‘Skeleton Climbing’: fast isosurfaces with fewer triangles

Tim Poston
CIeMed
National University of Singapore
tim@iss.nus.sg

H. T. Nguyen
Microcomputer Research Labs
Intel Corp., Santa Clara
htnguye4@gomez.sc.intel.com

Pheng-Ann Heng
Tien-Tsin Wong
Dept. of Computer Sci. & Eng.
The Chinese University of Hong Kong
{pheng,ttwong}@cse.cuhk.edu.hk

Abstract

Skeleton Climbing is an algorithm that builds triangulated isosurfaces in 3D grid data, more economically than Marching Cubes, and without the time penalty of current mesh decimation algorithms. Building the surface from its intersections with grid edges (1-skeleton), then faces (2-skeleton), then cubes (3-skeleton), treats the data in a uniform way; this allows a 25% reduction in the number of triangles produced, while still creating a true separating surface at similar speed.

CR Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry & Object Modeling.

1. Introduction

One must often extract from 3D grid data, sampled by CAT scan, MRI, seismic sensors, *etc.*, an approximate isosurface Σ separating the grid points with data above or at a certain threshold level τ from those below. For brevity below, call these $\bullet$ and $\circ$ points.

Given modern graphics hardware, it is convenient to build Σ as a set of polygonal faces: most simply, avoiding nonplanar faces, a set of triangles, usually large. Half a million triangles for the surface of a scanned brain, for example, are a serious burden on current display technology.

The current standard means for this is the Marching Cubes algorithm (MC), from the covering patents[10, 11] and other related papers[12, 13], modified by various fixes[1, 3, 8, 14, 18, 19] for its failure (as patented) to construct a true separating surface. The new scheme described here gives a 25% reduction in the number of triangles. Other schemes for triangle reduction, such as [2, 4, 6, 7, 16, 17], test many possible change-steps for each actually used, and take tens of minutes. They do not guarantee to keep $\bullet$ and $\circ$ points separated, and they are complex to parallelize, as are methods like [9] which track a component of the surface through the data volume. Skeleton Climbing’s mergers are valid *a priori*, producing a true separating surface at a speed little different from MC, with fewer triangles and none of the parallelization problems of reduction methods designed for general meshes. In interactive data exploration, one cannot wait an hour for a more easily drawn surface.

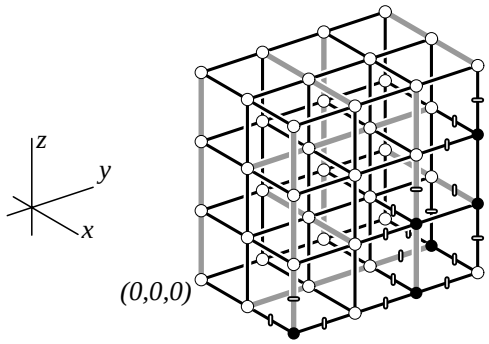
Our scheme constructs Σ in an efficient, direct way for each configuration: vertices in the edges of the cubical grid, triangle sides in the square faces, and finally triangles in the cubes. Such progression from the 1-dimensional ‘skeleton’ (edges, attached to corners), to the 2D skeleton (faces, attached to edges), to the 3D skeleton (cubes, attached to faces), to . . . , *etc.*, is known in topology as ‘skeleton climbing’, a suitable name for the proposed algorithm, summarized in Fig. 1. The triangulations chosen have a regular relation to the data grid, that allows a fast reduction to a simpler separating surface.

Section 2 discusses the construction of the 1-skeleton, fixing sides within faces. Section 3 constructs the 1-skeleton, placing triangles across them. Choosing certain specific rules (a) reduces total triangle side length, (b) prevents surface self-intersections within cubes, and (c) allows fast merging of triangles (Section 4). We discuss implementation in Section 5, results in Section 6, and state conclusions in Section 7, with some description of future and related work.

2. Building the 1-skeleton

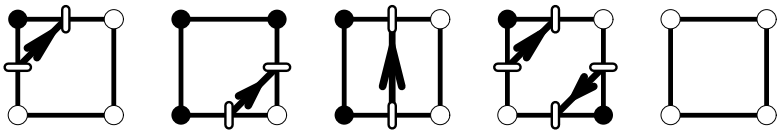
In steps 1–3, we choose surface vertices just as MC does: one on each cube edge joining a $\circ$ to a $\bullet$, positioned by linear interpolation. Rather than go directly to cubes, we then construct surface edges between vertices that share a cube face, and then surface triangles between edges that share a cube. The original MC direct step to cubes reduced the number of cases both by symmetry and by interchanging $\circ$ and $\bullet$. This can mismatch edges in a $\begin{smallmatrix}\circ & \bullet\\ \bullet & \circ\end{smallmatrix}$ face, giving holes in the surface. Building the 1-skeleton first, and reading it from both sides, avoids this.

The four vertices on the edges of a $\begin{smallmatrix}\circ & \bullet\\ \bullet & \circ\end{smallmatrix}$ face can join in two ways. The choice can use interpolation schemes such as [14], but all current ones assume that the sampled function ‘looks polynomial’ on the scale of a voxel; when one is fitting a surface of discontinuity, such as that between the density of bone and the density of soft tissue, the results are not especially meaningful. Where the surface as a whole is smooth on the voxel scale, the choice is typically without topological effect (Fig. 2). The simplest rule, “do not cross



Step 1

Determine those edges (marked with cross-bars) in the 1-skeleton of the cubical grid that join • and ∘ points, and must thus contain vertices (elements of the 0-skeleton of Σ). Call these *occupied* edges. *Grey edges* are the vertical edges through points (l, m, n) with l and m even, the x-direction edges with m and n odd, and y-direction edges with l odd and n even.

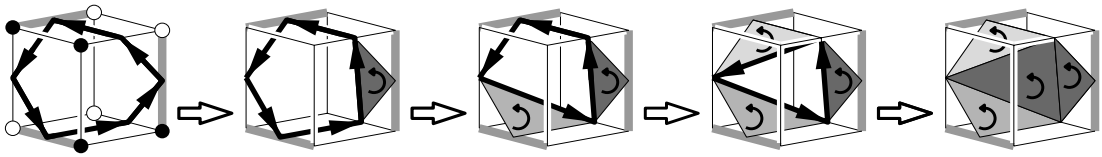


Step 2

Decide for each face F in the 2-skeleton of the cubical grid which pairs of occupied edges will be joined by triangle sides (elements of the 1-skeleton of Σ). Do not cross ∘—∘ diagonals. Orient sides to keep neighbouring •s to the left.

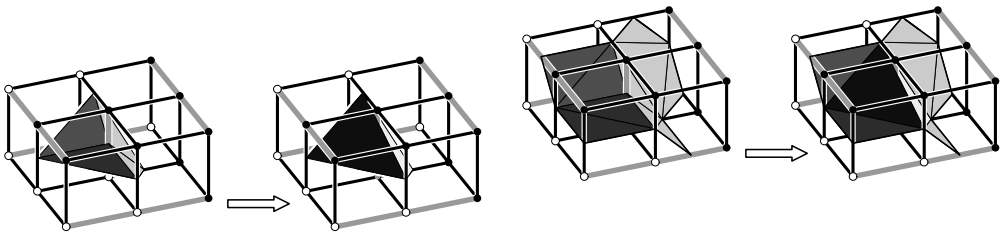
Step 2a

Transfer this side structure (with opposite orientations) to the two cubes sharing face F.



Step 3

Find triangles (elements of the 2-skeleton of Σ) in each cube of the 3-skeleton of the cubical grid, to span the loops around it formed by the sides across its faces from Step 2. Cube edge by cube edge (starting with the grey edges), if the edge E is occupied, shrink the set of sides by merging the two that end on E; emit a triangle with the orientation given by these two sides. If their ends were already joined, mark these ends' cube edges 'unoccupied'.



Step 4

For each occupied grey edge, replace the four triangles that meet on it by two triangles with the same boundary. We show this both with and without the other triangles found by Step 3 in the 2×2 block around E. Section 4 shows why such a four-triangle reducible group exists for every occupied grey edge.

Figure 1. The four steps of the Skeleton Climbing algorithm.

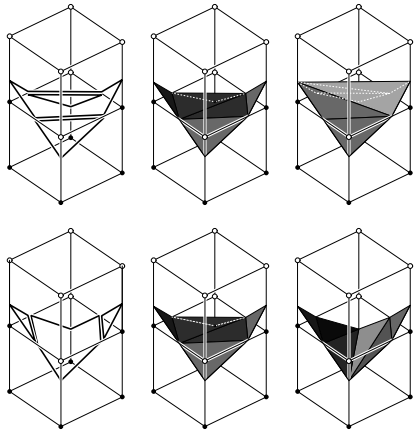


Figure 2. Where the regions separated by a true isosurface are more than a voxel thick, so that a ∘• / •∘ face is sandwiched between all-∘ and all-• faces, the choice of edges (left) makes no topological difference to the surfaces that fill in either set of three resulting polygons. Where a true surface of discontinuity is more complex on this scale, the sampling gives insufficient detail for any interpolations scheme to reconstruct it reliably.

∘–∘ diagonals", is as sensible as any more complex choice, and convenient for speed.

We do not reduce by symmetry in the Skeleton Climbing algorithm, but (to avoid displaying 256 configurations) Fig. 3 shows only one from each set of non-empty cases equivalent by reflection and rotation. Moreover, for complementary pairs of cases without a ∘• / •∘ face, only one is shown. The numbering and view are kept similar to those in [10, 11, 12], for easier comparison by readers familiar with MC. (Case 14 in the MC literature, identical to 11 by reflection in a diagonal plane, is omitted here.)

3. Surface construction

If an n-gon is filled in by triangles without introducing new vertices, and the triangles form a disk, there are $n-2$ of them. (Topological evidence for tubes and handles at the sampling-cube scale cannot be secure, so we assume these complications to be absent. This also minimizes the triangle count.)

Any such disk can be constructed by 'nibbling corners' as in Fig. 1, Step 3, in some order. Any order gives the same (MC) number of triangles within the cube. We choose an order allowing a systematic merger across cubes, that reduces the triangle count without sacrificing topological separation of ∘ and • points.

The nibbling process is algorithmic and straightforwardly coded. One can either use to build a look-up table,

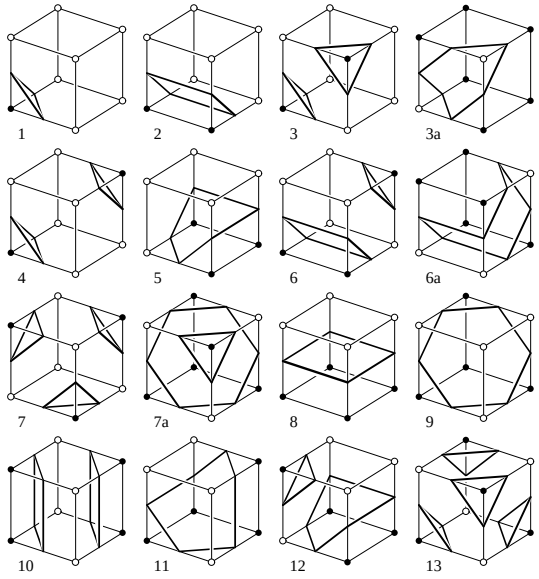


Figure 3. Side patterns in cube faces for the "do not cross ∘—∘ diagonals" rule. In 3 *vs* 3a, 6 *vs* 6a, and 7 *vs* 7a, complementation gives a pattern in a different reflection/rotation class. In cases 10, 12 and 13 the patterns are different, but equivalent by reflection and/or rotation.

or in SIMD parallelism (where lookup is expensive) build it into the active code.

Step 3 (Fig. 4) iteratively performs **EmitTriangle:**

(a) Choose a cube edge E from the cube's 'occupied' set.

(b) The oriented edge starting at a point on E ends at an edge E' we call 'forward' of E; the one arriving at E begins at E'', 'back' of E. If the edges E' and E'' are already 'back' and 'forward' of each other on the current cube, relabel both as unoccupied. Else, join them with a new side (that is, mark them as 'forward' and 'back' of each other).

(c) Relabel E as unoccupied, and add the oriented triangle joining the vertices on E, E'' and E' (in that order) to the surface Σ under construction.

until all cube edges are 'unoccupied'. The order of choices for E is discussed in Sections 3.1, 4 below.

3.1. Choice of triangulation

It is clear in Step 3 of Fig. 1 that trimming off vertices in a different order could give a different set of triangles; always four, as required to fill six sides. (Each step shrinks the set of vertices by one; the third leaves a single triangle.) There is a large set of possible edge choice sequences for

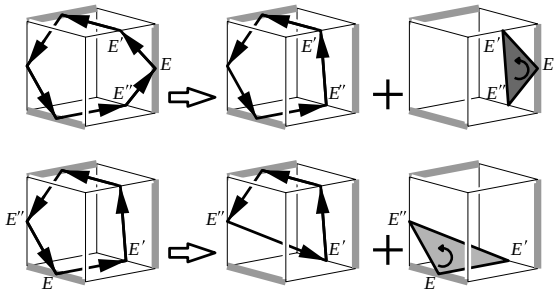


Figure 4. The first two triangle emissions in the Step 3 example of Fig. 1. Arrows indicate forward indices; at left, for instance, forward[E] = E', back[E'] = E.

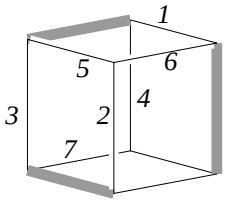


Figure 5. Describing possible edge sequences after the greys, up to symmetry. (The last two of the cube's twelve edges become unoccupied before they are reached.) If an edge joining two greys is chosen first, by symmetry we can assume it is the one seen at top, here labelled 1. If the second also joins two greys, assume it is that seen at left (in this example, 3), otherwise that it is one of the three front ones. If the first joins a grey to a black, assume it is the front vertical (in this example, 2).

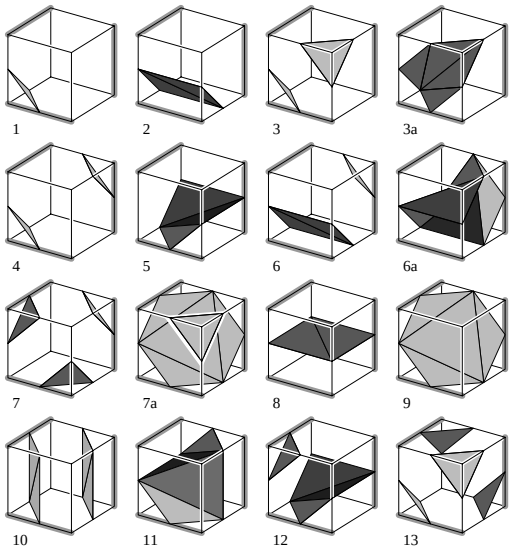


Figure 6. Typical Skeleton Climbing Step 3 triangle sets for the edge sequence in Fig. 5.

applying EmitTriangle, and hence a large class $\mathcal{R}$ of algorithmically distinct ways to assign a minimal triangle set, though not always with distinct results. For the example in Fig. 4, there are 360 distinct sequences E_1, E_2, E_3, E_4 of edge choices which actually emit triangles, embedded in longer sequences of Es that are unoccupied when reached. They give just 14 different surfaces.

Any minimal triangle set on any cube can be produced by many $R \in \mathcal{R}$, but there do exist strategies which no R matches for all cubes, such as the 'fan' strategy, radiating all interior edges from the first vertex seen. This produces ugly forms, with long thin triangles, resulting in a greater edge length and a more 'wrinkled' surface than necessary. Any fixed cube-edge sequence gives far less side length than the data-dependent one this corresponds to.

One way to select R is to emphasize geometrical qualities, and minimize the 'bentness' of Σ, defined by summing dihedral angles, $|\text{angle sum} - 360°|$ at vertices, and so on; or minimize lengths. All of these vary with the interpolated vertex positions, so strategies should be tested over real datasets. In the simple 'visit all cube edges, eliminate if occupied' coding approach there are 9,979,200 distinct sequences up to cube symmetry. We have therefore studied this optimization only for the much fewer Step-4-compatible cases that visit grey edges first.

There is enough symmetry (Fig. 5) to greatly reduce the sequences one need look at; the order shown produced about 2% less total edge length (using length$(x, y, z) = |x|+|y|+|z|$) than the maximum length found, on sample medical data. (Note that the grey edges in neighbouring cubes are reflections of those in this one, so that their numbering must match this reflection.)

A useful side effect of this optimization is that intersection between the triangles becomes impossible (by a tedious examination of cases). Using the edge sequence in Fig. 5, Skeleton Climbing produces only properly embedded triangle sets for each cube.

Fig. 6 shows sample Skeleton Climbing triangle assignments for the cases in Fig. 3 with this edge-elimination sequence. They are not 'the' assignments for these cases, since in cubes with different (i, j, k) parity and thus different grey edges, translations of the same side set may get different results.

In most cases the triangles generated lie strictly inside the cube, except at the polygon sides they are present to fill. In a few cases (Fig. 7) a set of sides may admit—though not require—a less perfectly internal set of triangles. If this happens also for the cube meeting C in F, the result is a small area covered twice, by triangles (or parts of triangles)

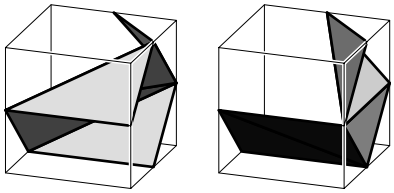


Figure 7. Two spanning surfaces for the same fixed side configuration, with a fold side and a triangle within a cube face (only 3a and 6a of Fig. 3 admit this).

with opposite orientations: or a fold edge from each cube may meet another, or a triangle. Where the lookup table generated by a particular rule R generates such an example, it is easily avoided by switching that case to a rotationally equivalent rule $\widehat{R}$, equally compatible with Step 4, which gives strictly-internal triangles for that case. In lookup-table-unfriendly SIMD, it is harder to avoid.

Any two schemes involving the same sides within cube faces, spanning them by triangulated disks interior to the cube without interior vertices, must involve exactly the same number of triangles. Fig. 7 points up the only possible reduction beyond this, for a fixed set of triangle sides. Where two cubes sharing a face F both allow a minimal span with triangles in F, one could put two such in each, spanning the same parallelogram, and cancel all four. This would still give a valid separating manifold surface Σ, though with a different topology. The reduction would be tiny, and (like our Step 4) would adjust what goes into one cube by checking its neighbours.

This apart, the minimum number of triangles can be found by summing $(|\text{sides}| - 2)$ over all loops around all cubes, so within-cube methods differ little in triangle counts. The original MC used complementation to retain 3 and 6 of Fig. 3, rather than 3a and 6a, each of which gives two triangles more; the result is rarely fewer by more than 1% overall, and leaves holes in the surface. Every amended MC uses more. For instance, [19] produces exactly the side set generated in Step 2, by a somewhat different route, and hence exactly the same triangle count as the steps 1 to 3 given here. (The actual triangles are different—[19] simply emits the sides as polygons, leaving the graphics hardware to triangulate them as fans—but the numbers are equal.) Step 4, which goes outside the cubes, yields an identical reduction from the triangle count of this whole class of methods.

4. Triangle reduction

In any cube C, the first occupied edge E handled in Step 3 produces a triangle T with two edges lying in cube faces (Fig. 8), and this is the only triangle in C with a vertex (v,

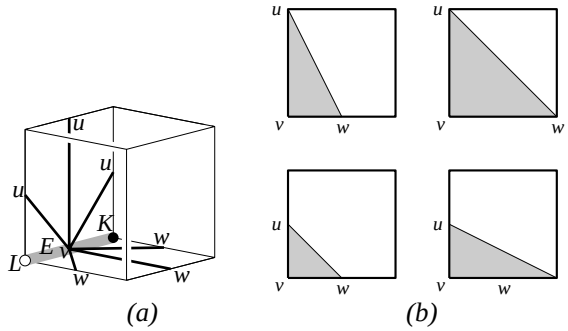


Figure 8. (a) Possible fixed sides in C ending at vertex v on edge E. (b) Triangles formed by these possible sides, viewed parallel to edge E.

say) on that edge. If the other vertices u, w of T are not themselves joined by a fixed edge (thus eliminated in finding T), each of the edges they occupy will later acquire at least one additional triangle incident on it, meeting T along the interior edge uw. Their surroundings in C thus become more complicated than those of v.

Suppose now that E is the first edge handled in each of the four cubes C_1, C_2, C_3, C_4 around it. It is then at the meeting point of a set Q of just four triangles in Σ, lying within $\mathbf{C} = C_1 \cup C_2 \cup C_3 \cup C_4$ with a quadrilateral boundary $\delta Q \subset \delta\mathbf{C}$ whose linking number with E is 1. Any other surface within $\mathbf{C}$ spanning δQ will also meet E (generically, an odd number of times) and separate the $\bullet$ and $\circ$ points at E's ends.

In particular, either triangle pair spanning δQ is within $\mathbf{C}$ and can replace Q without harming Σ's separation properties, as shown in Step 4 of Fig. 1. We do need (Fig. 9) to let Σ pass through $\circ$ points, as for $\bullet$ points the definition "above *or at* τ" already allows.

For any single edge, it is easily arranged that the above replacement of four triangles by two is possible. The gain comes when we can do it for many. If we take the set $\mathcal{E}$ of those vertical lines in the grid with both i- j-coordinate odd, every cube C in the grid has exactly one edge E on a line in $\mathcal{E}$, so we can order its edges to handle that edge first. After E is handled, we can no longer assume for an arbitrary edge E' that E' will meet at most one triangle in C (though if E is unoccupied, E' may still do so by being the first occupied edge handled); nor that the set of triangles meeting E' will be unaffected by the replacement we do around E.

We can assume it, however, for any edge F that does not share a face with E (Fig. 10); and after handling F we can assume it for the unique edge G that shares no face with E or F. Moreover, the replacements around E, F and G will not share the same triangles. As Fig. 1 shows, we can simultaneously find such a set E, F, G of 'grey' edges for every cube: three out of every twelve edges in the grid. Step 4

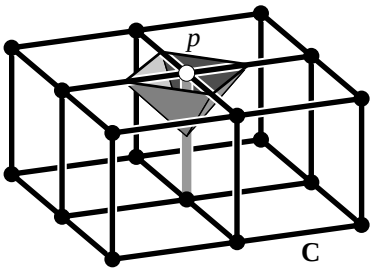


Figure 9. Four triangles around a single ∘ point p in C are replaced in Step 4 by two that meet in p.

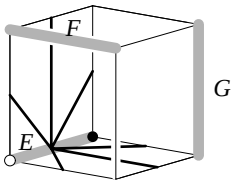


Figure 10. Three grey edges sharing no face.

replaces four triangles by two around every such edge, if occupied, and eliminates the vertex on it. Each instance of Step 4 can be performed in two ways, depending on which opposite corners of δQ are joined by a new triangle side. The choice has little impact on the appearance of the surface, and none on its topology, but does affect such issues as memory management; if for every horizontal grey edge we choose the merged edge to be horizontal, the triangles in each layer remain within it, and can be computed without reference to other layers. Choosing to cross x-direction grey edges with y-direction new sides, and so on cyclically, partitions the data into 2×2 blocks whose borders no triangles cross. The reduction therefore loses no parallelizability, at less than the extreme level of one processor per voxel.

If the occupied edges are distributed without some special parity bias in the data, Step 4 will thus eliminate about 25% of the vertices in Σ. A few-handled surface without boundary has about twice as many triangles as vertices, so this reduction translates as 25% fewer faces. Deviation from this is more often due to multiple components than to multiple handles; in the extreme case of N isolated $\bullet$ points, Step 3 would surround each $\bullet$ by an octahedron. Step 4 would replace about $3/4$ of these by flattened tetrahedra; $8N$ triangles become $8(n/4)+4(3N/4) = 5N$, a 37.5% reduction. But such isolated points are commonly pre-filtered out as 'noise'; fortunate here, as the flattened tetrahedra would look slightly odd as a way of marking them. Trials on medical data have yielded savings from slightly above 25% to about 31%; no case with less than 25% reduction has so far occurred, though a nasty example could of course be synthesized.

5. Implementation

One original motivation for the work was to find a more SIMD parallelism-friendly algorithm than MC. In its simplest form, using a 256-entry lookup table, SIMD MC would require each processor to march lockstep through 256 possible cases, since to use variables as array names is costly or impossible on such machines. Most steps, on most processors, are thus wasted. In [20], MC—in its original, quadrilateral-hole version—was reduced to a 63-step form, a useful speedup over [5]. Steps 1–3 of Skeleton Climbing give topological correctness, in a 24-step SIMD form. Some of these steps are more complex than those for MC (though further optimization may be possible), so that the resulting speed advantage over MC was about 12% rather than threefold, when tested on a Wavetracer SIMD machine. (Step 4 has not yet been parallelized.)

6. Results

The sequential version currently implemented, being en route to the parallel code, avoids lookup tables where possible. (In fact, the 8-bit indexing of 256 cube configurations—central to the MC patent—nowhere occurs in the parallel code.) Clearly, a lookup implementation of Steps 1–3 would run in about the same time as for MC; the current version, computing the triangles for each cube from first principles, runs about 6% slower than an 'original' MC implementation; that of [19] is twice as fast, but we have not yet done a similar optimization of Skeleton Climbing.

Step 4 decreases the combinatorial time by an average of 4%; the extra load is balanced by a 25% reduction in the triangle-emission work. The rendering speedup is thus essentially free. Quantitative results are summarized in Table 1.

The graphical results are very satisfactory. In each of the figures below we show results for the Wyvill[19] version of Marching Cubes, the Steps 1–3 version of Skeleton Climbing, and the fewer-triangle version with Step 4 included. Each surface is shown Gouraud shaded, with the triangle edges added to display the algorithms' effects in detail; these would normally be omitted in applications. The normals for shading are computed by the same gradient estimation, at the same vertices (except for Step 4's deletions) using each method.

Topologically, Step 4 is without penalty. Geometrically, when Σ is an isosurface for a smooth function (so that interpolation places vertices fairly accurately), some difference is visible. Fig. 11 shows results for the knotted torus $\{\sin\left(\pi\sqrt[6]{(k_1^2+k_2^2)}\right) = 0.9\}$, where

$$(k_1,k_2) = \left(p^2-q^2+r^3-3rs^2, 2pq-3r^2s+s^3\right),$$
$$(p,q,r,s) = \frac{\left(2x,2y,2z,\sqrt{1+x^2+y^2+z^2}-2\right)}{\sqrt{1+x^2+y^2+z^2}}, \quad (1)$$

sampled on a $64\times64\times64$ grid.

At the finest part of the tube Step 4 makes some visibly artefactual notches, which if the tube were a real artery might falsely suggest stenoses (narrowings); but medical data are less smooth. Linear interpolation locates vertices very accurately where the data are sampled from a differentiable function like F, since 'differentiable' precisely means that a local linear approximation is good. When the $\bullet$ and $\circ$ values cluster around two characteristic densities t_0 and t_1, the surface of interest is a surface of discontinuity, where linear interpolation fails badly. (A pair of height values along a stair cannot predict the position of the step, they only limit it.) In most important applications outside the synthetic data created in CAD for implicit surfaces, the user wants not a true isosurface (the set of points where a function has the value τ) but a discontinuity surface, the set of points where the gap between neighbouring values includes the value τ. Estimations of this are more difficult.

Linear interpolation puts the vertices at a characteristic position along $\bullet$—$\circ$ edges, depending on the ratio $(t_1-\tau)/(\tau-t_0)$. This produces the characteristic 'stepped forehead syndrome' in Fig. 12. The effect *decreases* in the Step 4 version of Skeleton Climbing, due to the triangles that stretch between cubes on the step edges, with no discernable loss of surface quality in the more complicated regions.

Fig. 13 shows real arteries, from a CT scan of a cadaver head injected with contrasting plastic. Each surface—MC, and Skeleton Climbing with and without Step 4—shows artefactual notches in the smaller arteries, often at different points, with at most a statistical difference between them. (The physician concerned with a *particular* narrow artery and its possible stenosis prefers a volume rendered image, such as a maximum intensity projection, to any extracted surface, with its τ-dependent artefacts.)

A higher-resolution discontinuity surface is shown in Fig. 14. Step 4 creates no reduction in quality in the smooth regions, and in the complex region visible through the nearer eye-socket (where some fine bones are of sub-voxel thickness) the resulting image is an equally good representation of structure.

In Fig. 15, from a volume data set of Mount San Antonio, Step 4 gives a clear reduction in step phenomena aligned with the data grid (and thus clearly artefactual). This allows the real striations of shape, created by differential weathering of rock layers, to stand out more clearly.

7. Conclusions and further work

Basing isosurface construction on ascent by dimension allows a reduction of 25% or more in triangle count, without speed penalty, and with equal or better surface quality. Skeleton Climbing is now integrated into the package DISHA (Determination of Short-Axis Slices of the Heart Automatically), developed at CIeMed.

Moreover, since the fundamental controls cube meetings first, it can be elaborated to handle blocks of variable size (adaptively to the local data complexity), without the cracks and crack patching in adaptive algorithms based on Marching Cubes. The adaptive approach[15] is no longer guaranteed to separate the identical grid points separated by Marching Cubes and Skeleton Climbing, and its greater complexity does have a cost in time, but it can often achieve a five- to twentyfold reduction in triangle count at remarkably little sacrifice in speed or surface quality.

A solid base in the topology of cell complexes, and the skeleton climbing approach, also offers a systematic and effective way to construct the multiple surfaces that separate muscle/blood/bone/nerve/$\cdots$ or granite/gravel/water/oil/$\cdots$ from each other as a consistent structure, rather than extract these regional surfaces separately. Work on this extension is in progress.

We would like to thank the National Science and Technology Board of Singapore for financial support through the Centre for Information-enhanced Medicine, and Dr Tushar Goradia of Johns Hopkins University for the cadaveric artery data used in Fig. 13.

Web availability

The C implementation of Skeleton Climbing and other supplementary materials are now available on web: `http://www.cse.cuhk.edu.hk/~vis/`

References

[1] Baker, H. H. Building surfaces of evolution: The weaving wall. *Int. J. Comp. Vision* **3** (1989), 51–71.

[2] Dehaemer, M. J., and Zyda, M. J., Simplification of objects rendered by polygonal approximations. *Computer & Graphics*, 15(2):175–184, 1991.

[3] Van Gelder, A. & Wilhelms, J. Topological Considerations in Isosurface Generation. To appear in *ACM Transactions on Graphics*.

[4] A. Gueziec and R. Hummel, The Wrapper Algorithm: Surface Extraction and Simplification, *Proceeding of the IEEE Workshop on Biomedical Image Analysis*, pp. 204–213, 1994.

[5] Hansen, C., and Hinker, P., Massively Parallel Surface Extraction, in *Proceedings Visualization '92*, 77-83.

[6] Hansen, C., and Hinker, P., Geometric Optimization, in *Proceedings Visualization '93*, 189–195.

[7] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald and W. Stuetzle, Mesh Optimization. *Computer Graphics (SIGGRAPH '93 Proceedings)*, pp. 19–26, 1993.

Data Set	Marching Cubes	Skeleton Climbing (1–3)	Skeleton Climbing (1–4)
Knot	$13,968\triangle$	$13,968\triangle$	$10,464\triangle$
$64 \times 64 \times 64$	1.81*sec*	1.55*sec*	1.61*sec*
CT skull	$100,830\triangle$	$100,066\triangle$	$75,318\triangle$
$128 \times 128 \times 57$	8.11*sec*	8.77*sec*	8.16*sec*
Arteries	$263,438\triangle$	$265,536\triangle$	$195,588\triangle$
$256 \times 256 \times 148$	55.82*sec*	59.80*sec*	63.15*sec*
CT skull	$592,368\triangle$	$595,802\triangle$	$446,990\triangle$
$256 \times 256 \times 113$	62.02*sec*	63.38*sec*	60.60*sec*
Mt San Antonio	$268,252\triangle$	$268,368\triangle$	$201,686\triangle$
$258 \times 258 \times 255$	113.77*sec*	92.91*sec*	96.33*sec*

Table 1. Comparison of Marching Cubes and Skeleton Climbing, without and with the Step 4 merger, by number of triangles and CPU running times.

[8] Kalvin, A. D., *Segmentation and surface-based modeling of objects in three-dimensional biomedical images*. PhD thesis, New York University, 1991.

[9] Livnat, Y., Shen, H-W, and Johnson, C. R., A near optimal isosurface extraction algorithm using the span space. *IEEE Transactions on Visualization and Computer Graphics*, 2(1):73–84, March 1996.

[10] Lorensen, W. E. & Cline, H. E., SYSTEM & METHOD FOR THE DISPLAY OF SURFACE STRUCTURES CONTAINED WITHIN THE INTERIOR REGION OF A SOLID BODY. US Patent 4,710,876 , 1987.

[11] Lorensen, W. E. & Cline, H. E., SYSTEM & METHOD EMPLOYING NON-LINEAR INTERPOLATION FOR THE DISPLAY OF SURFACE STRUCTURES CONTAINED WITHIN THE INTERIOR REGION OF A SOLID BODY. US Patent 4,729,098 , 1988.

[12] Lorensen, W. E. & Cline, H. E., Marching Cubes: a high resolution 3D surface construction algorithm. *Computer Graphics* **21** (1987), 163-169.

[13] Wyvill, G., McPheeters C. & Wyvill, B., Data Structure for Soft Objects, *The Visual Computer*, 2(4):227-234, February 1986.

[14] Nielson, G. M. & B. Hermann, B., The asymptotic decider: resolving the ambiguity in marching cubes. *Proceedings of Visualization '91* (1991), 83–90.

[15] Poston, T., Wong, T.-T. and Heng, P.-A., Adaptive Skeleton Climbing, submitted.

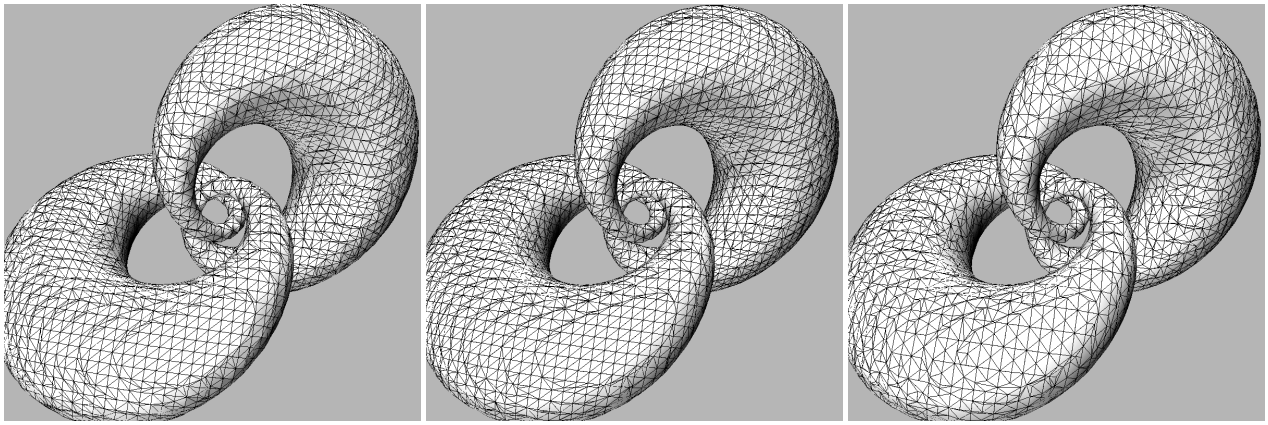
[16] Schroeder, W. J., Zarge, J. A., and Lorensen, W. E., Decimation of Triangle Meshes, *Computer Graphics* **26** (1992), 65–70.

[17] Turk, G., Re-Tiling Polygonal Surfaces, *Computer Graphics* **26** (1992), 55–64.

[18] Ning, P. & Bloomenthal J., An Evaluation of Implicit Surface Tilers, *IEEE Computer Graphics & Applications*, 13(6):33-41, November 1993.

[19] Wyvill, B. & Jevans, D., Table Driven Polygonisation. SIGGRAPH 1990 course notes **23**, 7-1–7-6.

[20] Zheng, M. & Nguyen H. T., An Efficient Parallel Implementation of the Marching-cubes Algorithm. *Massively Parallel Processing Applications & Development Proceedings* (1994), 903–910.

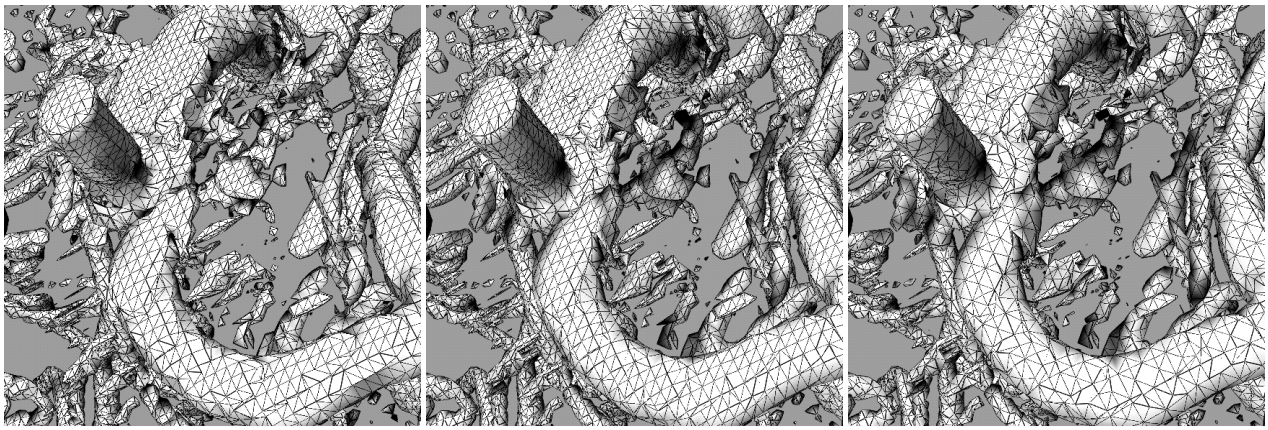


(a) (b) (c)

Figure 11. The surface defined by Equ. 1, sampled at $64 \times 64 \times 64$ resolution. Extracted by Marching Cubes (a) and by Skeleton Climbing without Step 4 (b), and with Step 4 (c), it requires (a) 13968, (b) 13968 and (c) 10464 triangles respectively. Step 4 preserves the topology, but for an isosurface of an analytic function it loses a little smoothness where the tube is finest.

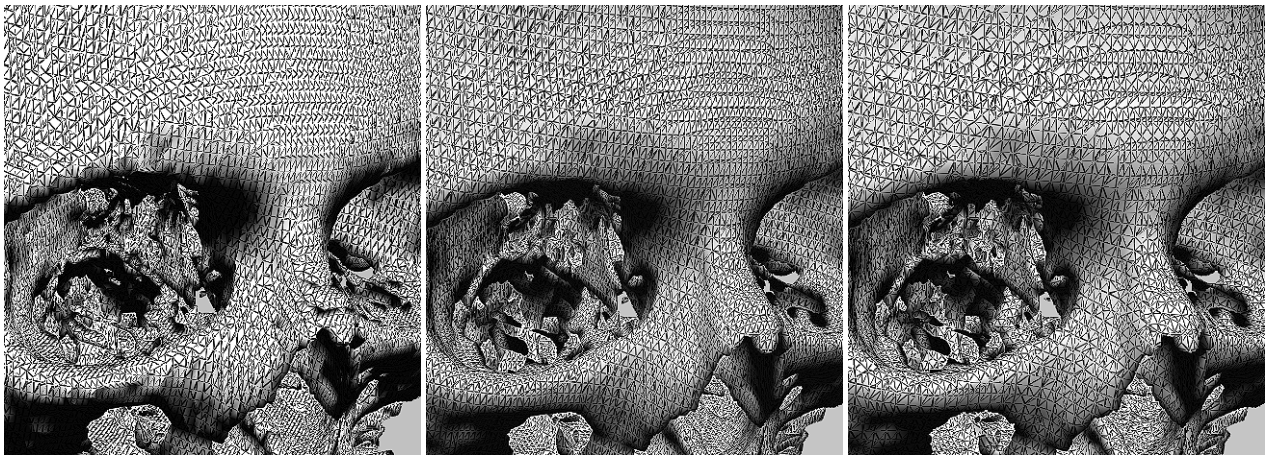
(a) (b) (c)

Figure 12. Surfaces extracted from a $128 \times 128 \times 57$ CT scan, with a threshold distinguishing bone from non-bone; (a) 100830, (b) 100066 and (c)75318 triangles. Skeleton Climbing with Step 4 gives a smoother forehead region, and equally good detail.



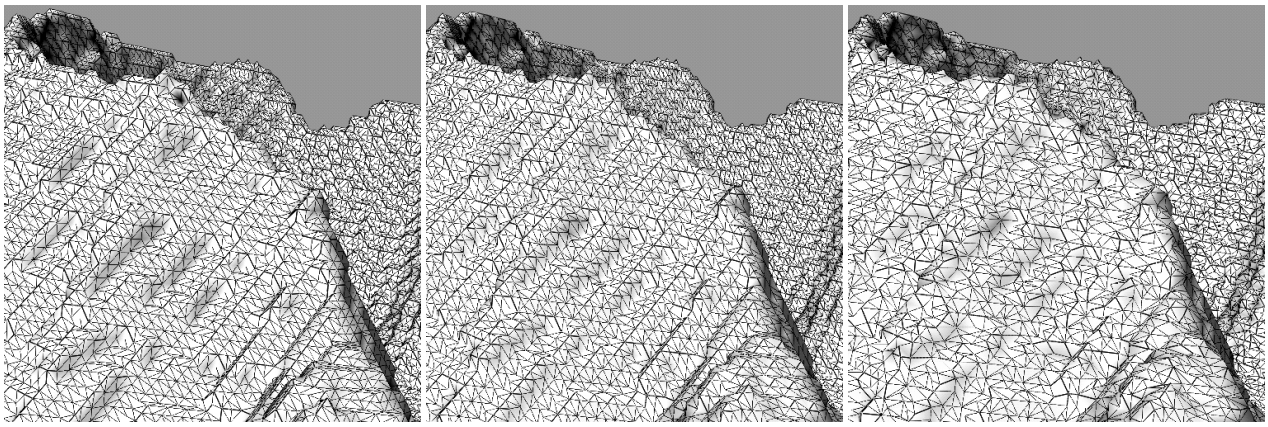
(a) (b) (c)

Figure 13. Artery surfaces extracted from a $256 \times 256 \times 148$ CT scan with injected contrast plastic; (a) 263438, (b) 265536 and (c) 195588 triangles. The quality is similar for all three.



(a) (b) (c)

Figure 14. A higher resolution ($256 \times 256 \times 113$) CT head; (a) 592368, (b) 595802 and (c) 446990 triangles. The quality is similar for all three.



(a) (b) (c)

Figure 15. The surface of a $258 \times 258 \times 255$ data set for Mount San Antonio; (a) 268252, (b) 268268 and (c) 201686 triangles. Artefactual ridges are reduced by Step 4, allowing the geology to show more clearly in the fewer-triangle surface.