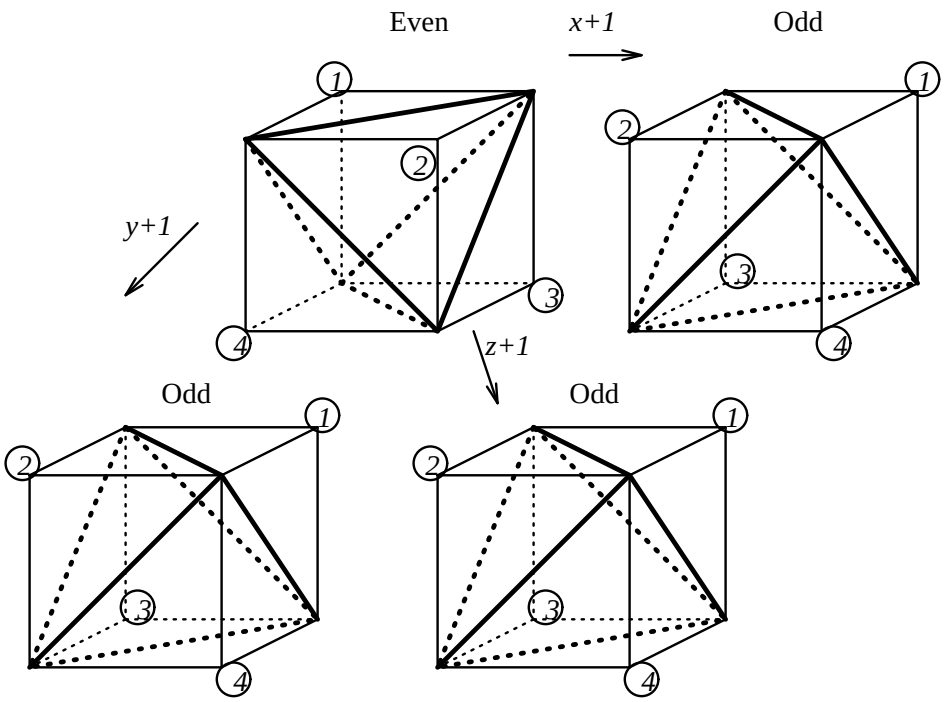

Figure 12: The correspondence between codes of neighboring tetrahedra. An "even 8-cell" (top left), where A_1 is vertex 000, is adjacent to six "odd 8-cells", where A_1 is vertex 001. The four neighbors of each of the tetrahedra will be found among these seven 8-cells.

are of interest, since tetrahedra of type I and II with zero values at the positive vertices are special cases, as discussed in Section 2.3. Recall that each type III tetrahedron has a single negative vertex (which is outside the volume), separated by a triangular patch from the tetrahedral face containing the three positive vertices, which we call the *outface*. By redefining these positive vertices to be zero, the triangular patch becomes the entire face of the type III tetrahedron.

In order to perform a rapid rendering of this approximate surface structure, we can use the following algorithm. We first locate all type III tetrahedra in the original volume, and using the tetrahedron index consisting of the (x, y, z) coordinates of the origin of the 8-cell as well as the tetrahedron number k $(1 \leq k \leq 5)$, we determine the number of the vertex opposite the outface $(1 \leq v \leq 4)$. This vertex can either be computed from the tetrahedron index and the bit code representing the vertex signs, or it can be stored in type III tetrahedra concurrent with the Wrapper Algorithm. Then, using the tetrahedron number k and the vertex v, it is easy to generate the coordinates of an oriented cycle giving

the vertices of the outface. For rendering the face, a better result is obtained if the face is assumed to have an orientation (i.e., surface normal) different than its actual direction. The phantom orientation can be taken from the orientation of the triangular surface patch in the type III tetrahedron before the perturbation, or by smooth interpolation of an orientation at each vertex obtained by averaging the surface normals of the adjoining surface elements, or by using a direction obtained from a local evaluation of the gradient of the voxel data. Figure 13 shows a result using the last method.

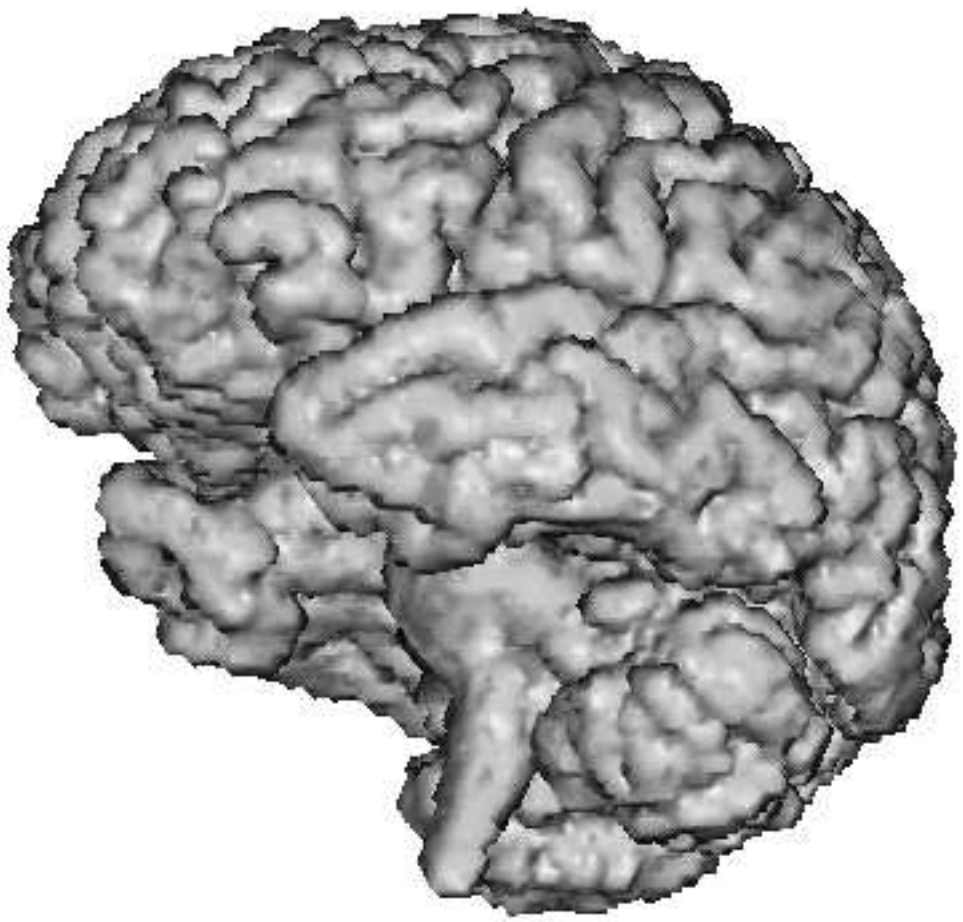

Figure 13: Approximate rendering of the cortical surface of the brain for a normal individual (104K type III tetrahedra), extracted from a 256 by 256 by 130 MR scan (courtesy of H. Rusinek, Department of Radiology, NYU School of Medicine). The surface normals are colinear with the gradient of voxel data. Prior to rendering, the segmentation of the volume uses techniques described in [MAB93].

3.3 Surface curvatures

Monga *et al.* [MBF] provide a method for computing principal curvature directions and curvature values of an isosurface directly from voxel data and spatial derivative of the voxel data (see also [Gué93]). Specifically, at a point (x_0, y_0, z_0), the normal vector $\mathbf{N}$ of the surface S defined by $I(x, y, z) \equiv I(x_0, y_0, z_0)$ lies in the same direction as the gradient of voxel data $\nabla I = (I_x, I_y, I_z)$: $\mathbf{N} = \nabla I / ||\nabla I||$.

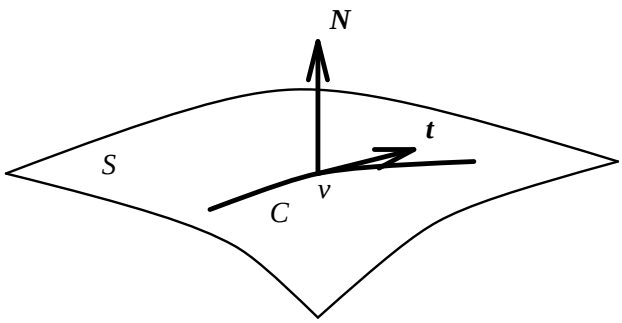

Figure 14: Normal N of the isosurface at vertex v.

In order to evaluate the surface curvatures, we consider a curve C, with tangent vector $\mathbf{t}$, along a normal section of S (see Figure 16). The normal of C is also $\mathbf{N} = (n_x, n_y, n_z)$. The differentiation of $\nabla I \cdot \mathbf{t} = \mathbf{0}$ yields

$$\mathbf{t}^t H \mathbf{t} + \nabla I \cdot k\mathbf{N} = 0$$

where k denotes the normal curvature of C which is also by definition the surface curvature in direction $\mathbf{t}$, and H is the three by three Hessian:

$$H = \begin{pmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{xz} & I_{yz} & I_{zz} \end{pmatrix}.$$

Thus $k = -\mathbf{t}^t H \mathbf{t} / ||\nabla I||$. Accordingly, the principal curvatures are the negatives of the eigenvalues of $P^T R^T H R P$ divided by the magnitude of the gradient $||\nabla I||$, where P is the projection of (x, y, z) onto (x, y) and is a three by two matrix:

$$P = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix},$$

and R is a rotation matrix composed from $R = R_1 R_2$, with

$$R_1 = \frac{1}{\sqrt{1-n_z^2}} \begin{pmatrix} n_x & -n_y & 0 \\ n_y & n_x & 0 \\ 0 & 0 & 1 \end{pmatrix}, R_2 = \begin{pmatrix} n_z & 0 & \sqrt{1-n_z^2} \\ 0 & 1 & 0 \\ -\sqrt{1-n_z^2} & 0 & n_z \end{pmatrix},$$

and R_1 is the identity if $n_z = \pm 1$. All vertices of surface patches occur along edges of 8-cells, and derivatives of the image data may be computed and interpolated at these locations by suitable methods. Solving the eigenproblem at the interpolated position of each surface vertex, we may attach curvatures (and also principal curvature directions) to each vertex stored in the surface representation. Interpolation of values into interior points of surface patches can be based on weighted sums of values at triangle vertices.

3.4 Surface simplification

The Wrapper Algorithm, based on a tetrahedral decomposition, results in significantly more triangles than the Marching Cubes method. If we count the number of vertices created by both methods, and assuming that all vertices land on the interior of edge segments, we find that the Wrapper Algorithm results in between two and three times more vertices. Experimental results for this ratio are presented in Table 4 for a variety of surface types.

In addition, the representation of the surface data has an impact on the volume of data. Alternative representations include the Baumgart winged-edge data structure [Bau74], and face or edge lists at each vertex. In most of our implementations, we choose to use a heap of vertex coordinates, with edge lists for each triangle represented through pointers into the collection of vertices. The representation that stores vertex information local to the tetrahedra, as presented in Section 2.3, is especially compact.

In order to reduce the number of triangles, and to eliminate badly shaped triangles, we have developed a simplification algorithm that we outline below. Details will be given in a subsequent report.

The simplification algorithm operates serially, successively visiting edges and testing them for removal. This approach is contrary to the methods in [SZL92,Tur92], but related to the method in [RR94]. A parallelization of the process is possible, but edges that are concurrently considered for deletion must be sufficiently distant in the graph data structure.

Figure 15 shows the configuration for an edge deletion test, where (v_1, v_2) denotes the edge that is under test. If the edge is deleted, then v_1 and v_2 are modified to terminate at v, which we call a simplified vertex.

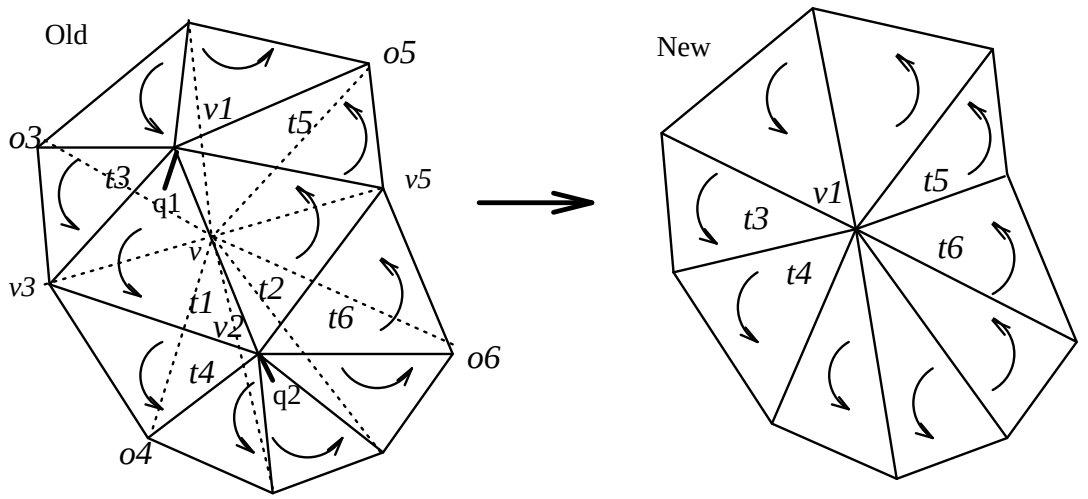

Figure 15: Removing the edge shared by triangles t_1 and t_2.

The criterion for deleting an edge can include an assortment of tests. If v_3 and v_5 denote vertices of the two triangles that contain the edge (v_1, v_2) as in Figure 15, then we demand that the distance between v_1 and v_2 be less than the distance from v_3 to v_5. Also, we might require that the former distance be sufficiently small. The Euclidean distances q_1 and q_2 between v_1 and v_2 and the new triangulated surfaces are computed. These distances are obtained by projecting v_1 and v_2 onto the closest triangle or edge of the surface, and should be small in order to pass the deletion test. The smallest angle in the new configuration of triangles is also computed and compared to the smallest angle in the old configuration. Again, we want that the new configuration improves on the old.

We maintain three positive real values d, e, and p at each simplified vertex v (see Figure 16). The value e is an upper bound to the closest distance between v and the set of triangles in the original polygonal surface, and d represents an upper bound to the distance between a triangle t that meets v and the vertices of the original surface patch (see Fig. 16). If an immediate neighbor w of v has been simplified more recently, then the value d at w takes precedence over d at v. Finally, we record in p the maximum value of q_1 and q_2, as defined above. The value of p is used in updating d and e during the

simplification process, and simplification terminates, when all values of d or e exceed a threshold.

Use of this simplification algorithm, which is incorporated into the visualizations of Figures 18, 19, 20 and 21, typically provides 10 to 1 compression ratios in the surface representation without noticeable degradation in surface fidelity. The thresholds may change dynamically, and the simplification can terminate when we obtain a given compression ratio, or when we reach at each vertex a maximum error bound.

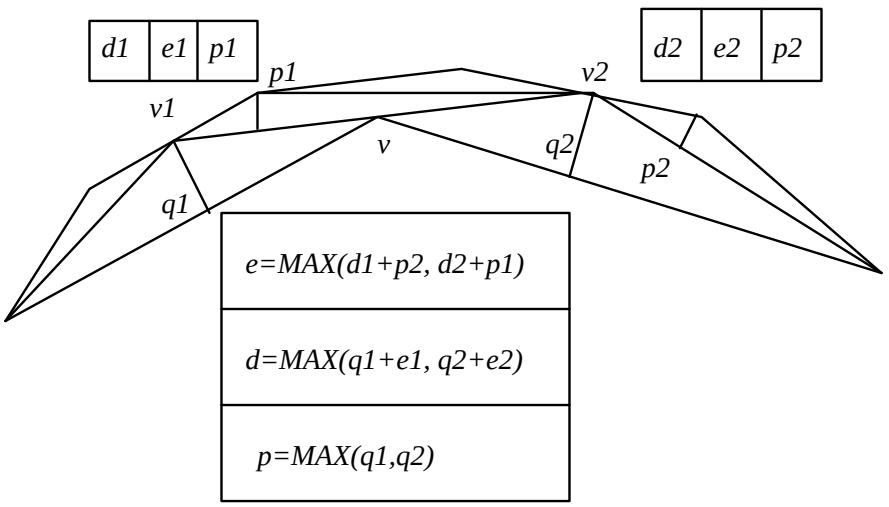

Figure 16: We associate to each simplified vertex v a maximum bound e for the distance between v and the original surface. Then d is the maximum distance between the triangles meeting at v and the original surface vertices. The value p reflects the latest simplification error when collapsing v_1 and v_2 into v.

4 The Analogy with Homology Theory

The basic idea behind the wrapper algorithm is quite simple, but has a sophisticated mathematical basis. The motivation is derived from singular theory in the mathematical field of homology theory, which in turn is part of algebraic topology. For the wrapper algorithm, it suffices to consider only q-simplices in singular theory for q equal to 1, 2, and 3.

Simply put, the idea of the wrapper algorithm is to represent the boundary of a region by a collection of oriented cycles, where each cycle defines a triangle lying in space forming a boundary element of the surface. Each triangle has three neighbors, where each neighbor is itself an oriented cycle representing a triangle. The orientations are such that any given edge in the representation is traversed twice, once in each cycle associated with the adjacent faces, and the directions are opposite for the two cycles. The upshot of the representation is that one has a graph structure representing an oriented solid, whose nodes are (oriented) triangles, and whose edges are pairs of adjacent triangles sharing a common edge.

The representation is very much akin to the Baumgart winged-edge data structure, but is specialized for triangular faces, and is augmented by the orientation information on each boundary face. Further, the wrapper algorithm makes use of a standard tetrahedral decomposition of the array of rectangular cells, and incorporates certain algorithmic efficiencies in the determination of boundary cycle orientations.

For the purpose of defining the connection to singular homology theory, and following Greenberg [Gre67], we define the simplex Δ_q to be the subset of $\Re^q$ obtained from convex combinations of (0, ..., 0), (1,0,...,0), and (0,1,0,...,0), etc. Thus Δ_1 is the line from 0 to 1, Δ_2 is the triangle (including the interior) spanned by (0,0), (1,0), and (0,1), and Δ_3 is a tetrahedron spanning the origin and unit vectors in $\Re^3$.

A singular q-simplex in $\Re^3$, for q equal to 0, 1, 2, or 3, is a continuous map from Δ_q into $\Re^3$. Roughly, a singular 0-simplex is a point, a singular 1-simplex is a line segment, a singular 2-simplex is a face, and a singular 3-simplex is a volume. When the continuous maps are affine linear, then the singular simplices are called affine, and the maps are completely defined by the images of the vertices. For singular affine simplices, all the faces and edges are flat.

The border of a singular q-simplex can be viewed as composed of singular $(q-1)$-simplices. That is, a singular 3-simplex is bordered by four singular 2-simplices, and a singular 2-simplex is bordered by three singular 1-simplices, etc. To systematize this observation, it suffices to show that the k-th border of the q-simplex ($0 \leq k \leq q$) can be seen as an affine singular $(q-1)$ simplex. To specify such an affine singular $(q-1)$ simplex, we simply define the image of the vertices of Δ_{q-1}, (0,...,0), (1,0,...,0), etc., to be the the points in $\Re^q$ (0,...,0,0), (1,0,...,0,0), etc., where the k-th vector (with the (0,...0,0)-vector counting as the zeroth) is left out. We call this affine singular $(q-1)$-simplex Δ_q^k, since it represents the k-th face of Δ_q (Figure 17).

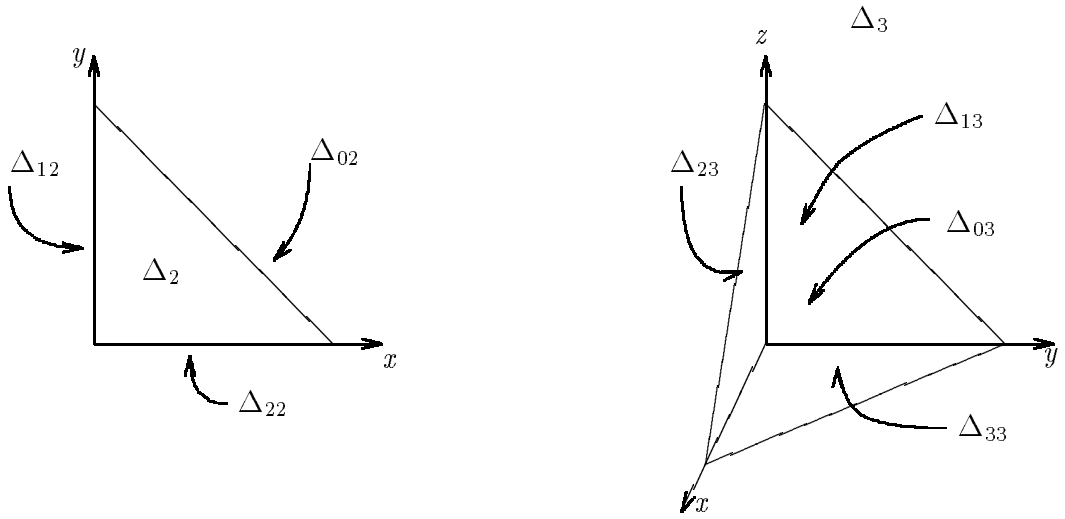

Figure 17: To specify such an affine singular $(q-1)$ simplex, we simply define the image of the vertices of Δ_{q-1}, (0,...,0), (1,0,...,0), etc., to be the the points in $\Re^q$ (0,...,0,0), (1,0,...,0,0), etc., where the k-th vector (with the (0,...0,0)-vector counting as the zeroth) is left out. We thus define the k-th face of Δ_q (here q =2,3).

A singular q-chain is a formal weighted sum (or difference) of singular q-simplices. Thus if $\sigma_1, ..., \sigma_n$ are n singular 3-simplices, their sum $\sigma_1 + \sigma_2 + ...\sigma_n$ is a singular 3-chain that represents nothing more than a "sum" of simplices. The singular simplices are allowed to have integer coefficients, either positive or negative. This is useful for defining the boundary operator. We define the boundary of Δ_q as $\sum(-1)^k \Delta_q^k$. Thus the boundary of the singular 3-simplex Δ_3 (which we can view as the identity map on Δ_3) is given by $\partial\Delta_3 = \Delta_3^0 - \Delta_3^1 + \Delta_3^2 - \Delta_3^3$. Similarly, $\partial\Delta_2 = \Delta_2^0 - \Delta_2^1 + \Delta_3^2$. Using the boundary operator defined for simplices, the boundary operator can be defined analogously for a singular q-simplex, and further extended in a linear fashion to singular q-chains.

A fundamental fact in singular theory is that the boundary operator applied to the boundary of any q-chain is zero, in the sense that all the coefficients of the resulting singular $q-2$ simplices cancel. Normally, in homology theory, we are only concerned with boundaries of q-chains, and two $(q-1)$-chains are considered equivalent if their difference is a cycle (i.e., the boundary operator applied to the difference results in the empty chain).

For the wrapper algorithm, however, we use the boundary operator to compute the boundary of a 3-chain composed of maps from the 3-simplex to tetrahedra in the volume. Specifically, we begin with a function $f(x, y, z)$ defined over a volume, which we model using data at voxel vertices and interpolate using tri-linear interpolation within the cells. We consider the shape $X = \{(x, y, z) | f(x, y, z) > 0\}$, using the assumption that f is trilinear within each cell. Using the tetrahedral decomposition of $\Re^3$, we consider a single tetrahedron T in the collection. We have:

Proposition 1 *The intersection $T \cap X$ has at most one connected component.*

The proof of this proposition depends critically on both the decomposition method and the interpolation scheme, and is part of the motivation for chosing the tetrahedral decomposition and the trilinear interpolation method. Once established, however, it assures us that the topology within any given tetrahedron is quite simple, and can be modeled as a singular 3-simplex. That is, for every tetrahedron T that meets the space X, either T is entirely contained in X, or the X portion in T is a single volume element, occupying a portion of T. But more is true:

Proposition 2 *Further, the boundary portion of X within T is either empty or a single "face," (i.e., a single component) which either meets three faces of T or all four faces of T.*

For the sake of brevity, we omit proofs of these and subsequent propositions, but instead comment on the significance.

Using these propositions, we proceed to represent X by a singular 3-chain $\sigma = \sum \sigma_j$. The main idea is that the boundary $\partial\sigma$ will be a 2-chain that represents ∂X. The 3-chain is constructed in such a way that all interior boundaries in $\partial\sigma$ cancel, and thus all singular simplices with nonzero coefficients in $\partial\sigma$ have ranges that cover ∂X. This is done as follows.

Each tetrahedron T_i is either disjoint from X, contained in X, or intersects X but is not contained in X. In the first case, no singular 3-simplex is required. In the second case, we include in the sum for σ an affine singular 3-simplex σ_i mapping Δ_3 to T_i. The orientation of σ_i matters, and depends on the identity of the tetrahedron T_i. In the third case, where ∂X meets T_i, we use:

Proposition 3 *There exists a polyhedral region Y_i in T_i that is homotopically equivalent to $X \cap T_i$ such that if E is the union of (1-D) edges of the tetrahedron T_i, then $E \cap Y_i = E \cap X$. Further, Y_i is either the range of a single affine singular 3-simplex σ_i or equals the range of a singular 3-chain containing three affine singular 3-simplicial terms $\sigma_i = \sigma_{i,1} + \sigma_{i,2} + \sigma_{i,3}$ such that the range of $\partial\sigma_i$ is ∂Y_i.*

Accordingly, the region X, using the tetrahedral decomposition, is represented by a singular 3-chain $\sum \sigma_i$, where each σ_i is either a singular 3-simplex mapping onto the tetrahedron T_i, or a singular 3-simplex mapping onto a subregion Y_i, or the sum of three singular 3-simplices whose sum maps onto a subregion Y_i. We may consider the range of σ to be a region X' that provides a polyhedral approximation of X. By construction, X' and X are homotopically equivalent, and thus have the same topology and homotopy. We then have:

Proposition 4 *The affine singular 3-chain σ may be constructed so that the range of the singular 2-simplices in $\partial\sigma$ cover $\partial X'$.*

That is, proposition 4 tells us that if the affine singular 3-simplices that are used to construct σ are chosen correctly, all of the internal faces cancel when computing the boundary. The wrapper algorithm uses Proposition 4 to construct a representation of $\partial X'$, computing the singular 2-simplices that remain in $\partial\sigma$.

Proposition 4 tells us even more. The fact that $\partial\sigma$ is a 2-cycle that lies entirely in $\partial X'$ means that σ establishes an orientation on X'. An orientation in homology theory can be precisely defined in terms of elements of relative homology modules, but it suffices to exhibit a q-chain σ such that, as in Proposition 4, $\partial\sigma$ lies in $\partial X'$. (An additional requirement is that σ must be a generator of a particular homology module, which is guaranteed in our case, because the construction of σ involves singular 3-simplices all of whose coefficients are 1 or -1.) Moreover, it is then true that $\partial\sigma$ establishes an orientation on $\partial X'$.

In our case, the orientation on $\partial X'$ can be understood simply. Since X' is a polyhedron in $\Re^3$, $\partial X'$ is a 2-manifold, and an orientation on X' is established if for every $x \in X'$, there is a cycle defined around x such that the direction of the loops "match up" for neighboring x. For the interior of the faces represented in $\partial\sigma$, the orientation for the corresponding face of X' is established in its entirety by the singular 2-simplex that covers the face. That is, suppose that τ is a singular 2-simplex in $\partial\sigma$, which covers a triangular face of X'. Then $\partial\tau$

is a singular 1-chain, containing three singular 1-simplices, which are simply oriented lines circling the border of the triangular face. The direction of the cycle gives the orientation of the face. To define an orientation of the entire 2-manifold $\partial X'$, it is only necessary to observe that the orientations of the faces "match up." That is, if we have two neighboring triangular faces on $\partial X'$, then the orientations of the two must be compatible, in the sense that if the two cycles about the triangles are combined, a single cycle about the pair of triangles results which maintains the same sense as the original cycles. This is assured, however, because $\partial\partial\sigma = 0$, which means that if an edge e is traversed in one direction along a face, then it is traversed in the opposite direction (the coefficient of the singular 1-simplex is -1) on the adjacent face. Accordingly, if the edge is removed, then the points along the edge belong to a quadrilateral (a conjoined pair of faces), and the orientation of the points along the edge is defined by the cycles about the quadrilateral.

5 Experimental Results

We applied the Wrapper Algorithm to the representation of the cranium in CT scans at full resolution. The Figures 18 and 19 represent a normal individual, from of the CMNH Hamman-Todd collection. Figure 21 represents a pathological case, an individual with Crouzon's syndrome, from the collection of the Vienna Museum of Natural History.

We extracted the cortical surface of a normal individual from an MR scan (courtesy of H. Rusinek, Department of Radiology, NYU School of Medicine). The result is in Fig. 20. We thank Gregoire Malandain [MAB93] for applying his segmentation algorithms to the MR-scan. Rendering in these images is done using an orthographic projection of the polyhedral structure. We employ Gouraud shading, using surface normals in the direction of the gradient of voxel data. The color display (Fig. 21) uses the curvature computation of section 3.3. The maximum principal curvature is color-coded, such that high curvature regions appear in red and low curvature regions appear in blue. Green and yellow areas indicate medium curvature values. Derivatives of voxel data are estimated by convolving image data with B-spline functions [Gué93].

We gratefully acknowledge Drs. Court Cutting and David Dean for helpful discussions with the authors. We thank Bruce Latimer, Director, Laboratory of Physical Anthropology, Cleveland Museum of Natural History, for access to the Hamman-Todd morgue collection.

References

[AGTG93] N. Ayache, A. Guéziec, J.P. Thirion, and A. Gourdon. Evaluating 3D registration of CT-scan images using crest lines. In *Mathematical Methods in Medical Imaging II*, volume 2035-06, pages 60–71, San Diego, California, July 14–15 1993. SPIE, The International Society for Optical Engineering.

[Bak89] H. Harlyn Baker. Building surfaces of evolution: The weaving wall. *The International Journal of Computer Vision*, 3:51–71, 1989.

[Bau74] B.G. Baumgart. *Geometric Modeling for Computer Vision.* PhD thesis, Stanford University, 1974.

[CLR89] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms.* Mac Graw Hill, 1989.

[DK91] 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.

[Gre67] Marvin J. Greenberg. *Lectures on Algebraic Topology.* W.A. Benjamen, Reading, Massachussets, 1967. Mathematics Lecture Notes Series.

[Gué93] André Guéziec. Large deformable splines, crest lines and matching. In *Fourth International Conference on Computer Vision (ICCV 93)*, Berlin, May 1993. IEEE.

[Hum86] Robert A. Hummel. *Connected Components Labelling in Image Processing with MIMD Architectures*, chapter Seven, pages 101–127. Intermediate-Level Image Processing. Academic Press, 1986.

[HW90] Mark Hall and Joe Warren. Adaptive polygonalization of implicitly defined surfaces. *IEEE Computer Graphics & Applications*, 10(6):33–42, November 1990.

[Kal91] Alan Kalvin. *Segmentation and Surface-Based Modeling of Objects in Three-Dimensional Biomedical Images.* PhD thesis, New York University, June 1991.

[Kar92] Dan B. Karron. The "SpiderWeb" algorithm for surface construction in noisy volume data. In R.A. Robb, editor, *Visualization in Biomedical Computing' 92*, volume 1808, pages 462–476, 1992. SPIE, Society of Photo-Optical Instrumentation Engineers.

[KCHN91] Alan Kalvin, Court Cutting, Betsy Haddad, and Marylin Noz. Constructing topologically connected surfaces for the comprehensive analysis of 3D medical structures. In *Medical Imaging V: Image Processing*, volume 1445, pages 247–258, Bellingham, Washington, October 1991. SPIE, The International Society for Optical Engineering.

[KDK86] Akio Koide, Akio Doi, and Koichi Kajioka. Polyhedral approximation approach to molecular orbital graphics. *J. Molecular Graphics*, 4:149–160, 1986.

[KT93] Alan D. Kalvin and Russel H. Taylor. Superfaces: Polyhedral approximation with bounded error. Technical report, IBM T.J. Watson Research Center, Yorktown Heights, New York, 1993.

[LC87] 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, Anaheim, California, July 1987.

[MAB93] Gregoire Malandain, Nicholas Ayache, and Gilles Bertrand. Topological segmentation of discrete surfaces. *International Journal of Computer Vision*, 10(2):183–197, 1993.

[MBF92] Olivier Monga, Serge Benayoun, and Olivier Faugeras. Using third order derivatives to extract ridge lines in 3D images. In *Conference on Vision and Pattern Recognition*, Urbana Champain, Illinois, June 1992. IEEE.

[NB93] Paul Ning and Jules Bloomenthal. An evaluation of implicit surface tilers. *IEEE Computer Graphics & Applications*, 13(6):33–4, November 1993.

[NH91] G. Nielson and B. Hamann. The asymptotic decider: Resolving the ambiguity in marching cubes. In *Conference on Visualization*, pages 83–91. IEEE, 1991.

[PT90] 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.

[RR94] Rémi Ronfard and Jarek Rossignac. Simplifying a triangular mesh with multiple planar constraints. Technical report, IBM T.J. Watson Research Center, Yorktown Heights, New York, 1994.

[Sch93] Larry L. Schumaker. Triangulations in CAGD. *IEEE Computer Graphics & Applications*, 13(1):47–52, January 1993.

[SZL92] Willian Schroeder, Jonathan Zarge, and William E. Lorensen. Decimation of triangular meshes. In *ACM Siggraph*, volume 26-2, pages 65–70, July 1992.

[TG93] J. P. Thirion and A. Gourdon. The marching line algorithm: New results and proofs. Technical Report 1881, INRIA, Sophia Antipolis, France, 1993.

[TR93] Gabriel Taubin and Rémi Ronfard. Implicit simplicial models I: Adaptive curve reconstruction. Technical report, IBM T.J. Watson Research Center, Yorktown Heights, New York, 1993.

[Tur92] Greg Turk. Retiling polygonal surfaces. In *ACM Siggraph*, volume 26-2, page 55064, July 1992.

[Wal91] Ake Wallin. Constructing isosurfaces from CT data. *IEEE Computer Graphics & Applications*, 11(6):28–33, November 1991.

[WMW86] G Wyvill, C. McPheeters, and B. Wyvill. Data structures for soft objects. *Visual Computer*, 2:227–234, 1986.

Bit code	Index	Sign of det() in an even cell
0000	0	+
0001	1	+
0010	2	−
0011	3	+
0100	4	+
0101	5	−
0110	6	+
0111	7	+
1000	8	−
1001	9	+
1010	10	−
1011	11	−
1100	12	+
1101	13	+
1110	14	−
1111	15	+

Table 1: Tabulating the number of transpositions in order to map from the bit code representing the signs of the values at the tetrahedra vertices to the sign of the determinant that determines whether the constructed cycle has the correct orientation (negative determinant) or must be reversed (positive determinant). For an odd cell, the entries are opposite those of an even cell.

Bit code	e_1	e_2	e_3
0000			
0001	1	2	
0010	2		1
0011		2	1
0100		1	2
0101	1		2
0110	2	1	
0111			
1000			
1001	2	1	
1010	1		2
1011		1	2
1100		2	1
1101	2		1
1110	1	2	
1111			

Table 2: Storage locations for edges e_1, e_2, and e_3, as a function of the bit code on the tetrahedron containing the secondary storage location. A "1" entry indicates that the primary storage location should be used for the edge, whereas a "2" denotes the secondary storage. A blank indicates that the edge does not have a zero-crossing with the corresponding bit code.

Tetrahedron		Neighbors $(\Delta x, \Delta y,\ \Delta y, k)$			
Even or Odd	Number	1	2	3	4
Even	1	$(-1,0,0,1)$	$(0,-1,0,2)$	$(0,0,-1,3)$	$(0,0,0,5)$
Even	2	$(1,0,0,2)$	$(0,1,0,2)$	$(0,0,-1,4)$	$(0,0,0,5)$
Even	3	$(1,0,0,3)$	$(0,-1,0,4)$	$(0,0,1,1)$	$(0,0,0,5)$
Even	4	$(-1,0,0,4)$	$(0,1,0,3)$	$(0,0,1,2)$	$(0,0,0,5)$
Even	5	$(0,0,0,1)$	$(0,0,0,2)$	$(0,0,0,3)$	$(0,0,0,4)$
Odd	1	$(1,0,0,1)$	$(0,-1,0,2)$	$(0,0,-1,3)$	$(0,0,0,5)$
Odd	2	$(-1,0,0,2)$	$(0,1,0,2)$	$(0,0,-1,4)$	$(0,0,0,5)$
Odd	3	$(-1,0,0,3)$	$(0,-1,0,4)$	$(0,0,1,1)$	$(0,0,0,5)$
Odd	4	$(1,0,0,4)$	$(0,1,0,3)$	$(0,0,1,2)$	$(0,0,0,5)$
Odd	5	$(0,0,0,1)$	$(0,0,0,2)$	$(0,0,0,3)$	$(0,0,0,4)$

Table 3: List of four neighboring tetrahedra for each type of tetrahedron.

Algorithm	Ellipsoid	Sphere		I=rand()		Cranium		Brain
Wrapper	1720	3102	6200	9228	186703	37594	65752	65668
Marching Cubes	808	1328	2648	4432	93976	15784	27500	30004
Ratio	2.12	2.33	2.34	2.08	2.09	2.38	2.39	2.18

Table 4: Experimental result for the ratio between number of vertices created by the marching cubes and the Wrapper. The ratio is relatively invariant to the sampling rate in the volume.

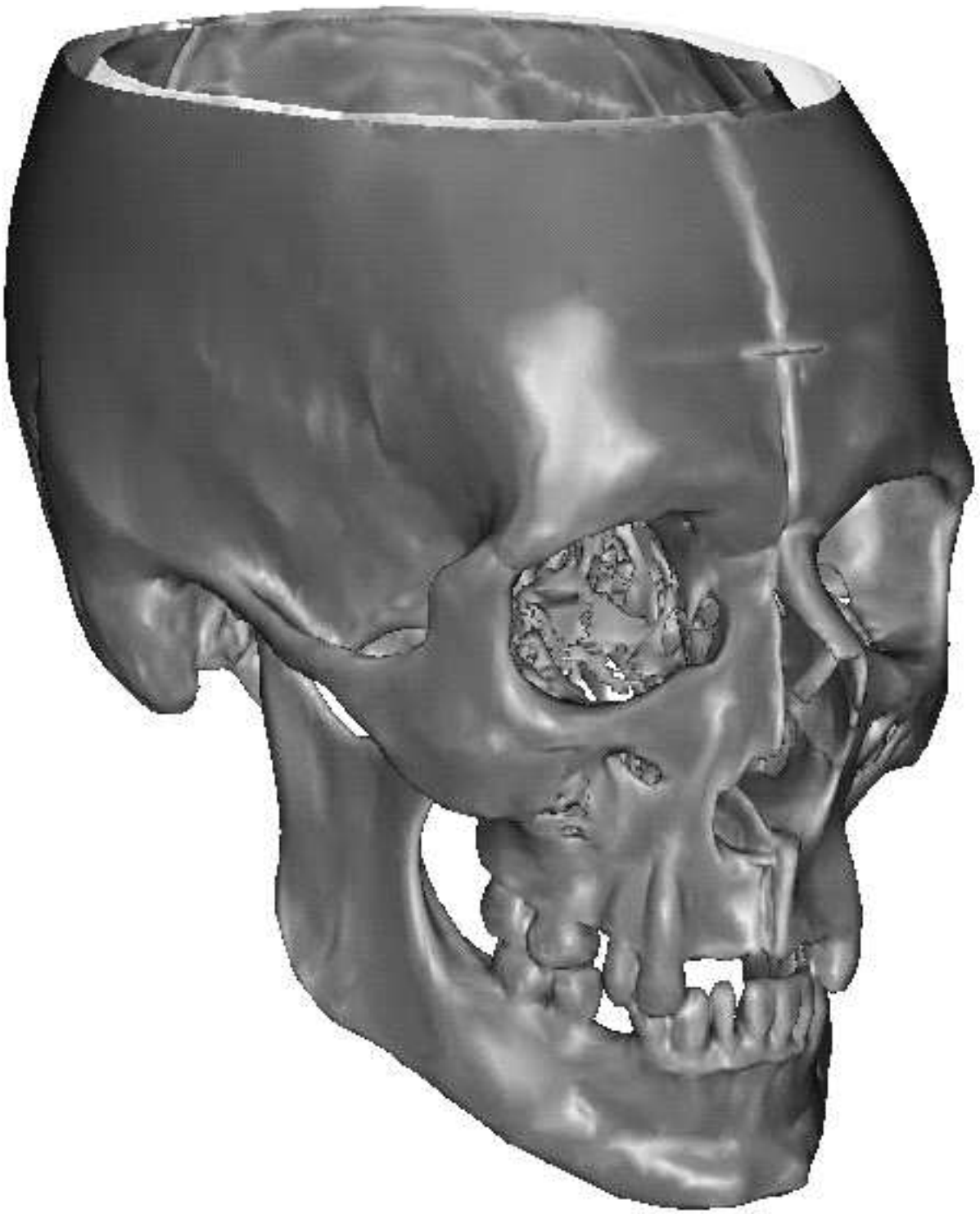

Figure 18: Model of cranium CMNH # 1331, comprising 66K vertices and 129K triangles, extracted from an original 512 by 512 by 150 CT scan. The total number of triangles before simplification was 3.450K, which corresponds to a compression ratio of 27. The maximum error bound is 0.5 millimeter.