces (e.g., B-splines) provide good detail control, but creating topologically arbitrary surfaces with them is difficult. We propose a modeling paradigm wherein an artist creates a sketch of an object using a 3D paint program such as Sculpt [GH91] from which a smooth parametric representation of an isosurface is constructed, at some level of fidelity. Further adjustment of the surface detail is accomplished using the parametric representation.

In order to produce a parametric model from an isosurface we must first determine a *patch mesh*. We formally define a patch mesh in Section 4; informally, a patch mesh is a

[*]This work was supported in part by grants from NSF, DARPA, IBM, NCR, Sun Microsystems, DEC, HP, and ONR grant N00014-91-J-4052 ARPA order 8225.

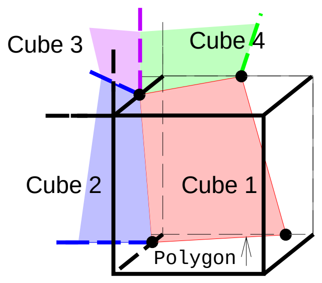

Figure 1: The polygon generated by the isosurface passing through a cube of the marching cubes algorithm.

collection of vertices, edges, and faces which describe how many surface patches there are and how they are connected (e.g., one patch per face). The quality and ease of the data fitting process is determined in part by how suitable the patch mesh is. If the mesh is regular, with the amount of curvature in the data spread evenly among the patches, then the fitting process is much simpler and less prone to inaccuracy. In this paper we present a method for generating a patch mesh which meets several criteria for suitability. We then give an example of fitting a parametric surface [Gri96] to the data using the patch mesh.

2 Previous work

Implicit surfaces are typically defined in two different ways; as functions or implicit offset surfaces (also known as "blobby models") [Mur91] [GCB86] [BS91] and as 3D discretely sampled data sets. The latter are usually constructed from real world data (e.g., medical resonance imaging scans) but in [GH91] the data set is created using a 3D version of a paint brush. Blobby models produce very smooth surfaces but their geometry is somewhat limited by the shapes of the underlying skeletons. Sculpting with a 3D paint brush produces very free-form objects but the surfaces are "warbly", in part due to the difficulty of controlling and reading a 3D input device.

Very little work exists on converting any but algebraic implicit surfaces to a parametric representation. If the surface is tessellated [Blo88] [LC87] then the problem can be viewed as one of fitting a smooth surface to a set of data points. In [HDD+94] a technique is presented that produces a C^1 surface from unorganized points, but this surface does not have a parametric representation. A different approach is to use the tessellation as a patch mesh, with one patch for each polygon in the tessellation. This produces a huge number of patches; one possible approach to reducing the number of triangles is *triangle decimation* [WSL92]. Unfortunately, triangle decimation is only concerned with local changes between triangles that are relatively flat with respect to each other and so cannot reduce a cylindrical-shaped object to one or two triangles, although such a shape can be covered by one or two patches.

3 The basic algorithm

The algorithm presented here begins with a tessellation and reduces it by exploiting topological properties of the tessellation, as opposed to geometrical properties such as "are the normals of these two triangles nearly parallel". The steps of the algorithm are as follows:

1. Produce a tessellation of the isosurface.
2. From the tessellation produce a patch mesh satisfying certain properties.
3. Reduce the number of patches in the patch mesh while maintaining these properties.
4. Find locations for the vertices of the patch mesh so that the patches of the mesh are fairly evenly distributed on the isosurface.
5. Fit patches to the isosurface using the patch mesh to determine the size and location of the patches.

We chose to produce a patch mesh with a valid topology and then simplify it because it is easier to maintain topological properties then it is to achieve them initially.

The following section describes the patch mesh and its properties. Sections 5– 8 detail steps 2-4 in the algorithm.

4 The patch mesh

This definition of a patch mesh is an informal simplification of an oriented 2-complex [CF67][1]. A patch mesh is a set of vertices $\mathcal{V}$, edges $\mathcal{E}$, and faces $\mathcal{F}$ where:

1. $\mathcal{V}$ is a finite set of vertices $v_1, \ldots, v_n$.
2. $\mathcal{E}$ is a set of edges where each edge $e = \{v_i, v_j\}$ is an unordered set of two distinct vertices.
3. $\mathcal{F}$ is a set of faces where each face $f = [v_{j_0}, \ldots, v_{j_n}]$ is an ordered list of at least 3 distinct vertices. The edges of f (e.g., $\{v_{j_i}, v_{j_{i+1}}\}$) must be in $\mathcal{E}$.

[1]We are essentially defining an abstract 2 dimensional surface.

4. Every vertex is in some face.

5. Every edge is an edge of some face.

6. Faces that meet do so in one of two ways:

   (a) They share exactly one vertex.

   (b) They share exactly one edge (and the two vertices of the edge).

7. The faces are orientable, i.e., if $e = \{v_j, v_k\} \in \mathcal{E}$ then there is at most one face containing the sequence $\ldots, v_j, v_k, \ldots$ and at most one face containing the sequence $\ldots, v_k, v_j, \ldots$

In addition to these properties we require our patch mesh to satisfy:

8. Each face has 3, 4, 5, or 6 edges.

9. Each vertex has exactly four distinct faces adjacent to it (i.e., vertices have valence four).

The first 7 properties ensure that the patch mesh is an oriented surface (what is usually called a polyhedron). Property 8 reduces the number of types of patches we need to consider but is not a necessary condition. Property 9 provides the topological structure necessary for several parametric surface models. For example, both [LD89] and [Gri96] require a patch mesh with four faces incident upon a vertex. Tensor-product B-spline surfaces require 4-sided polygons; the dual of the patch mesh described here is a mesh with 4-sided faces.

5 The initial mesh

In this section we detail the creation of a patch mesh satisfying the above properties from a tessellation. The tessellation we use is the marching cubes algorithm [LC87], here-after referred to as the MCA. We use the MCA because the tessellation it produces results in a patch mesh that satisfies the properties listed above.

We consider a *polygon* of the MCA to be the polygon formed by intersecting the iso-surface with a single cube of the MCA. The number of sides of the polygon is determined by the number of cube faces the isosurface intersects. Figure 1 shows a cube of the MCA and the polygon produced from it, as well as some of the neighboring cubes and adjacent polygons in the tessellation. Some points to notice about the tessellation:

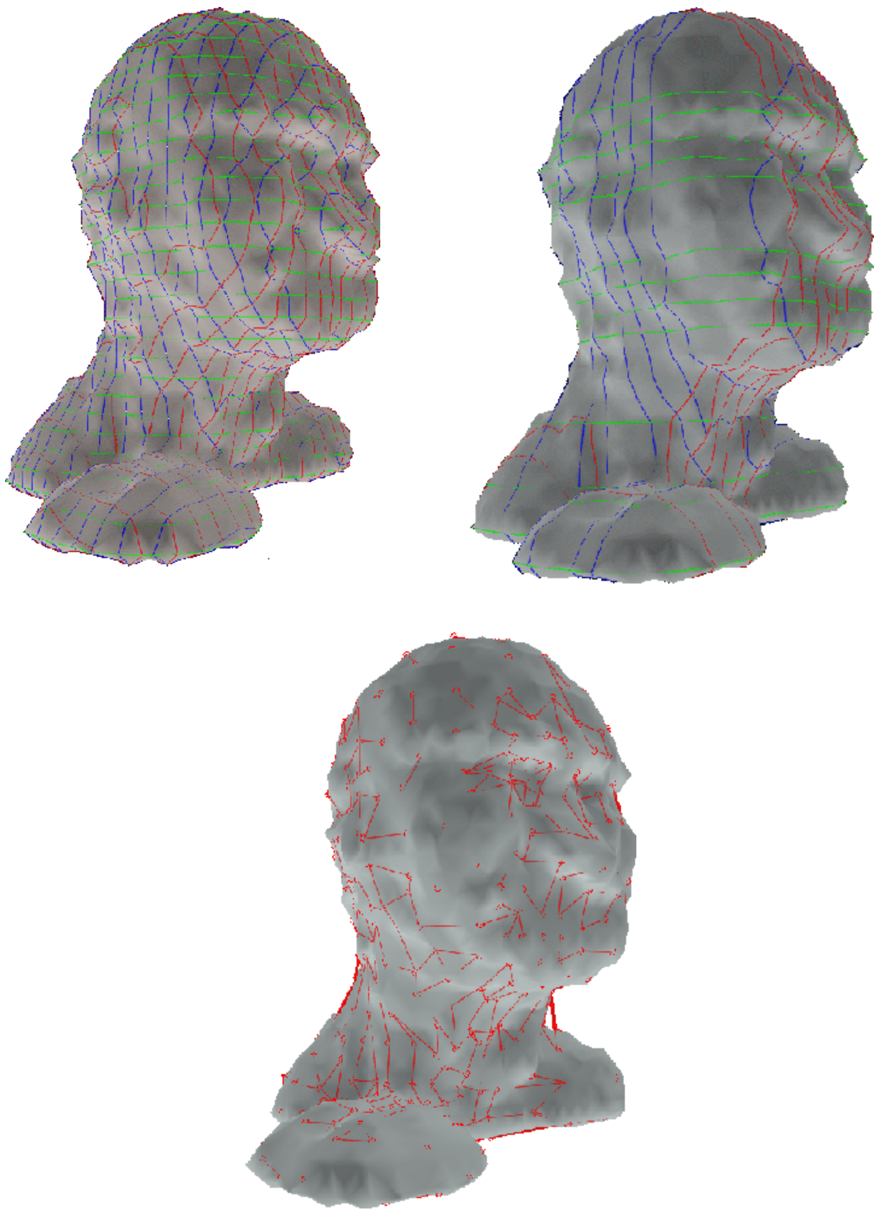

Figure 2: Top left: All isothetic chains. Top right: A minimum set of isothetic chains. Bottom: The adjusted vertex locations.

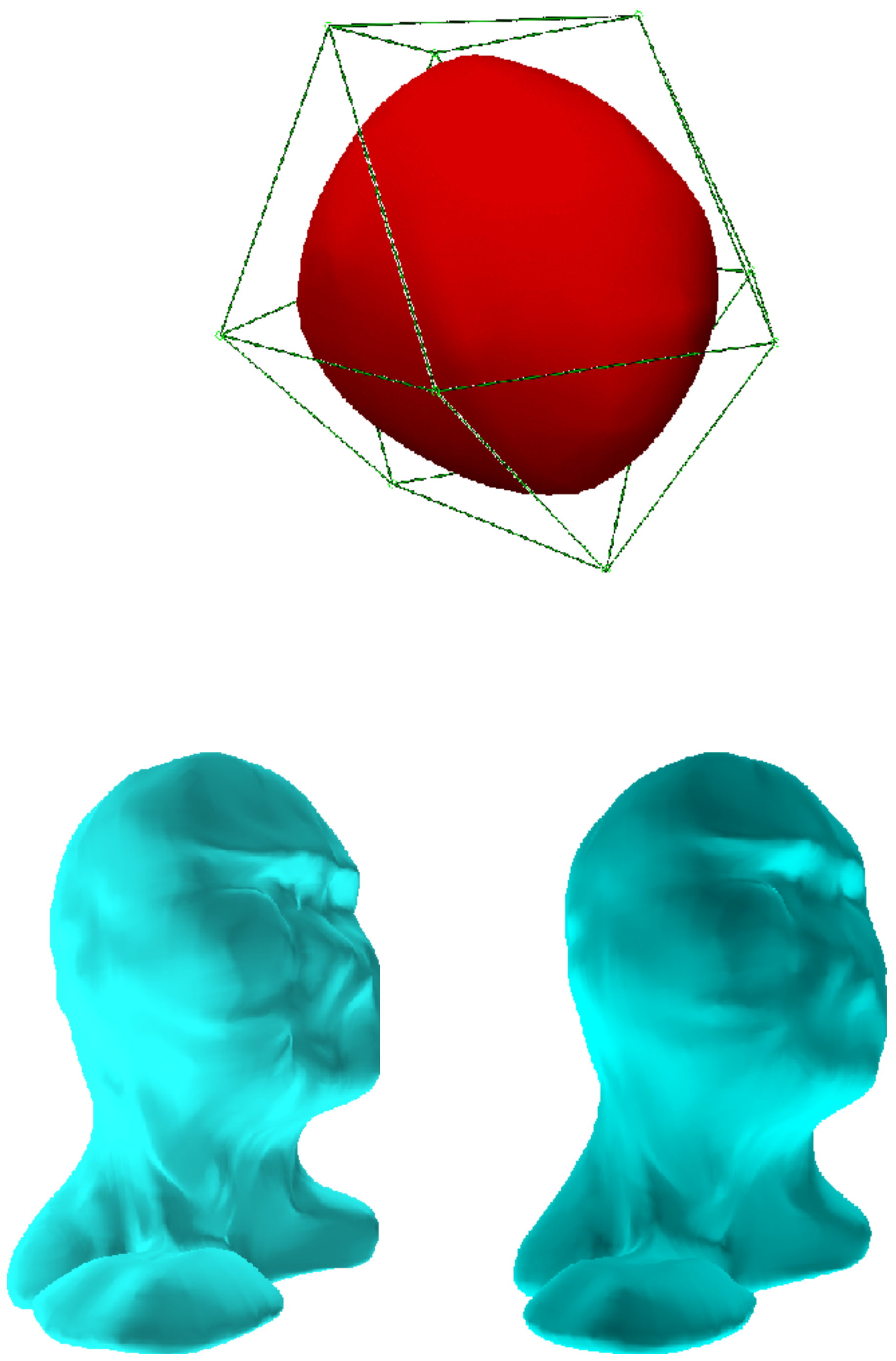

Figure 3: Top: An example of a spherical mesh P and the resulting surface Q. Bottom left: The surface Q resulting from a mesh P which is identical to the patch mesh. Bottom right: The surface after Q had been moved closer to the isosurface.

- Each vertex of the polygon lies on an edge of the cube. This edge is shared with exactly 3 other cubes [2].

- Each edge of the polygon lies within a face of a cube and hence is in a plane parallel to either the XY, YZ, or XZ plane.

- Each polygon has 3, 4, 5, or 6 sides.

Because the isosurface is continuous and without boundary, every vertex of the tessellation will have exactly four polygons adjacent to it. [3] To produce an initial patch mesh we create one face in the mesh for each polygon generated by the MCA, i.e,

- For every point where the isosurface crosses an edge of the MCA grid we have a vertex in the patch mesh, i.e., we create a vertex in the patch mesh for every vertex in the tessellation.

- For every edge in the tessellation we create an edge in the patch mesh.

- For every polygon we create a face in the patch mesh.

We need to verify that this patch mesh satisfies the properties given in Section 4. Because the isosurface is an oriented, continuous surface the tessellation satisfies properties 1–7; since the structure of the patch mesh is identical to the tessellation the patch mesh is an orientable 2-complex. The polygons of the tessellation have 3, 4, 5, or 6 sides, so the faces of the patch mesh satisfy property 8. The last property is satisfied because every vertex of the tessellation has four polygons adjacent to it.

6 Isothetic chains: reducing the mesh

To reduce the patch mesh we replace combinations of faces in the patch mesh with a single face. If we simply take a pair of faces and combine them the resulting patch mesh will no longer satisfy the given properties (in particular, the last property). There is, however, a way to join a set of pairs of faces and still maintain a valid mesh.

The patch mesh constructed above has an additional property – each edge lies in one of the planes parallel to the XY, YZ, or XZ plane. If we examine one of these planes and all of the edges lying in that plane, we see that the edges form closed loops, i.e. *chains* [Lef49]. We call these loops *isothetic chains* because the loop lies in a single plane. Figure 2 shows a surface and the isothetic chains of the patch mesh.

[2] If a vertex lies exactly on the corner of the cube we perturb the data slightly.

[3] We assume that the entire isosurface is contained in the MCA grid.

We reduce the patch mesh by removing an isothetic chain and joining the pairs of faces which share an edge of the isothetic chain. In Figure 4 we see a patch mesh with 5 isothetic chains (left). The middle chain is removed and the four pairs of faces which share an edge of that chain are joined (right). For example, face A and face B are joined together to become the new face $A+B$.

In this example the new patch mesh is still valid; this is not true for removing an isothetic chain in general. To reduce the patch mesh we identify which chains can be removed, pick one and remove it, then repeat the process until there are no more "free" chains. Note that there may be many "minimal" subsets of the initial chain set, in the sense that no more chains can legally be removed. We seek only to find some minimal set. In Appendix A we provide an heuristic for determining an order in which to remove chains.

To understand which chains can be safely removed, we need to know when removing a chain will result in a mesh that does not satisfy the properties in Section 4. When a chain is removed the following happens (refer to Figure 5):

- Let $e_1 = \{v_0, v\}$ and $e_2 = \{v, v_1\}$ be two edges such that v is in the chain and v_0 and v_1 are not in the chain. Note that the edges e_1 and e_2 must belong to the same chain. These two edges are deleted and replaced with the edge $\{v_0, v_1\}$. The new edge is in the same chain as the original two edges.

- Let f_0 and f_1 be two faces which share an edge $e = \{v, v'\}$ of the chain. The two faces are deleted and replaced with a new face containing the vertices of the original faces (except v and v'). If f_0 has n sides and f_1 has m sides then f will have $n+m-4$ sides.

When does this operation result in an invalid mesh? If $v_0 = v_1$ in the first step then the new "edge" is not an edge. Because v_0 and v_1 both belong to the same chain this only happens if that chain has no other chains crossing it. If f_0 and f_1 both border on another face f' then the new face f will border on f' twice. Note that the vertices will maintain their four faces at a vertex property – we either delete the vertex and all its edges, or delete an edge from a vertex and replace it with a new edge. The operations that fail to maintain a valid mesh are summarized below:

- Removing a chain that results in a face with less then 3 or more then 6 sides.

- Removing a chain that leaves a chain without any other chains crossing it (a chain without another chain crossing it is an edge whose endpoints are the same vertex).

- Removing a chain that results in an annular region (see Figure 6).

- Removing a chain that results in a face which shares two different edges with another face.

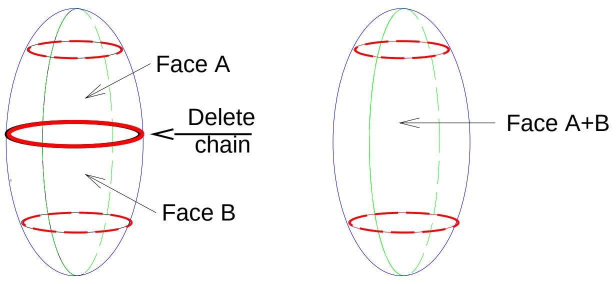

Figure 4: Removing the middle chain.

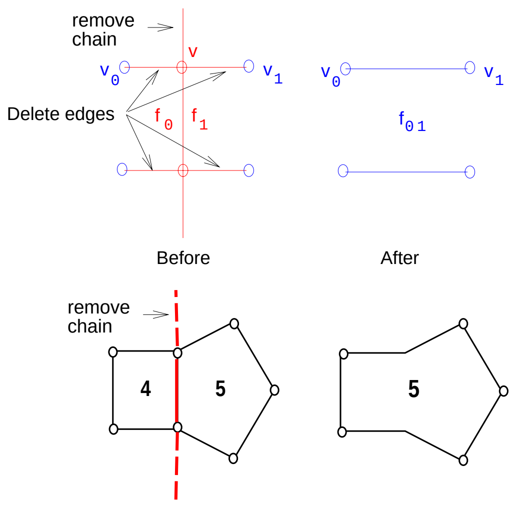

Figure 5: Left: A chain to be deleted and two chains which cross it. The items to be deleted are in red. Right: The two new edges and face. The two faces are joined together by removing the edge between them and combining the adjacent edges to form a single edge.

Figure 2 shows an example of a minimal set of chains.

7 Adjusting the patch mesh

We now have a reduced patch mesh which can be associated with the isosurface by placing the vertices at their original locations (each vertex in the patch mesh was initially associated with a vertex of the tessellation, and the reduction process does not introduce new vertices). We now adjust the locations of the vertices to more evenly distribute the patches across the isosurface.

We do this by growing regions r_v around each vertex v of the patch mesh until these regions cover the isosurface. (This is similar in concept to creating the Vornoii regions of each vertex, except we use "number of edges of polygons" as a metric instead of Euclidean distance.) We then move the vertex v to the geometric center of the region r_v. This process is repeated until the number of vertices that move falls below a threshold [4].

The algorithm for marking and moving the vertices is as follows. Let T be the tessellation and P be the patch mesh:

```
Mark all vertices of $T$ with **unmarked**
For all vertices $v_i$ of $T$
     If $v_i = p_j$, $p_j \in P$ set Mark($v_i$) to $j$
While **unmarked** vertices in $T$ exist
     For each marked vertex $v_i \in T$
          Mark all **unmarked** neighboring vertices of $v_i$ with Mark($v_i$)
For each mark $j$ with $p_j \in P$
     Let $J$ be all the vertices marked $j$
     Let $v_c$ be the geometrical average of the vertices in $J$
     Set $p_j$ to be the vertex $v \in J$ closest to $v_c$
```

This process distributes the vertices of the patch mesh more evenly across the isosurface, without affecting the abstract topology of the patch mesh. It is possible that two vertices might "cross over" each other; an additional check can be added to detect this. It has not been a problem in practice. Figure 2 shows an adjusted mesh.

8 Fitting a surface to the patch mesh

[4]It is possible for vertices to oscillate between two positions; we therefore stop when the number of vertices that move stabilizes.

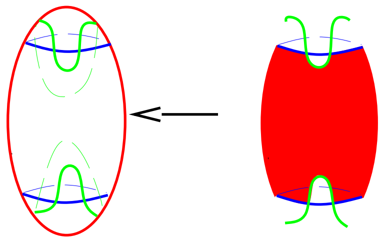

Figure 6: An example of an annular face created by removing the indicated chain.

We now show how to fit the surface described in [Gri96] to the isosurface using the patch mesh to determine the number and location of the patches. Briefly, the surface in [Gri96] is an extension of B-splines to a topologically arbitrary domain. The surface is specified by a polyhedral mesh M (similar in concept to a control point mesh) whose faces meet four at a vertex. The equation of the surface is given by

$$Q(x) = \sum_{s \in S} G_s B_s(x)$$

where x lies on a *manifold* [Spi70], the set S is a finite set, the B_s are basis functions [BBB87], and the G_s are control points. The locations of the G_s are determined by the subdivision surface [CC78] of M. Figure 3 shows an example polyhedron M and the resulting surface Q.

To fit a surface Q to the isosurface we first construct a polyhedron M which has the same topology as the patch mesh and whose vertices are placed at the vertex locations of the patch mesh (see Fig. 3). We then calculate how to move the vertices of M to bring Q closer to the isosurface. Because the basis functions are local and the control points are linear combinations of some number of the vertices of M, it suffices to do the following.

```
Error = $\infty$
Do while Error decreases
     Error = 0
     For each vertex $v \in M$
          vec = 0
          For each control point $v$ influences
               pt = closest point on isosurface to control point
               vec += pt - control point
               Error += length(vec)
          Move $v$ by vec / (number of control points)
     Recalculate surface
```

This moves Q closer to the isosurface (see Figure 3). Because we are interested in a smooth approximation to the isosurface, we do not require that points of Q exactly lie on the isosurface.

9 Adjusting the level of fit

If desired, we can produce smaller patches by removing fewer chains from the patch mesh. If we associate with each face in the patch mesh a *total absolute Gaussian curvature* [MP77], we can require that the aggregate absolute Gaussian curvature of any patch not exceed a certain threshold. This prevents the removal of chains which pass through areas of high curvature.

10 Remarks

The algorithm presented here produces a parametric surface which is an approximation of the isosurface. This process is "lossy" in the sense that we do not require the surface Q to exactly meet the isosurface (although the *topology* of Q is identical to that of the isosurface). This is desirable for this application because we are trying to "smooth out" the data produced from Sculpt. This loss of data, however, makes this technique unsuitable for visualization of an isosurface if the user wants to see the *exact* data.

11 Acknowledgments

Many thanks to Dan Robbins and Ken Herndon for the Sculpt models, and Tinsley Galyean for the Sculpt program.

A Order of removal of chains

We rank the chains according to the following criteria:

- Removing the chain reduces the number of 6-sided regions.
- The number of edges in the chain.
- We have not removed many chains lying in planes parallel to this one.

We prefer chains that remove 6-sided regions and are medium in length (short chains tend to capture information near the extrema of the isosurface). We also prefer to remove an equal number of chains in each of the three planes.

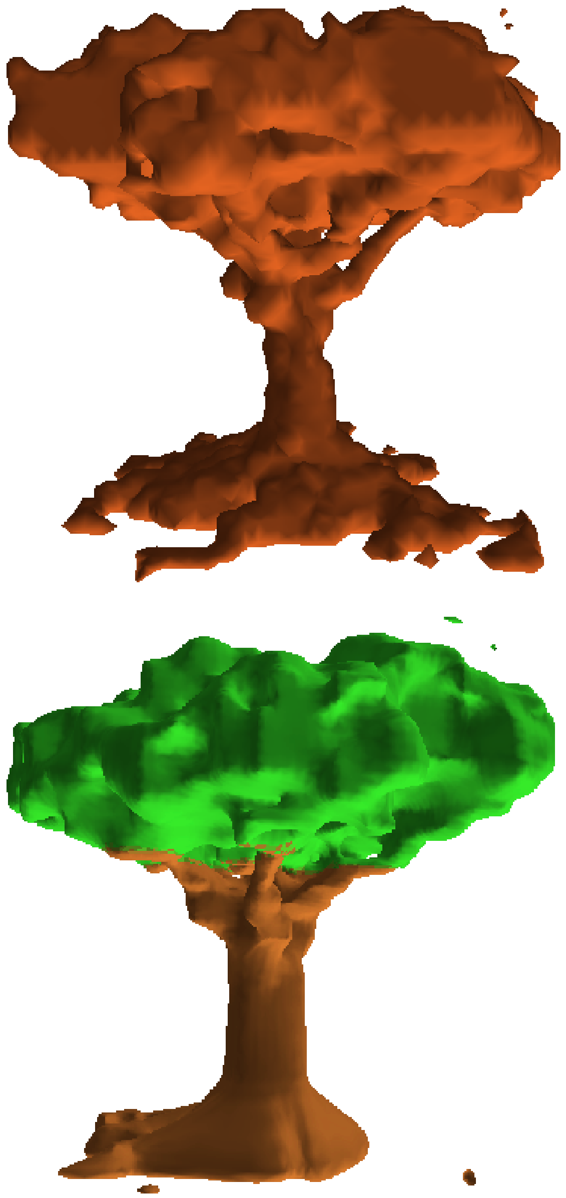

Figure 7: Sculpted tree. Top: Marching cubes surface. Bottom: Parametric surface.

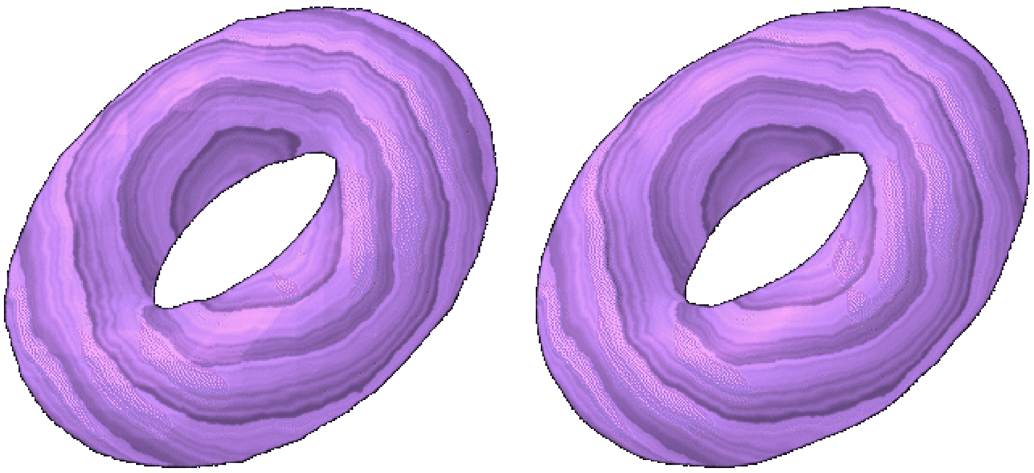

Figure 8: Generated two-holed torus. Left: Marching cubes surface. Right: Parametric surface.

References

[BBB87] R. Bartels, J. Beatty, and B. Barsky. *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling.* Morgan Kaufmann, first edition, 1987.

[Blo88] J. Bloomenthal. Polygonization of Implicit Surfaces. *Computer Aided Geometric Design*, 5(4):341–356, November 1988.

[BS91] J. Bloomenthal and K. Shoemaker. Convolution Surfaces. *Computer Graphics*, 25(4):251–256, July 1991. Proceedings of SIGGRAPH '91.

[CC78] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer Aided Design*, 10(6):350–355, November 1978.

[CF67] G. Cooke and L. Finney. *Homology of Cell Complexes.* Princeton University Press, Princeton, New Jersey, 1967.

[GCB86] Wyvill G., McPheeters C., and Wyvill B. Data Structures for Soft Objects. *Visual computer*, 2(4):227–234, August 1986.

[GH91] T. Galyean and J. Hughes. Sculpting: An Interactive Volumetric Modeling Technique. *Computer Graphics*, 25(4):267–274, July 1991. Proceedings of SIGGRAPH '91 (Las Vegas, Nevada, July 29 - August 2, 1991).

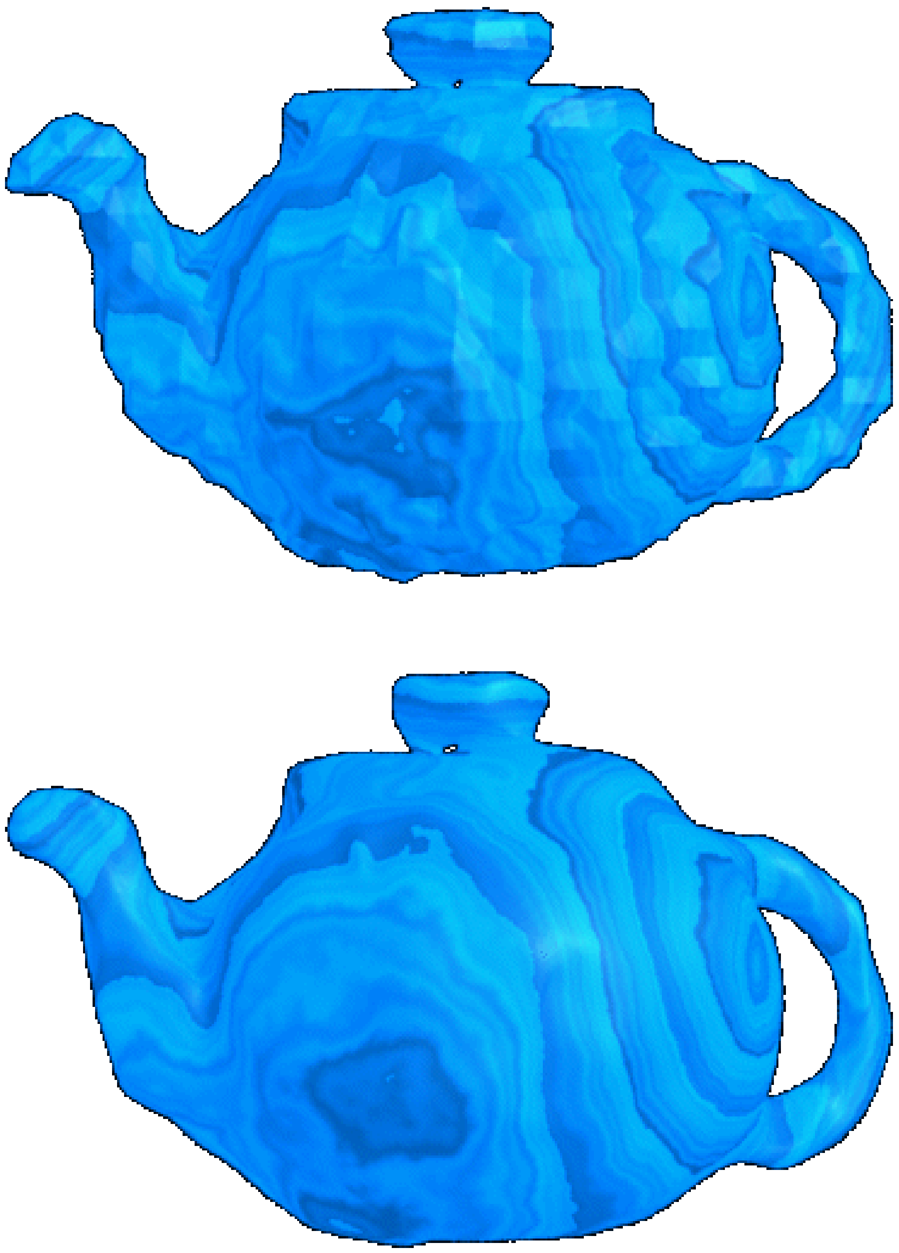

Figure 9: Sculpted teapot. Top: Marching cubes surface. Bottom: Parametric surface.

[Gri96] Cindy Grimm. *Modeling Surfaces of Arbitrary Topology using Manifolds.* PhD thesis, Brown University, 1996.

[HDD$^+$94] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise Smooth Surface Reconstruction. *Computer Graphics*, 28(2):295–305, July 1994. Proceedings of SIGGRAPH '94.

[LC87] W.E. Lorenson and H.E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *Computer Graphics*, 21(4):163–169, July 1987.

[LD89] Charles Loop and Tony DeRose. A Multisided Generalization of Bézier Surfaces. *ACM TOG*, 8(3):204–234, July 1989.

[Lef49] S. Lefschetz. *Introduction to Topology.* Princeton University Press, Princeton, New Jersey, 1949.

[MP77] R. Millman and G. Parker. *Elments of Differential Geometry.* Prentice-Hall Inc., first edition, 1977.

[Mur91] S. Muraki. Volumetric Shape Description of Range Data using "Blobby Model". *Computer Graphics*, 25(4):227–235, July 1991. Proceedings of SIGGRAPH '91 (Las Vegas, Nevada, July 29 - August 2, 1991).

[Spi70] M. Spivak. *Differential Geometry Volume 1.* Publish or Perish Inc., first edition, 1970.

[WSL92] J.A. Zarge W.J. Schroeder and W.E. Lorenson. Decimation of Triangle Meshes. *Computer Graphics*, 26(2):65–70, July 1992.