Warping Cubes: Better Triangles from Marching Cubes

LeeAnn Tzeng [a,1]

[a] *Dartmouth College, 6211 Sudikoff Laboratory, Hanover, NH 03755*

Key words: triangulation, isosurface reconstruction, Marching Cubes

1. Introduction

The Marching Cubes algorithm [6] for extracting a triangulation of an isosurface of a function defined over a three-dimensional space is famous throughout graphics. However, the triangles created by Marching Cubes are often quite skinny. These thin triangles can create artifacts in computer rendering, and make the surface triangulation unsuitable for volume rendering. To avoid these artifacts in rendering, thin triangles need to be eliminated from the triangulation created by the Marching Cubes algorithm.

The most basic form of Marching Cubes places a grid over the region containing the function whose isosurface we wish to extract. The function is evaluated at the vertices of this grid (I will call these the "corners" of the cubes to avoid confusion with the vertices of the resulting triangulation), and each corner is labelled as being positive or negative. Based on the pattern of labels, each cube is triangulated via a lookup table (for speed). The result is a triangulation that has vertices on the edges of the cubes where the isosurface intersects the edge. [8]

2. Warping and Collapsing

My algorithm modifies Marching Cubes by adding a "warping" phase to the original algorithm. This phase might better be called the "collapsing" phase, as will become clear shortly (though it could be described as "warping" the triangulation). The warping here should not be confused with Oct-tree Warping, which is visually similar but actually very different [7]. This process of warping or collapsing is based on the observation that skinny triangles take one of two forms. The first kind of skinny triangle is very tall, with a tiny base. This kind has one sharp angle at the top. The other kind of skinny triangle has an extremely short altitude and a very long base. (Fig. 1) My approach attacks each of these problems directly.

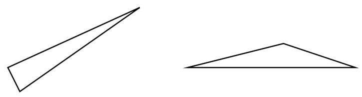

Fig. 1. Two kinds of skinny triangles.

2.1. *Short edges*

A short edge occurs when the isosurface intersects a cube very close to the cube's corner. Each endpoint of the short edge lies on one of the cube's edges close to that corner. A naive first thought is to pull the corner away from the short edge, creating a distorted cube that lengthens the short edge. However, even in 2D, it is easy to see how this might cause additional problems in adjacent cubes. (Fig. 2)

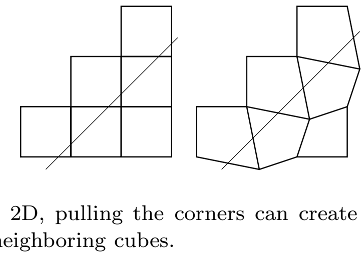

Fig. 2. In 2D, pulling the corners can create more short edges in neighboring cubes.

Instead, I choose to move the corner *onto* the short edge. This removes the short edge altogether

---

Email address: `ltzeng@cs.dartmouth.edu` (LeeAnn Tzeng).

[1] Supported on a National Science Foundation Graduate Fellowship.

by replacing its two endpoints, each a vertex in the triangulation, with a single vertex. This *warps* the cubes in a more unilaterally beneficial way: I eliminate a short edge but do not shorten any other edges. In fact, more often than not, the remaining edges are actually lengthened. One can easily see how this technique might also be seen as *collapsing* the short edge into a single vertex. (Fig. 3 and 4)

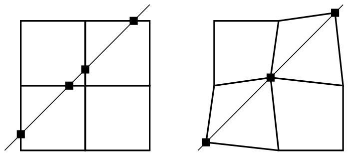

Fig. 3. Warping onto the short edge in 2D. This can also be seen as collapsing the short edge into a vertex.

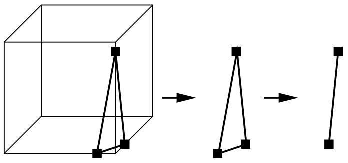

Fig. 4. Collapsing the short edge in 3D. The two long edges merge together into one edge. The top endpoint stays the same, and the bottom endpoint is the new collapsed vertex.

A quick mental experiment shows that the best place to which to move the corner is the midpoint of the short edge. Ideally, the warped triangulation stays as close to the known isosurface as possible. The midpoint of the short edge avoids ever getting too far away from the known isosurface, since it minimizes how far each endpoint has to move to achieve a collapsed edge. (Fig. 5)

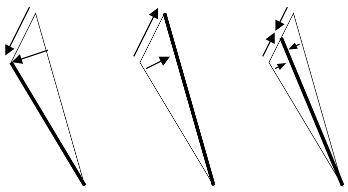

Fig. 5. Collapsing to the short edge's midpoint minimizes the distance from the known points of the isosurface.

One obvious question is what to do if there are two or more short edges at the same corner. If there are two short edges at the same corner, but not a third, then the two short edges must be in neighboring cubes. I choose to collapse the two edges into their shared vertex. This is better than collapsing first one and then the other short edge both because it keeps the resulting vertex on the known isosurface, and because it minimizes how far each of the neighboring non-short edges has to move. (Fig. 6)

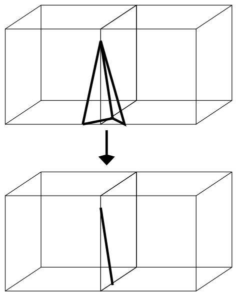

Fig. 6. Collapsing two adjacent short edges into their common vertex.

If there are three short edges at the same corner, then we have a small triangle that is likely to be well-shaped (by Delaunay standards). In this case, all three edges are collapsed into a single vertex. This looks like shrinking the small triangle into a vertex. Since the triangle is so small, any point within the triangle is a reasonable candidate for the final vertex, so I am currently using the incenter simply because it is guaranteed to be inside the triangle. (Fig. 7)

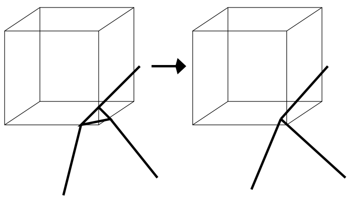

Fig. 7. Collapsing a small triangle into its incenter.

This concept is extended as edges are added. There can be a total of twelve short edges at one corner, which implies a "bubble" in the triangulation. If this bubble is completely disconnected from the rest of the triangulation, then all of its triangles will be nicely shaped, so there will not be any rendering artifacts. If the bubble is internal, then for the purposes of rendering, we can ignore the bubble completely. If the bubble has other edges extending from its vertices, one can collapse all of its edges collectively into the incenter of the octahedron defined by the twelve edges. This leaves a vertex with any outside edges now coming into it.

For any given number of short edges at one corner, I always choose to collapse the centermost piece at that corner first. Four edges forces the fifth, which is two adjacent triangles sharing a common edge. I collapse the common edge first, leaving two short edges meeting at the new collapes vertex. I then treat it as a two-edge adjacent pair. A sixth edge forces eight edges, and the centermost piece is a vertex at the "peak" of the resulting pyramid. Since the centermost piece is already a vertex, I collapse all of the edges directly into the peak vertex. A ninth edge forces a total of twelve edges which has already been mentioned above.

2.2. *Short altitudes*

The other kind of thin triangle is in some ways much more insidious. The wide-base, short-altitude triangle does not necessarily have an obvious short edge to collapse, and the collapsing procedure has more potentially dangerous consequences. In isolation, it seems rather innocuous. The triangle can simply be collapsed along its short altitude, resulting in an edge. (Fig. 8) This altitude is the shortest distance that this triangle can be flattened, thus again minimizing how far the collapsed triangulation strays from the known isosurface.

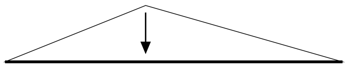

Fig. 8. The triangle is flattened onto its longest edge.

Within a triangulation, however, this collapse creates an extra vertex and requires the addition of a new edge to the triangulation. (Fig. 9) Whereas the short-edge collapse avoids ever shortening a remaining edge, the short-altitude collapse cuts a longer edge in two. This new edge has the potential to create a new thin triangle where there was not one before.

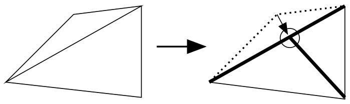

Fig. 9. The collapsed altitude results in a new vertex along the long edge and a new edge on the opposite side.

It is worth noting that there is only one way to get a short-altitude triangle from the original Marching Cubes. In particular, every triangle generated by Marching Cubes is contained within a cube, so the possible triangle configurations come from the Marching Cubes list. To get a short-altitude triangle, the two shorter edges must lie "across an edge of the cube", meaning that each edge lies in a face of the cube, and the two faces have an edge in common. (See Fig. 10) The vertex of the triangle that is an endpoint of the short altitude lies on the edge that is shared by the two faces of the cube. The two shorter edges of the triangle each extend to an adjacent edge of the cube respectively, and the distance along each of the cube edges where the triangle's edge ends is very small.

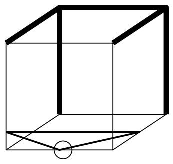

Fig. 10. A short-altitude triangle and the five edges where the opposite vertx may lie.

This means that the triangle on the other side of the short-altitude triangle's long edge can only take on a specific range of shapes. The opposite vertex can be in one of two relative locations. Consider the two faces across which the two shorter edges of the triangle lie. All seven of the edges bounding those two faces are not possible locations for the opposite vertex. This leaves five edges on which the opposite vertex may lie. Of these five, four give very similar triangle possibilities, and then there is the fifth. The fifth edge is the edge that shares both endpoints with other edges in this collection. All five of these possible edges yield the same worst-case triangle. This triangle *can* yield one short-altitude triangle, but that triangle can then be resolved by collapsing that triangle's shortest edge in the manner described in the previous section. This works because the only way to get another bad short-altitude triangle is to have a very short edge resulting from the first altitude-collapse. Further, the collapse of short edges can not generate a short-altitude triangle, so this ends the process.

There is one situation from the Marching Cubes 33 set of possible triangulations that allows for the short-altitude triangle to lie completely within one face of the cube [3]. In the cases where this occurs, there is always a fourth vertex on the remaining edge of that cube face, and this fourth vertex completes the triangle on the other side of the short-altitude triangle's longest edge. This opposite-side triangle is, at worst, better than the worst-case triangle from the regular Marching Cubes triangulation. This triangle can thus be handled the same way as described above.

2.3. *Short edges before short altitudes*

It is important to collapse the short edges first, before collapsing the altitudes. Once the altitudes have been collapsed, another (much smaller) pass of short-edge-collapsing removes any new short edges left behind by the altitude collapses. The reason for this order is to avoid creating an unnecessary new short edge when the short altitude is collapsed onto the longest edge. The longest edge is broken into two, and a short edge in the starting triangle will cause the long edge to be broken into one very short edge and one reasonably long edge. This new short edge is unwanted. If the short edges are not collapsed first, this kind of short-edge creation has the potential to turn into a series of such short edges appearing. By collapsing the short edges first, the probability of these series arising during the altitude collapse is greatly reduced, and the few newly created short edges can be handled quickly afterwards.

3. Results and Conclusions

Without loss of generality, I treat the grid size as 1, and define ε to be the minimum acceptable edge length as a fraction of the grid size. I define η to be the minimum acceptable altitude, also as a fraction of the grid size. On the first pass, edges that are shorter than ε are collapsed. Once the short edges have been addressed, I find all triangles with altitude smaller than η and collapse those. Another quick pass through gets rid of any newly created short edges. Let B be the circumradius-to-shortest-edge ratio for a triangle. Then if I choose $\varepsilon = 0.4$ and $\eta = 0.35$, then $B \leq 1.5$.

The fourteen cube configurations defined by the Marching Cubes algorithm can be reduced to six based on combinations, and each of these has a worst-case triangle. For about half of these cases, it is sufficient to have $\eta = 0.25$, but there are a few cases that have particularly unwieldy potential triangles, and these require that $\eta = 0.35$. Since the shortest edge length is set by ε, the resulting bound on B directly implies a nice upper bound on the circumradius of the triangles.

Quality bounds on triangulations in isosurface extraction are still relatively rare. While a great deal of quality analysis has been done on mesh-generation and triangulation algorithms for known surfaces and point-sets, there has been remarkably little analysis on the quality of the triangulations generated by isosurface-extraction algorithms, where the surface is not known *a priori*. Attali and Lachaud give an algorithm that locally constructs an isosurface that is Delaunay conforming [2] [1], thus giving some sense of triangulation quality, but even Delaunay triangulations often still contain the skinny triangles that can cause havoc in rendering contexts. I have presented an isosurface extraction algorithm that respects a guarantee of quality in terms of circumradius-to-shortest-edge ratio, an increasingly popular measure of mesh and triangulation quality.

References

[1] D. Attali, J.-O. Lachaud, Constructing Iso-Surfaces Satisfying the Delaunay Constraint. Application to the Skeleton Computation., *Proc. 10th Intl. Conf. on Image Analysis and Processing* (Venice, Italy, 1999) 382–387.

[2] D. Attali, J.-O. Lachaud, Delaunay Conforming Isosurface, Skeleton Extraction and Noise Removal, *Comp. Geometry: Theory and Applications* **19** (2001) 175–189.

[3] E.V. Chernyaev, Marching Cubes 33: Construction of Topologically Correct Isosurfaces, *CERN Report* **CN-95-17** (1995).

[4] P. Crossno, E. Angel, Isosurface Extraction Using Particle Systems, *IEEE Visualization* (1997) 495–498.

[5] K. Hormann, U. Labsik, M. Meister, G. Greiner, Hierarchical Extraction of Iso-Surfaces with Semi-Regular Meshes, *Proc. 7th ACM Symposium on Solid Modeling and Applications* (Saarbrücken, Germany, 2002) 53–58.

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

[7] S. A. Mitchell, S. A. Vavasis, Quality Mesh Generation in Higher Dimensions, *SIAM Journal on Computing* **29:4** (2000) 1334–1370.

[8] J. Sharman, The Marching Cubes Algorithm, *http://www.exaflop.org/docs/marchcubes/ind.html.* (2003).

[9] G. Varadhan, S. Krishnan, Y. J. Kim, D. Manocha, Feature-Sensitive Subdivision and Iso-Surface Reconstruction, to appear in *Proc. IEEE Visualization* (2003).