The Wrapper Algorithm for Surface Extraction in Volumetric Data

André Guéziec
Courant Institute of Mathematical Sciences and New York University Medical Center, New York University
e-mail: gueziec@cs.nyu.edu

Robert Hummel
Courant Institute of Mathematical Sciences
New York University
e-mail: hummel@cs.nyu.edu

Abstract

Beginning with digitized volumetric data, we wish to rapidly and efficiently extract and represent surfaces defined as isosurfaces in the interpolated data. The Marching Cubes algorithm is a standard approach to this problem. We instead perform a decomposition of each 8-cell associated with a voxel into five tetrahedra, as in the Payne- Toga algorithm [Payne and Toga, 1990]. Following the ideas of Kalvin [Kalvin *et al.*, 1991], Thirion *et al.* [Thirion and Gourdon, 1993], and using essentially the same algorithm as Doi and Koide [Doi and Koide, 1991], we guarantee the resulting surface representation to be closed and oriented, and we evaluate surface curvatures and principal directions at each vertex, whenever these quantities are defined. We define a valid triangulation by representing the body as a collection of tetrahedra, some of which are only partly filled, and extracting the surface as a collection of closed triangles, where each triangle is an oriented closed curve contained within a single tetrahedron. The entire surface is "wrapped" by the collection of triangles. The representation is similar to the homology theory that uses simplices embedded in a manifold to define a surface.

From the triangles that comprise the wrapping of the surface, we can easily perform an analysis of the entire structure. We use a data structure that permits efficient access to neighboring triangles and vertices, by labeling each triangle with a unique identifier. We thus create a graph structure, which is very efficiently encoded, whose nodes are the surface elements (the triangles), and the edges are the common edges joining adjacent triangles. We can thus identify each surface using a connected component labelling algorithm applied to the graph.

This provides a highly parallelizable approach to boundary surface representation, providing an efficient and compact surface representation. The Wrapper algorithm has been used to extract surfaces of the cranium from CT-scans and cortical surfaces from MR-scans at full resolution.

Key-words : *Tetrahedral Decomposition, Homology Theory, Graph Structure, Surface Curvature.*

Introduction

The Wrapper Algorithm is an efficient, simple, parallelizable method for creating a boundary representation (a "B-rep") of isosurfaces in volumetric data defined on a 3-D grid. The B-rep will consist entirely of planar triangular faces, linked in a graph structure to form a coherent, valid surface. Indeed, the B-rep will necessarily bound a polyhedral solid or multiple polyhedra. The principal advantages of the Wrapper Algorithm over Marching Cubes [Lorensen and Cline, 1987] is that (1) the algorithm is parallelizable, and (2) the algorithm provides a provably valid global representation of the isosurface, in all cases. The surface representation provided by the Wrapper Algorithm is relatively uniform, being composed of triangular faces, and vertices in the representation have degree no higher than ten (i.e., no more than ten triangles ever meet at a single vertex). The Wrapper Algorithm uses a tetrahedral decomposition of the voxel space, and thus operates at a fine scale of detail, which provides the provable correctness of the representation, but also provides a voluminous data structure. In order to reduce the quantity of data in the representation, we have also developed ***simplification algorithms*** that operate on the B-rep provided by the Wrapper Algorithm. We display results obtained from the simplification process.

The details of the Wrapper Algorithm coincide, to a large extent, with the algorithm of Doi and Koide, of IBM Japan, Ltd, described in [Doi and Koide, 1991]. Doi and Koide use the same tetrahedral decomposition, produce oriented cycles to represent triangular faces of the isosurface, and even use the same determinant test to establish the orientation of the cycles. Since their work predates ours, a large part of our contribution is to explain the Doi/Koide algorithm, and to draw attention to its advantages.

Apart from the Doi/Koide work, there is a large body of literature on methods of constructing B-reps of isosurfaces in an efficient and accurate fashion. The Wrapper Algorithm provide a natural extension and (we believe) improvement on other methods. While problems with the Marching Cubes algorithm have largely been addressed and fixed [Nielson and Hamann, 1991], the Wrapper Algorithm, with its appeal to homology theory for provable correctness, provides a much simpler and intuitive approach to surface representation. The Wrapper Algorithm also provides an orientation of the surface without additional cost. Although the Wrapper Algorithm operates at a finer scale and provides a more detailed representation, which can be disadvantageous, it provides a more uniform platform for representation simplification, and can be viewed as a decomposition of Marching Cubes (and other methods) to a finer scale so as to make all the special cases and difficult surface topologies within a single voxel break up into small collections of simple cases.

Starting with a 3-D image, that is a regular grid of volume elements representing the sampling of an intensity function $I(x, y, z)$, we examine the problem of constructing a polygonal approximation to the surfaces characterized by $I = I_0$, a constant value. This problem is also known as ***implicit surface tiling***, for $I(x, y, z) = I_0$ is an implicit equation for a surface in three-space, or as ***isosurface construction***, since an approximation to an iso-level surface is generated. Cline and Lorensen (*ibid*) developed "Marching Cubes" which partitions the space into cubical elements composed of eight neighboring voxel values. Inside each of these cells, a decision is made as to whether the surface intersects the cell, in which case a polygon approximating this intersection is constructed. The original algorithm exploited a symmetry between situations involving positive and negative voxel values. It turns out that some of the polygonal representations so obtained are not valid, i.e., there are "holes" in the surface. The basic reason for this is that some configurations of positive and negative voxel values are ambiguous. For example, on a square face of a cube, when two vertices diagonally opposite are positive and the other two negative, two different polygonalizations are possible.

Several methods have been proposed to cope with ambiguous configurations, without creating holes. Wallin [Wallin, 1991] suggests using an interpolated voxel value at the center of the face, in order to choose one polygonalization or the other. The polygons he defines in each cell can have up to 12 edges. They may not be trivially triangulated.

Kalvin (*ibid*) developed another method for disambiguation by selecting a preferred polarity. Then, by using 6-connectivity in the volume, and 26-connectivity for the exterior, the key observation is that positive vertices cannot be connected along diagonals, the decomposition to the right is the one always performed). Kalvin proves that the surfaces produced are valid. He uses the "Winged Edge" data structure [Baumgart, 1974] to represent and manipulate the surfaces, and builds them up sequentially, by applying a succession of surface construction operators.

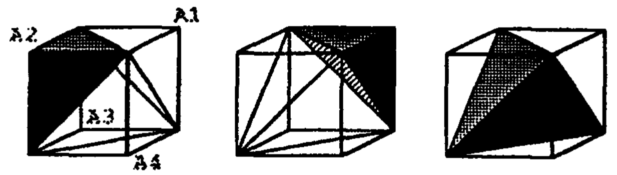

Figure 1: Decomposition of the cube into five tetrahedra. Four are right isoscele and thus isomorphic (only two are shown here on the left), the fifth one is equilateral (right).

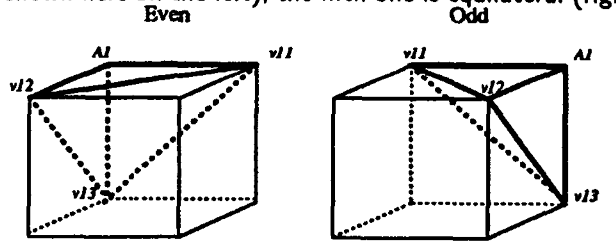

Figure 2: Tetrahedron # 1 is spanned by $(A_1, v_{11}, v_{12}, v_{13})$

Thirion (*ibid*) defines cycles in each voxel, similarly to Wallin, and shows that the cycles can be oriented. Further, he introduces oriented segments inside the cycles and a method, "Marching Lines" to draw characteristic lines on isosurfaces. The observation by Monga [Monga *et al.*, 1992], that it is possible to compute surface curvatures from the discrete differentiation of voxel values, plays a central role in the Thirion work. Marching lines has been applied to the problem of registering 3-D medical images [Ayache *et al.*, 1993].

Description of the wrapper algorithm

Tetrahedral Decomposition

We regard voxels as being values defined at points of a rectangular lattice, and eight adjacent voxels are taken to form the 8 vertices of an 8-cell, or cube. We use the same tetrahedral decomposition as Doi and Koide (see Fig. 1).

For any given cube, two such tetrahedral decompositions are possible, one which is mirror symmetric with respect to the y-z plane to the other. In order to be consistent between neighboring 8-cells, i.e., in order that faces and edges of tetrahedra in one cell match faces and edges of tetrahedra in the neighboring cells, we must alternate between the two decompositions from cell to cell. One type of cell we call an even cell, and the other type is a odd cell. Even and odd cells are composed in a checkerboard fashion. Within each 8-cell, the five tetrahedra are numbered from one to five in a consistent way. Any given tetrahedron can be described by either the cell number and tetrahedron number, or by its four vertices (v_1, v_2, v_3, v_4).

Intersection of Surfaces with Tetrahedra

The next step consists in determining whether a portion of the isosurface will intersect a given tetrahedron

$_1, v_2, v_3, v_4)$. The intensity values (I_1, I_2, I_3, I_4) cor-
sponding to the four vertices (Hounsfield numbers in
e case of X-ray Computed Tomography) are retrieved
om the 3D image.

Supposing $I < I_0$ at a vertex, we will assume that the
rtex lies outside the body bounded by the isosurface.
$I \geq I_0$, then the vertex lies inside the body. If the
rtices of a tetrahedron are of mixed sign with respect
I_0, then the isosurface will intersect the tetrahedron.
ur first task is to determine the points of intersection
ong the edges of the tetrahedron. The location of
ese points depends on the interpolation function that
used.

For each edge that exhibits an intensity sign change,
e will create a vertex of the polygonal approximation
the isosurface $I = I_0$. The exact position of the
rtex is determined by the zero-crossing of a function
terpolating intensity values along the edge. Our expe-
ence shows that using linear interpolation along each
ge results in a poor result, with excessive spikiness of
e surface. We instead choose a bilinear function that
terpolates the four intensity values at the corner of
ch face in the original volumetric grid. This allows
to evaluate values on the faces of the 8-cells; we will
ver need to explicitly evaluate values internal to the
cells. The bilinear interpolation reduces to linear in-
rpolation along the edges of an 8-cell, but results in a
adratic interpolation along diagonal edges on a 8-cell
ce. Specifically, if (a, b, c, d) denote the four intensity
lues on the face of an 8-cell then the intensity value
ong the (a, d) diagonal edge is given by:

$$I(u) = (a + d - b - c)u^2 + (-2a + b + c)u + a.$$

re u varies from 0 to 1 linearly along the diagonal.
ovided $ad < 0$, I will have exactly one zero in the
nge $0 < u < 1$, which can easily be determined from
e quadratic formula (the other zero falls outside this
nge). Having established the interpolation function
ong tetrahedral boundaries, we can thus compute the
ations of the intersections of the isosurface with those
ges.

Consider a tetrahedron (v_1, v_2, v_3, v_4) which is de-
ed by the ordered tuple of vertices spanning the vol-
e. The order is as determined by one of the five steps
m section . The corresponding intensity values are
en by a four-tuple (I_1, I_2, I_3, I_4). If all four values
ve the same sign, then the surface does not intersect
e tetrahedron. If the signs are mixed, however, we
en have three major cases. Among I_1, I_2, I_3, I_4, there
e either one, two or three positive values. These cases
e illustrated in Figure 3. For the cases I and III, three
the values have the same sign, and the surface inter-
ts the tetrahedron on three faces, corresponding to
e faces containing the vertex with the opposite sign.
r Case II, two vertices are positive and two are neg-
ve. In this case, the surface will intersect all four
es, and we have a quadrilateral. By choosing arbi-
rily a diagonal of the quadrilateral, we obtain two

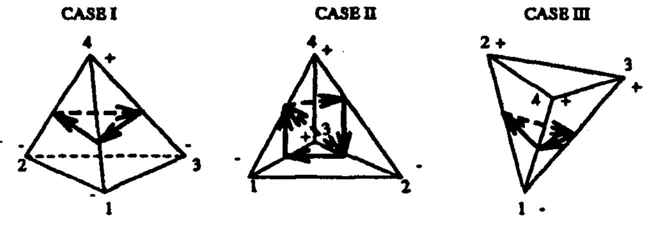

Figure 3: Defining one or two oriented triangles.

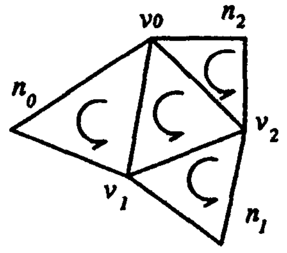

Figure 4: The neighbors of an oriented triangle t are numbered such that t and n_0 share the edge (v_0, v_1).

triangles within the tetrahedron. Combining all cases, we have the surface decomposed into either one or two triangles if it intersects the tetrahedron. An orientation for a triangle is given by a cycle, or an ordering of the edges, such that viewed from the outside, the triangle is traversed in a counter-clockwise direction.

Determining the orientation

Providing the orientation of each triangular patch is determined correctly, then the resulting surface representation consists of a collection of oriented cycles. For any given edge, there are exactly two cycles that traverse the edge. The orientation of the cycles will be such that the edge is traversed in opposite directions by the two cycles.

We have developed two methods to determine the correct orientation of the cycle generated by a tetrahedron. One method, also used by Doi and Koide, uses a determinant of the positions of the vertices of the tetrahedron, ordered according to the signs of the values at the vertices, to either reverse or leave alone a cycle. The other method precompiles all possibilities, and uses a bit code pattern determined by the sign of the vertices, together with two cases depending on whether the 8-cell is even or odd. Once the three vertices forming an oriented triangle (see Fig. 4) have been processed, the triangle is stored as three pointers to the vertices, such that the (x, y, z) coordinates of a given vertex v are physically represented only once.

Experimental Results

We applied the Wrapper, followed by a simplification algorithm not detailed here, to the representation of the cranium in a CT scan (courtesy of Court Cutting, NYU), for a pathological -namely, Crouzon syndrome- case. the result is illustrated in Fig. 5. We

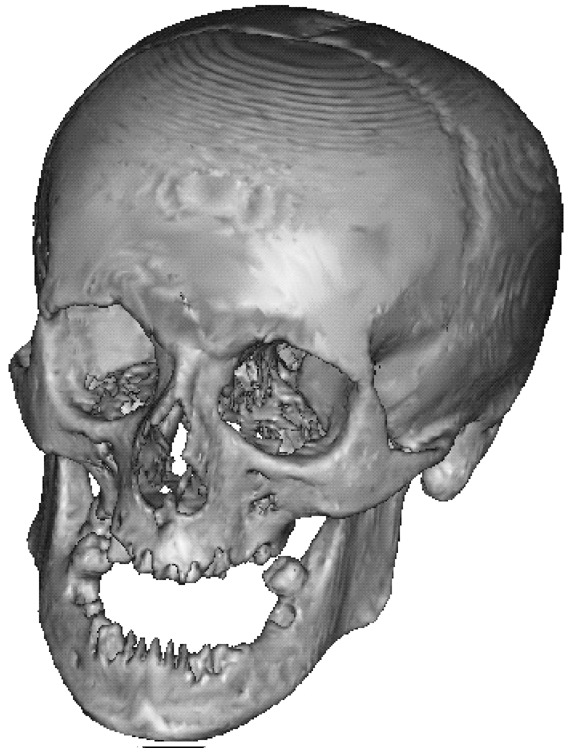

Figure 5: Surface of the cranium for a Crouzon syndrome (100K vertices and 205K triangles), extracted from a 256 by 256 by 150 CT scan. Before simplification, the original number of triangles was 1.800K. The maximum error *bound* is 1.0 millimeter.

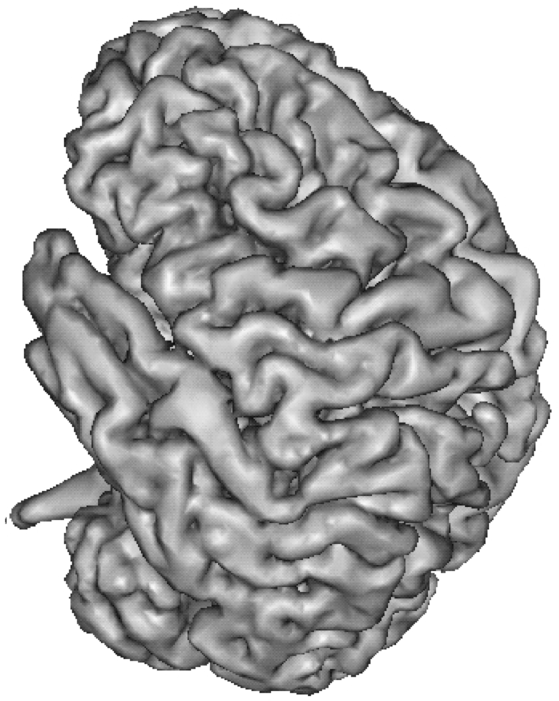

Figure 6: Cortical surface of the brain for a normal individual (31K vertices and 62K triangles), extracted from a 256 by 256 by 130 MR scan. Before simplification, the original number of triangles was 370K. Some important structures stand out, such as the Inter-Hemispheral Fissure, the Temporal Lobe, the Rolandic Fissure and the Middle Temporal Sulcus.

extracted the cortical surface of a normal individual from a MR scan (courtesy of Henry Rusinek, NYU). The result is in Fig. 6.We acknowledge Gregoire Malandain [Malandain *et al.*, 1993], for pre-processing the MR-scan.

References

[Ayache *et al.*, 1993] N. Ayache, A. Guéziec, J.P. Thirion, and A. Gourdon. Evaluating 3D registration of CT-scan images using crest lines. In *SPIE, Mathematical Methods in Medical Imaging II*, volume 2035-06, pages 60–71, July 14-15 1993. San Diego.

[Baumgart, 1974] B.G. Baumgart. *Geometric Modeling for Computer Vision*. PhD thesis, Stanford University, 1974.

[Doi and Koide, 1991] Akio Doi and Akio Koide. An efficient method of triangulating equi-valued surfaces by using tetrahedral cells. *IEICE Transactions*, E74(1):214–224, January 1991.

[Kalvin *et al.*, 1991] A.D. Kalvin, C.B. Cutting, B. Haddad, and M.E. Noz. Constructing topologically connected surfaces for the comprehensive analysis of 3d medical structures. In *SPIE, Proc. Medical Imaging V: Image Processing*, volume 1445, pages 247–258, October 1991. Bellingham, Wash.

[Lorensen and Cline, 1987] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM Siggraph*, volume 21(4), pages 163–169, July 1987. Anaheim.

[Malandain *et al.*, 1993] G. Malandain, N. Ayache, and G. Bertrand. Topological segmentation of discrete surfaces. *Inria*, 10(2):183–197, 1993.

[Monga *et al.*, 1992] Olivier Monga, Serge Benayoun, and Olivier D. Faugeras. Using third order derivatives to extract ridge lines in 3–D images. In *Proceedings of the IEEE Conference on Vision and Pattern Recognition*, Urbana Champaign, June 1992.

[Nielson and Hamann, 1991] G. Nielson and B. Hamann. The asymptotic decider: resolving the ambiguity in marching cubes. In *IEEE Conf. on Visualization*, pages 83–91, 1991.

[Payne and Toga, 1990] B.A. Payne and A.W. Toga. Surface mapping brain function on 3-D models. *IEEE Computer Graphics & Applications*, 10(5):33–41, September 1990.

[Thirion and Gourdon, 1993] J.P. Thirion and A. Gourdon. The marching lines algorithm: new results and proofs. Technical Report 1881, INRIA, 1993.

[Wallin, 1991] Ake Wallin. Constructing isosurfaces from ct data. *IEEE Computer Graphics & Applications*, 11(6):28–33, November 1991.