Published in final edited form as:
Comput Med Imaging Graph. 2011 April ; 35(3): 206–219. doi:10.1016/j.compmedimag.2010.11.002.

An Improved Representation of Regional Boundaries on Parcellated Morphological Surfaces

Xuejun Hao, **Dongrong Xu**, **Ravi Bansal**, **Jun Liu**, and **Bradley S. Peterson**
MRI Unit, Psychiatry Department, Columbia University & the New York State Psychiatric Institute

Abstract

Establishing the correspondences of brain anatomy with function is important for understanding neuroimaging data. Regional delineations on morphological surfaces define anatomical landmarks and help to visualize and interpret both functional data and morphological measures mapped onto the cortical surface. We present an efficient algorithm that accurately delineates the morphological surface of the cerebral cortex in real time during generation of the surface using information from parcellated 3D data. With this accurate region delineation, we then develop methods for boundary-preserved simplification and smoothing, as well as procedures for the automated correction of small, misclassified regions to improve the quality of the delineated surface. We demonstrate that our delineation algorithm, together with a new method for double-snapshot visualization of cortical regions, can be used to establish a clear correspondence between brain anatomy and mapped quantities, such as morphological measures, across groups of subjects.

Keywords

region delineation; parcellation; marching-cubes; surface construction; surface simplification; B-spline curve; visualization; surface rendering

1. INTRODUCTION

The primary goals of human brain mapping are to integrate brain anatomy with measures of brain function and behavior and to establish correspondences between them. Sulci and gyri on the surface of the cerebrum, for example, are anatomical landmarks that help to define the locations of major functional areas. Labeling and, ultimately, parcellation of geometric features of the cortical surface is thus important for analyzing and visualizing both functional and structural neuroimaging data. Mapping of functional activations or morphological measures onto parcellated cerebral cortex across ages, for example, can improve understanding of brain development.

Individual variability in the folding patterns of each individual cortex makes the automated identification and labeling of cortical structures challenging. Various techniques have been

Address Columbia College of Physicians & Surgeons, Columbia University and the New York State Psychiatric Institute, 1051 Riverside Drive, Unit 74, New York, NY 10032. **Phone** 212-543-1905 (Xuejun Hao), 212-543-5495 (Dongrong Xu), 212-543-6145 (Ravi Bansal), 212-543-5207 (Jun Liu), 212-543-5330 (Bradley S. Peterson) **Fax** 212-543-0522 HaoX@childpsych.columbia.edu (Xuejun Hao).

developed to address this difficulty, especially the automated delineation of sulci. They can be categorized broadly as either a surface-based [1–8], graph-based [9,10], or volumetric-region-based approach [11]. One algorithm, for example, warps a pre-labeled brain atlas onto the surface model of each participant's brain to establish the correspondences between them that permit labeling of each voxel as a particular tissue class and a specific region of interest (ROI) [2]. Another algorithm models the sulci as vertices and the relation between sulci as arcs, and then assigns labels to the identified sulci using a manually generated training set [10]. Still another applies a watershed algorithm on the cortical surface and then manually labels identified sulci [5].

The automatic labeling of tissues and parcellation of the brain, however, requires quantitative knowledge of the geometric variation in brain anatomy, as well as its critical functional interfaces. A manual parcellation of the cortex by an experienced neuroanatomist can take advantage of much additional information, such as cytoarchitectonic labeling properties and knowledge of structure-function relationships. The inclusion of such prior information is critically important for the success of any algorithm for cortical parcellation [6].

Anatomical correspondence is especially important at functional interfaces and cytoarchitectonic boundaries [1,12]. A parcellated cortical surface that has a clear delineation of regional boundaries can be used for mapping onto the cortex and regionally delineating functional imaging data and measures for morphological differences, thus revealing the correspondences and spatial relationships between brain structure and function. The general approaches to this problem, however, focus on the parcellation itself, rather than on generating and manipulating the boundaries of the regional delineation of the cortical surface that the parcellation produces. These approaches generally aim to associate a label to each surface element (e.g. a triangle or quadruple) on an existing surface without adjusting the surface geometry. The region delineations thus generated, though satisfactory for many purposes, can be imprecise because the assumption may not always be true that the actual regional boundary coincides with the existing discrete surface geometry. Moreover, these approaches make difficult the control of precision for further processing of the delineated surface, such as smoothing of a regional boundary and surface simplification.

We propose an efficient algorithm for the delineation of cortical surfaces using manually parcellation volumes generated previously by experienced neuroanatomists. Our program takes as input two 3D volumetric data -- one that is a scanned intensity volume, and the other that is a parcellated categorical volume coinciding spatially with the first and that encodes all the expert knowledge of the parcellation. Our algorithm integrates surface delineation with surface generation and guarantees precise coincidence of regional boundaries with surface geometry. We extend the marching-cubes algorithm [13] and generate the cortical surface while simultaneously delineating cortical regions by adaptively subdividing surface triangles that are located on regional boundaries. We then develop various automatic algorithms for surface processing, such as boundary-preserved surface simplification and smoothing of regional boundaries under user-specified error estimates. In addition, we design a new visualization algorithm that displays regional boundaries clearly, even when portions of the curves that define the region are buried inside sulci and are not visible from a particular viewing perspective.

2. METHODS

Parcellation of a surface requires the accurate partitioning of that surface into disjoint regions according to the anatomical or functional sub-structures that each of the regions represents, and with computationally efficient visualization of the regions that are thus

generated. A surface is normally represented graphically as a mesh of triangles. Region delineation thus partitions the triangular mesh into a set of disjoint sub-meshes. This partition then can be visualized by assigning different colors to different sub-meshes or by superimposing onto the mesh curves that mark the boundaries between regions.

2.1 Partition of a Surface into Regions

A surface is defined in a 3D volumetric data space V by specifying the conditions that points on the surface must satisfy. For example, the cortical surface of a brain can be generated from the iso-value contour of gray matter that has been defined on an MR image. The surface thus generated is called the iso-surface.

Let V be a continuous 3D space, $S(a)$ be an iso-surface having the iso-value a, x be a point in V, and the volumetric data defined continuously over V be intensity $I(V)$, then $S(a)$ is the collection of points that have the intensity value a:

$$S(a) = \{x | x \in V, I(x) = a\}$$

An intensity volume such as $I(V)$ is usually defined by a set of continuous values that have not been classified previously into different sub-volumes or subregions. This classification can be performed using a wide range of procedures, such as the manual or automatic, mathematically based segmentation of $I(V)$. Let $C(V)$ be another volumetric data space defined in the same space V that partitions V categorically into the collection of disjoint sub-spaces $V_1, V_2, \ldots V_n$, such that the points in each sub-space V_i have the same categorical value C_i. The delineation of a surface $S(a)$ according to $C(V)$ maps each point x on the surface to a categorical value C_i. This process defines the segmentation of V.

The assignment of categorical values to a surface is straightforward if volumetric data $I(V)$ and $C(V)$, as well as the surface S, are continuous. One needs only to determine in which sub-space V_i each point x on S belongs. If either one or both of $I(V)$ (and thus $C(V)$) and S are discretely sampled, however, and if they produce a discontinuity of the sampled points on S with the sampled points on V, then the mapping of C_i to x will not be obvious and will need to be more precisely defined. Here we assume that the set of values of $I(V)$ and $C(V)$ are given at the same discrete locations in the space where they overlap. This is the case when $C(V)$ is the segmented version of $I(V)$, which is the consequence of manual or automatic segmentation approaches. If $C(V)$ is generated by a different procedure than is $I(V)$, however, as when $C(V)$ is defined by cytoarchitectonic maps of one subject and $I(V)$ is defined on a T1-weighted MR image, then these two volumes must be coregistered into the same space, where they will be discontinuous with one another. The points in the region where these volumes overlap must be sampled before a segmented surface $S(a)$ of $I(V)$ can be delineated by $C(V)$.

2.1.1 Extension of the Marching-Cubes Algorithm—To delineate a surface $S(a)$ that is represented as a collection of triangles, we must map each triangle within $S(a)$ to a categorical region C_i. Assigning categorical regions to a triangle's three vertices is relatively easy if we map those vertices to the categorical label of the nearest voxel, as defined by volume $C(V)$. A triangle T, however, can have vertices that belong to differing categories -- i.e., that have differing values of C_i. In that instance, the segmentation of T cannot be determined from this simple mapping. Moreover, even if all three vertices of T do belong to a single category, a portion of T still could belong to another category.

A voxel in a discretely sampled space represents a unit volume of that space. The shape of this unit volume depends on the method used to sample the space. With a homogeneous

orthogonal sampling, the space is divided evenly into unit cubic volumes. We assign sampled values at each unit volume to its center and call it a voxel. Thus a voxel of $C(V)$ having a categorical label C_i represents a unit volume of space (Figure 1a) that belongs to subspace V_i, and all voxels of $C(V)$ together impose upon the space a discrete partition. To identify the category to which a triangle belongs, one needs only to query the voxels of $C(V)$ that enclose the triangle. If the triangle is fully enclosed by one or more voxels having the same value C_i, the triangle simply is assigned this single categorical label C_i. If a portion of the triangle crosses voxels of differing C_i (Figure 1b), however, then we cannot assign its category unambiguously, and so we will need to subdivide this triangle into smaller ones until each of them lies entirely in a single subspace.

Marching-cubes is a popular algorithm used to generate an iso-surface $S(a)$ from a rectangular sampled volume $I(V)$, in which each sampled point is called a voxel. The algorithm is computationally efficient, employing a divide-and-conquer approach in treating separately each unit rectangular cube that composes the volume and that is defined by eight neighboring voxels positioned at its corners. The algorithm assesses whether the sampled value in each cube matches the iso-value of the surface to determine whether the cube intersects the iso-surface. The algorithm has an efficient linear $O(N)$ complexity, where N is the number of voxels in the data volume. One advantage of using the marching-cubes algorithm for delineation of a surface is that each triangle is guaranteed to lie in the cubic space enclosed by the eight voxels at the corners of the cube (Figure 1b). Thus, we need only to identify the labels C_i at these eight voxels to determine the label at the triangle. For surfaces generated using other algorithms, even though our method remains applicable, more voxels may need to be assessed, the consequence being a longer computational time.

The eight voxels at the corners of the unit cubic volume of the marching-cubes algorithm inherently divides the cubic volume into eight subregions, each of which belongs to the corresponding voxel (Figure 2). The division occurs at the boundaries of the unit volume that is represented by these voxels, which are three orthogonal planes positioned midway between the voxels, dividing the space into eight equally sized portions. If the labels differ across these eight voxels, then these three planes can be used to subdivide a triangle into several smaller triangles, each belonging to a single voxel and thus being assigned a single segmentation label.

We extend the marching-cubes algorithm by incorporating surface delineation through dynamically labeled triangles. Our extension and other contributions in this report can be briefly summarized as follows:

- **(1)** Dynamic assignment of labels to surface triangles using triangular subdivisions, if required, during surface generation
- **(2)** Simplification of the surface thus generated so as to reduce the number of triangles while preserving a valid regional interface
- **(3)** Smoothing of the regional interface using B-spline curves under user-controlled error thresholding and smoothing
- **(4)** Redistributing vertices during the smoothing of the regional interface

In addition, we develop a double-snapshot algorithm for visualizing the labeled surfaces with other information superimposed. Our algorithm is implemented in C++, and visualization is realized using OpenGL.

An iso-value triangle generated from a unit cubic volume in $I(V)$ is used to identify the label C_i for each of the eight voxels that define the corresponding cubic space in $C(V)$. As noted above, if all eight voxels belong to the same category, we need no further processing and

simply assign this categorical value to the triangle. If, on the other hand, these labels are not all the same, then the triangle is on the regional boundary and must be subdivided. We subdivide a triangle on a boundary by identifying how it intersects the three planes that equally divide the unit cubic volume enclosing the triangle. Each possible intersection splits the triangle into one or more smaller triangles (Figure 3). Then we assign the corresponding labels at the eight voxels to the subdivided triangles.

A non-boundary triangle may need to be re-tessellated if it lies adjacent to one or more boundary triangles that have been subdivided (Figure 4). The re-tessellation of these non-boundary triangles depends on how many edges they share with boundary triangles and how those shared edges are subdivided.

Our adaptation of the marching-cubes algorithm uses the same divide-and-conquer approach as the original algorithm. It therefore has a similar linear O(N) complexity in terms of the number of voxels, adding only a constant factor to that efficiency associated with subdividing boundary triangles and re-tessellating triangles that are adjacent to subdivided boundary triangles. As an algorithm that needs to scan all its input to determine which operation to perform on its input will require a running time no better than a linear function of N, as each input has to be at least read once. Our algorithm has a worst-case running time that is a linear function of its input size (the number of voxels), thus is computationally efficient, and requires no more than a few seconds of CPU time. The memory requirements of our adapted algorithm, however, are higher than those of the original. Additional memory is needed for the load-in of the volume data $C(V)$, whereas the original algorithm requires the loading of $I(V)$ alone.

2.1.2 Simplification of the Boundary-Preserved Surface—The surface generated by our adapted marching-cubes algorithm will have many new vertices and new triangles compared with the surface generated by the original marching-cubes algorithm. The increased number of triangles slows the speed of surface rendering, which may be undesirable for applications that require user interaction in real time. To improve interactivity of the algorithm during display, we must reduce the overall number of vertices, and thus also the number of representational triangles, to the number present if no regional subdivisions were required. It is preferable if this number can be reduced further without sacrificing the quality of visualization.

Simplification of the graphical representation of a surface has been studied extensively by researchers in interactive graphics [14]. Surfaces are generally simplified by reducing the number of representational triangles, which requires reducing the number of vertices using methods such as edge collapse (Figure 5). The number of triangles is reduced while preserving the quality of surface display according to criteria such as the retention of overall shape and curvature of the surface. An additional important criterion for our purposes is to preserve the regional boundaries that were generated from the adapted marching-cubes algorithm.

Our algorithm for surface simplification is based on edge collapse using quadric error metrics (QEM) [15]. QEM compute the area-weighted sum of squared distances of a new vertex position to a set of planes neighboring the edge that is collapsed. Simplification of an algorithm using QEM can produce high quality approximations while retaining the overall shape and curvature of the original surface, as the quadric error relates directly to surface curvature [16].

To preserve the regional boundaries, we enforce the following two rules during the simplification process:

(1) Only non-boundary vertices can be merged into a boundary vertex, but not vice versa.

(2) A boundary vertex P may be merged into another boundary vertex, provided that the merge will not change the local shape of the regional boundary involving P. As an example, if P_a, P_b, and P_c are three consecutive vertices on a boundary that separates two regions, and if they are collinear, then vertex P_b can be merged into either vertex P_a or P_c.

One can easily see that rule (1) does not change boundary vertices at all, whereas rule (2) removes boundary vertices without affecting the shape of the boundary. In this way we maintain a high resolution mesh around the regional boundary while lowering resolution of the mesh in regions where the quality of visualization is affected little by the simplification.

2.1.3 Perturbation of the Vertex Location for the Smoothing of Regional Boundaries—A regional boundary is a curve that separates regions belonging to differing segmentation categories. This boundary will be a smooth curve if the surface is continuous. For discrete surface that are represented as a collection of triangles, however, a regional boundary is defined as a series of line segments, each of which is an edge shared by two triangles belonging to differing segmentation categories. The regional boundaries thus defined are usually zigzag in contour rather than smooth (Figure 6), a direct consequence of the discrete sampling of $C(V)$. We can improve the smoothness of these regional boundaries, however, through a perturbation of the locations of vertices on the boundary that is based on the overall curvature of the line segments that compose the boundary. With the accurate region boundaries defined in section 2.1.1, our smoothing algorithm can provide a user complete and precise control over the smoothed curves.

The smoothing procedure consists of two steps. First, we construct a smooth, curvilinear approximation of the boundary to be smoothed. Second, we locally perturb the vertices of the boundary and then move them toward this approximation curve.

We have elected to use one kind of cubic polynomial—a uniform, non-rational B-spline [17], to approximate the boundary curve. Cubic polynomial representations of curves offer tangent continuity (i.e., curves at the joint point share the same tangents, or identical first parametric derivatives) and flexibility in controlling the shape of the representation. In addition to tangent continuity, B-splines provide continuity of curvature (i.e., identical second parametric derivatives) at the joint point and thus are inherently smoother than general cubic polynomial curves.

Another important property of B-splines is the local control that they provide, in that B-splines are curvilinear segments whose polynomial coefficients depend on just a few local control points, so that moving a control point affects only a small portion of the curve. This local control property greatly reduces the time required to compute the polynomial coefficients. More importantly, this local control provides better constraints on the error induced during smoothing of the curve. Because each segment of a B-spline curve is governed by four control points, their blending functions sum to one and are everywhere nonnegative; thus the curve segment is constrained to the convex hull of its four control points.

Following the construction of B-spline curves using the vertices on the boundary as control points, we then perturb the location of each vertex on the boundary toward this curve by moving each vertex onto the curve (Figure 7a&b). With the local control property, we need only test against the curve segments that have this vertex as a control point for moving a

particular vertex, and then we ensure that this movement will not flip the orientation of any triangles thus involved, so that local topology will not be altered.

We estimate the smoothness of a curve using the turning angles at vertices along the curve (i.e., the angle formed between the left and right tangent lines). The turning angle will equal zero for a continuous curve with tangent continuity, as these two lines will always coincide. For a discrete curve consisting of line segments, the left tangent line is approximated by the line that connects the vertex to the vertex's predecessor along the boundary curve, and the right tangent line is approximated by the line connecting the vertex to its successor (Figure 8). The angle will then be nonzero, and its value will indicate the local smoothness of the boundary curve at the measured vertex.

Multiple iterations of the B-spline smoothing algorithm provide progressively improved smoothing of regional boundary curves. Nevertheless, the overall smoothness of a boundary curve might still be less than ideal at locations where the curve takes sharp turns, regardless of the number of iterations of the algorithm, because the desirable movement of vertices at these locations for ideal smoothing might flip the orientation of neighboring triangles (Figure 9).

One solution to this flipping difficulty is to adjust the position of neighboring vertices of a boundary vertex during the smoothing process. For example, one can treat each vertex as a positive point charge. The movement of a boundary vertex will then produce newly unbalanced repulsion forces among neighboring vertices and produce a redistribution of their positions. This redistribution creates room for further movement of the boundary vertex during the next iteration of smoothing. The redistribution, however, is also constrained in that it will not change the position of a boundary vertex -- i.e., movement of a boundary curve will redistribute its neighboring non-boundary vertices, but not the vertices on the curve itself, nor vertices on other boundary curves. Thus the redistribution is a local property of our algorithm. Our implementation of redistribution in this report has adopted a Laplacian operator [18], which will redistribute an affected vertex to the barycentric position of all its neighboring vertices.

To enable complete and precise control over the smoothing procedure, we assess at each iteration of the perturbation the distance of the perturbed vertex to the vertex's nearest regional boundary plane, as defined in section 2.1.1. Our algorithm stops perturbing a vertex when the error induced at the vertex approaches a user-defined, preset threshold. The preset threshold is determined by the precision required by each particular application and usually governed by the sizes of interesting areas or regions. Our program takes the size of half voxel as the default threshold if no value is provided by the user. This default threshold guarantees that the boundary curve lies between voxels belonging to different categories, instead of cutting space of voxels belonging to the same category.

The total number of iterations for smoothing is thus determined by the balance between the smoothness of the boundary (in terms of its average turning angle) and the increase in error that is induced by the perturbation of the positions of the boundary vertex. Both of these criteria are pre-established by the user. For example, in our first sample of smoothing of the cortical surface shown in Section 3, we stipulated the maximum error to be less than 0.5mm (half the width of a voxel) and the desired smoothness to be 15 degrees (approximately 0.27 radians). These pre-established thresholds yielded a termination of the smoothing algorithm after 50 iterations.

The final output of our algorithm for representing a surface parcellation thus consists of a mesh of triangles on that surface, each of which is assigned a unique segmentation label, with smooth regional boundaries represented by lists of edges.

2.1.4 Automated Correction of Small, Misclassified Regions—Human error in the manual parcellation of a brain is unavoidable. The brain is a highly complicated structure, and the signal-to-noise ratios of imaging data are usually not ideal. To make the parcellation task even more error-prone, the 3D viewing angle provided by interactive editing tools to neuroanatomists for manual parcellation is usually limited to only a few directions. Errors in parcellation produce a misclassification of small regions on the parcellated surface that range from several to tens of voxels in size.

We have included in our algorithm an automatic procedure to correct these small misclassified surface areas. From the constructed curves of regional boundaries, we identify those curves that are self-closed. The curve lengths are then compared against a pre-established threshold, below which the curve-enclosed area is labeled as misclassified and its categorical value is changed to the same value as the area outside of the curve. This threshold must be designated on an ad hoc basis by a knowledgeable user. For example, in the thalamic surface shown in Section 3, a threshold of 10mm was used to remove small, misclassified regions, assuming that no functional sub-regions have sizes smaller than 3mm in diameter. Another decision of a user is the number of delineated regions to require. The algorithm will rank the regions according to their surface area and remove the excessive number of small, self-looped regions with ranks below the user-defined number.

2.2 Visualizing the Parcellated Surface

To visualize the final parcellation, different colors can be assigned to regions defined by differing segmentation labels (Figure 15). The colored surface thus represented will have a clear delineation of the parcellation units [6] if the colors between each pair of neighboring regions have discernable contrast. This method is simple and effective if displaying only the final parcellation. When used together with the display of other measures on the morphological surface, however, as when representing functional activations or other statistical parameters using color encoding, this colored regional display will likely confuse the visual interpretation of those measures. In this case, an alternative is needed to display the parcellated surface. One option is to display the regional boundaries using a color that is located outside of the color gamut used for display of the statistical parameters and that also contrasts well with the representation of the background surface.

The explicit display of the regional boundaries can create another problem, however. A boundary curve usually is not visible in its entirety, even though the two regions that it delineates are clearly visible, because a portion of the boundary curve might in some 3D viewing perspectives be located behind one of the regions that it delineates. This is especially true for boundary curves on the cortical surface that are located deep within sulci. Surface flattening provides one solution to this problem of boundary visualization [19]. Flattening procedures, however, are computationally complex, and interpretation of the flattened surfaces is non-intuitive. An alternative solution for surfaces that have not been flattened is sometimes desirable.

We have developed a double-snapshot display algorithm that combines the advantages of a colored display of surface regions with the display of boundary curves to represent parcellation units. A colored regional display uses different colors to provide an implicit definition of boundary curves. We make this boundary explicit by tracing the implicit boundary curves and superimposing them onto a uniformly colored surface to provide a complete delineation of the parcellated surface. The algorithm works as follows:

For every possible viewing perspective,

(1) Generate a 2D snapshot of the surface in which each triangle is displayed with a uniform intensity that is proportional to its label value. No lighting is yet applied to the surface representation.

(2) Obtain the explicit boundary curves from the 2D snapshot by tracing out the individual segments of the implicit curve that delineates two adjacent regions having different intensity values.

(3) Generate another lighted 2D snapshot of the surface in which the other quantitative measures (in our example, statistical parameters) are mapped onto the surface as colored regions.

(4) Superimpose the boundary curves from step (2) onto the snapshot from step (3), replacing the values of pixels on the boundary with a new, pre-selected color value to mark the boundary.

(5) Display the superimposed snapshots from step (4).

Display of the parcellated surface using this double-snapshot algorithm requires approximately twice the computational time as a conventional display, as the same surface is displayed twice, in steps (1) and (3). Steps (2) and (4), however, require very little computational time.

2.3 Performance on a Simulated Dataset

Application of our algorithm to a simulated dataset provides a known ground truth to evaluate its performance and to justify the choice of its operational parameters, such as the number of smoothing iterations to employ. Moreover, we sample the simulated dataset at various resolutions and investigate the effects of partial volume averaging on the performance of our algorithm.

We generate an ellipsoid of size comparable to a human brain:

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2} \le 1; (a=72.5mm, b=40.5mm, c=91.5mm)$$

where the surface is defined by the equal sign of the equation. The portion of the ellipsoid inside a half cone is classified as region one and the portion that is outside of the cone is classified as region two. The cone is given by:

$$(x-a_1)^2+(y-b_1)^2=[r(z-c_1)]^2; (a_1=45mm, b_1=15mm, c_1=15mm, r=1.5; z \le c_1)$$

The boundary curve on the ellipsoid surface separates region one and region two. Essentially it is the intersection of the ellipsoid surface with the cone surface (Figure 10). Its curvature varies continuously from $7.13m^{-1}$ to $47.30m^{-1}$.

Volume $I(V)$ is generated at resolutions of 1, 2, 3, 4, 6, and 8mm, respectively. Each voxel of the volume is assigned an intensity value that measures the distance from the ellipsoid surface, with a value of zero defining the surface of the ellipsoid. The classification volume $C(V)$, generated at the same resolution as $I(V)$, coincides with $I(V)$. A voxel of C(V) inside the ellipsoid (with an intensity value less than or equal to zero) is assigned a classification value based on whether or not more than half of the volume of the voxel is inside the cone. Voxels of $C(V)$ outside of the ellipsoid are assigned a classification value of zero.

Given $I(V)$ and $C(V)$ as inputs, our algorithm generates the surface of a regionally delineated ellipsoid. We measure for each point on our generated regional boundary the distance to the true regional boundary as defined by the intersection of the surfaces of the ellipsoid and cone. We then evaluate the performance of our algorithm under differing values of its operational parameters.

3. RESULTS AND DISCUSSION

Computer Simulations

For volumes at a resolution of 1mm, performance of our algorithm that redistributed neighboring vertices during smoothing iterations, compared with performance of our algorithm without redistribution, was superior on all categories tested: average distance of smoothed boundary to the true boundary, average turning angle, maximum distance of the smoothed boundary to the true boundary, maximum turning angle (Figure 11). With an increasing number of smoothing iterations, the average turning angle and average distance of the smoothed boundary to the true boundary decreased monotonically. Moreover, the boundaries that were generated using a redistribution of neighboring vertices during smoothing converged to lower values than did the boundaries generated without redistribution (average turning angles of 0.018 vs. 0.055 radians, and an average distance of the smoothed boundary from the true boundary of 0.092mm vs. 0.107mm, respectively). The maximum distance of the smoothed boundary from the true boundary generated using the redistribution approach decreased as iterations increased, up to a value of approximately 50 iterations, and thereafter remained constant. Both the maximum turning angle and the maximum distance from the true boundary generated without redistribution of voxels decreased for the first few iterations and thereafter remained constant. As noted earlier, ideal smoothing cannot be achieved at some sharp turning points without flipping the orientation of their neighboring triangles, unless neighboring vertices are redistributed. Thus these simulation findings suggest that the redistribution schema is useful for smoothing curve segments that take sharp turns and that an iteration count of approximately 50 provides optimal performance.

We also note that the simulation suggests that performance of our algorithm does not depend on the boundary's curvature (Figure 12).

Based on our findings using the simulated dataset at a resolution of 1mm, we assessed the performance of our algorithm on simulated datasets at varying spatial resolutions using the redistribution of neighboring voxels during smoothing of the boundaries and an iteration count of 50 (Figure 13 and Table 1). Both the average and maximum distances of the smoothed boundary from the true boundary increased nearly linearly with decreasing resolution, as expected. However, the distances at a resolution of 4mm approximated those generated using the original marching-cubes algorithm at a resolution of 1mm. Moreover, both the average and maximum turning angle generated by our algorithm at a resolution of 8mm are much smoother than the boundaries generated by the original marching cube at a 1mm resolution (Figure 14 and Table 1). One unexpected finding is that the maximum turning angle decreased as resolution increased from 1 to 8mm (Figure 13b). This finding can be visualized by noting that with increasing voxel size, the surface has fewer vertices and each of those vertices has more unoccupied adjacent space to adjust their positioning during boundary smoothing. Similarly, the misclassified surface area generated by our algorithm at a resolution of 4mm approximated those generated using the original marching-cubes algorithm at a resolution of 1mm (Table 1).

Real-World MRI Datasets

We next show the output of our algorithm on a single subject template brain with its associated demarcated parcellation labels from the International Consortium for Brain Mapping (ICBM) [20]. This template is the average of 27 high resolution, T_1-weighted MRI acquisitions from a single subject, aligned within the stereotaxic space of the ICBM average template. The 3D parcellation of cortical gyri, subcortical nuclei, and cerebellum was defined manually by an expert neuroanatomist.

Approximately 611,000 triangles were needed to completely represent the cortical surface using the original marching-cubes algorithm, with roughly 1/5 of those triangles (i.e., 135,466) located at regional boundaries. In contrast, approximately 1,440,376 triangles were required using our adapted marching-cubes approach that subdivides triangles along regional boundaries, a 140% increase over the number required using the original algorithm. We then applied our algorithm for the boundary-preserved simplification of the newly generated mesh, and a standard simplification algorithm to the original mesh, so that the both meshes had equal number of triangles following application of the algorithms, which was approximately half of the count from the original marching-cubes algorithm. In addition, we applied the B-spline smoothing of the regional boundary curves and corrected small misclassified regions. Less than one minute was required to generate, delineate, simplify, and smooth the parcellated surface when running on an Intel Xeon 3.0GHz PC with 2GB RAM.

The regional delineation using our algorithm is clearly more accurate than are approaches that simply assign labels to a previously generated surface mesh without an adaptive subdivision of that mesh (Figure 15). The report on average turning angle over all boundary vertices, as well as the maximum and average distance over all boundary vertices to their nearest region boundary plane as defined in section 2.1.1 confirms the advantage of our algorithm (Table 2). The average turning angle is reduced from 1.636 radians to 0.908 radians using our accurate region delineation method, and further down to 0.271 radians after smoothing. It represents a factor of six improvements. In addition, both the maximum and average distances are improved by a factor of approximately two.

Moreover, our double-snapshot algorithm achieved a clear display of the improved regional parcellations. Regional boundary curves generated by the first snapshot (Figure 16a) clearly demarcate the parcellation units (Figure 16b&c). The transfer of the boundary curves back to the second lighted snapshot (Figure 16d) gives a final clear visualization of the parcellation units (Figure 16e), contrasts well with the results generated by a regular visualization method (Figure 16f).

Our algorithm achieved similar improvements on right thalamus (Figure 17). Thalamus surface was delineated using volumetric parcellated histological maps [21]. As before, the average turning angle is largely reduced (a factor of 12), accompanied by the decrease in the maximum, as well as the average distance deviated from the actual boundaries (Table 3).

We applied our complete algorithm for region delineation and visualization to the display of results from a previous morphological study [22]. At each point on the cerebral surface, the statistical significance (probability values) of differences in cortical thickness across groups (high vs. low risk in major depressive disorder) in participants from 2^{nd} and 3_{rd} generations are color coded and displayed with and without overlay of the delineation of regional boundaries (Figure 18). The statistical models accounted for the degree of familial relatedness, as well as for the age and sex, of all participants. The p-values were thresholded at $p<0.05$ after correction for multiple comparisons using the theory of Gaussian Random

Fields on a two-dimensional manifold [23]. The images that display the regional delineation provide exquisite anatomical localization of group differences.

We have presented results for the delineation and visualization of parcellation units on the cortical surface and thalamus surface. Our algorithm, however, can be applied to the representation of a parcellation scheme on any surface, as long as that parcellation has been previously defined. Our algorithm can also be applied to the representation of surfaces generated by means other than the marching-cubes algorithm, as our algorithms for region definition and visualization are independent of the particular methods for graphical representation that is used to generate the surface display.

Our delineation algorithm, however, requires that a 3D volume parcellation is previously defined over the same space where the surface is to be generated. If the parcellation exists in another space, then it has to be transformed into the space of surface definition and transformation errors might be induced and degrade the accuracy of the delineation results.

Our paper assumes an accurate parcellation of the volumetric space where the surface is generated. The assumption could be a limitation of our algorithm. In addition to the deterministic atlases our algorithm is designed for, there exist probabilistic atlases that assign each voxel some probability values to specify the chances of the voxel belonging to each category. For such data, the optimum separation of regions will not be at the middle plane separating voxels. Moreover, the optimum separation will not even be a plane at all. To extend our algorithm for these atlases, a curved surface cutting schema will be required at the surface generation stage. Moreover, to accurately estimate the error will be more difficult during the simplification and smoothing stages as a collinearity of points along the boundary curve is no longer a quality guarantee. We plan to explore the problem in the future. In addition, different expert neuroanatomists might parcellate data slightly different and render the surface regions thus defined not identical. We would also like to investigate the reproducibility of surface delineation limited by the volumetric-parcellation reproducibility in the future.

4. CONCLUSION

We have presented an algorithm for the improved delineation of parcellation units on a 3D morphological surface. Our algorithm is both accurate and computationally efficient. Moreover, our new algorithm for visualization of regional boundaries using a double-snapshot technique helps to display clearly and unambiguously both the parcellation units and other quantitative measures of interest, while avoiding the computational demands and interpretive difficulties associated with surface flattening algorithms. Our algorithms can be applied easily to the representation of parcellation schemes across a wide variety of morphological surfaces to aid the representation and interpretation of data.

Acknowledgments

This work was funded in part by NIMH grants K02 74677, MH089582, MH068318, 1P50MH090966, NIEHS grant ES015579, NIDA grants DA027100 and DA017820, and a grant from the National Alliance for Research in Schizophrenia and Affective Disorders (NARSAD).

Biographies

Xuejun Hao is an associate research scientist at Columbia College of Physicians & Surgeons, Columbia University and the New York State Psychiatric Institute. He received his Ph.D. in Computer Science from University of Maryland, College Park in 2004.

Dongrong Xu is an assistant professor at Columbia College of Physicians & Surgeons, Columbia University and the New York State Psychiatric Institute. Prior to join Columbia University, he had extensive research experience at Stony Brook University, Johns Hopkins Univeristy, and University of Pennsylvania. He obtained his Ph.D. in Computer Science from Zhejiang University in 1995.

Ravi Bansal is an associate professor at Columbia College of Physicians & Surgeons, Columbia University and the New York State Psychiatric Institute. He obtained his Ph.D. in Electrical Engineering from Yale University in 2000.

Jun Liu is an assistant professor at Columbia College of Physicians & Surgeons, Columbia University and the New York State Psychiatric Institute. He obtained his Ph.D. in Statistics from Rutgers University in 2000.

Bradley S. Peterson is Suzanne Crosby Murphy Professor, Director of Child & Adolescent Psychiatry, and Director of MRI Research at Columbia College of Physicians & Surgeons, Columbia University and the New York State Psychiatric Institute. He obtained his M.D. from University of Wisconsin-Madison Medical School in 1987.

REFERENCES

1. Thompson PM, Schwartz C, Toga AW. High-resolution random mesh algorithms for creating a probabilistic 3D surface atlas of the human brain. NeuroImage. 1996; 3(1):19–34. [PubMed: 9345472]
2. Sandor S, Leahy R. Surface-based labeling of cortical anatomy using a deformable atlas. IEEE Trans. Med. Imaging. 1997; 16(1):41–54. [PubMed: 9050407]
3. Le Goualher G, Collins L, Barillot C, Evans A. Automatic identification of cortical sulci using 3D probabilistic atlas. Lecture Notes in Computer Science. 1998; 1496:509–18.
4. Lohmann G. Extracting line representations of sulcal and gyral patterns in MR images of the human brain. IEEE Trans. Med. Imaging. 1998; 17(6):1040–8. [PubMed: 10048861]
5. Rettmann ME, Han X, Xu C, Prince JL. Automated sulcal segmentation using watersheds on the cortical surface. NeuroImage. 2002; 15(2):329–44. [PubMed: 11798269]
6. Fischl B, van der Kouwe A, Destrieux C, Halgren E, Segonne F, Salat DH, Busa E, Seidman LJ, Goldstein J, Kennedy D, Caviness V, Makris N, Rosen B, Dale AM. Automatically parcellating the human cerebral cortex. Cereb. Cortex. 2004; 14(1):11–22. [PubMed: 14654453]
7. Makris N, Schlerf JE, Hodge SM, Haselgrove C, Albaugh MD, Seidman LJ, Rauch SL, Harris G, Biederman J, Caviness VS Jr. Kennedy DN, Schmahmann JD. MRI-based surface-assisted parcellation of human cerebellar cortex: an anatomically specified method with estimate of reliability. NeuroImage. 2005; 25(4):1146–60. [PubMed: 15850732]
8. Desikan RS, Segonne F, Fischl B, Quinn BT, Dickerson BC, Blacker D, Buckner RL, Dale AM, Maguire RP, Hyman BT, Albert MS, Killiany RJ. An automated labeling system for subdividing the human cerebral cortex on MRI scans into gyral based regions of interest. NeuroImage. 2006; 31(3): 968–80. [PubMed: 16530430]
9. Mangin J-F, Frouin V, Bloch I, Régis J, López-Krahe J. rom 3D magnetic resonance images to structural representations of the cortex topography using topology preserving deformations. J mathematical imaging and vision. 1995; 5(4):297–318.
10. Le Goualher G, Procyk E, Collins L, Venegopal R, Barillot C, Evans A. Automated extraction and variability analysis of sulcal neuroanatomy. IEEE Trans Med Imag. 1999; 18(3):206–17.
11. Lohmann G, von Cramon DY. Automatic labelling of the human cortical surface using sulcal basins. Med. Image Anal. 2000; 4(3):179–88. [PubMed: 11145307]
12. Rademacher J, Caviness VS Jr. Steinmetz H, Galaburda AM. Topographical variation of the human primary cortices: implications for neuroimaging, brain mapping, and neurobiology. Cereb. Cortex. 1993; 3(4):313–29. [PubMed: 8400809]

13. Lorensen W, Cline H. Marching cubes: a high resolution 3D surface reconstruction algorithm. Computer Graphics. 1987; 21(4):163–9.
14. Luebke, D.; Reddy, M.; Cohen, J.; Varshney, A.; Watson, B.; Huebner, R. Level of Detail for 3D Graphics. Morgan-Kaufmann, Inc; 2003.
15. Garland M, Heckbert PS. Surface simplification using quadric error metrics. SIGGRAPH97 Proceedings. August.1997 :209–216.
16. Heckbert PS, Garland M. Optimal triangulation and quadric-based surface simplification. Computational Geometry. 1999; 14(1):49–65.
17. Bartels, R.; Beatty, J.; Barsky, B. An Introduction to Splines for Use in Computer Graphics and Geometric Modeling. Morgan Kaufmann; Los Altos: 1987.
18. Hermann LR. Laplacian-Isoparametric Grid Generation Scheme. J Engineering Mechanics Division. 1976; 102(5):749–56.
19. Fischl B, Sereno MI, Tootell RBH, Dale AM. High-resolution intersubject averaging and a coordinate system for the cortical surface. Human Brain Mapping. 1999; 8(4):272–84. [PubMed: 10619420]
20. Mazziotta J, Toga A, Evans A, Fox P, Lancaster J, Zilles K, Woods R, Paus T, Simpson G, Pike B, Holmes C, Collins L, Thompson P, MacDonald D, Iacoboni M, Schormann T, Amunts K, Palomero Gallagher N, Geyer S, Parsons L, Narr K, Kabani N, Le Goualher G, Boomsma D, Cannon T, Kawashima R, Mazoyer B. A probabilistic atlas and reference system for the human brain: International consortium for brain mapping (ICBM). Philos Trans R Soc Lond Ser B Biol Sci. 2001; 356(1412):1293–322. [PubMed: 11545704]
21. Chakravarty MM, Bertrand G, Hodge CP, Sadikot AF, Collins DL. The Creation of a Brain Atlas for Image Guided Neurosurgery using Serial Histological Data. NeuroImage. 2006; 30(2):359–76. [PubMed: 16406816]
22. Peterson BS, Warner V, Bansal R, Zhu H, Hao X, Liu J, Durkin K, Adams PB, Wickramaratne P, Weissman MM. Right hemisphere cortical thinning in persons at increased familiar risk for major depression. Submitted to PNAS; in revision.
23. Bansal R, Staib LH, Xu D, Zhu H, Peterson BS. Statistical analyses of brain surfaces using Gaussian random fields on 2D manifolds. IEEE Trans Med Imag. 26(1):46–57.

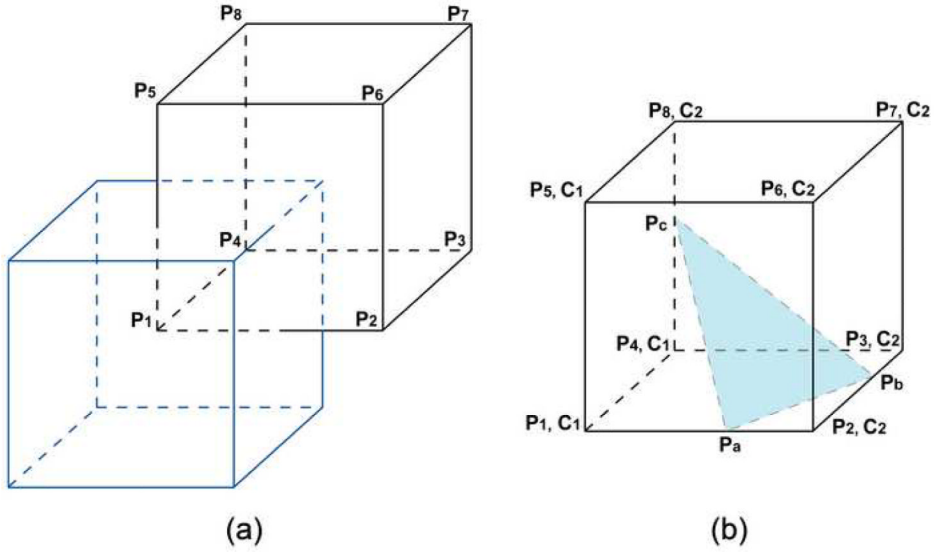

FIGURE 1. Voxel Representation and Triangle Labeling

(a) a voxel P_1 represents the unit volume centered at P_1 (its boundary is marked by blue lines); (b) a triangle with corner points at (P_a, P_b, P_c) is categorically labeled according to the categorical values at voxels P_1, P_2, …, P_8 enclosing the space where the triangle locates. Here P_1, P_4, and P_5 belong to category C_1, while the other five voxels belong to category C_2. We show later that this triangle needs to be sub-divided in order to unambiguously determine its categorical label.

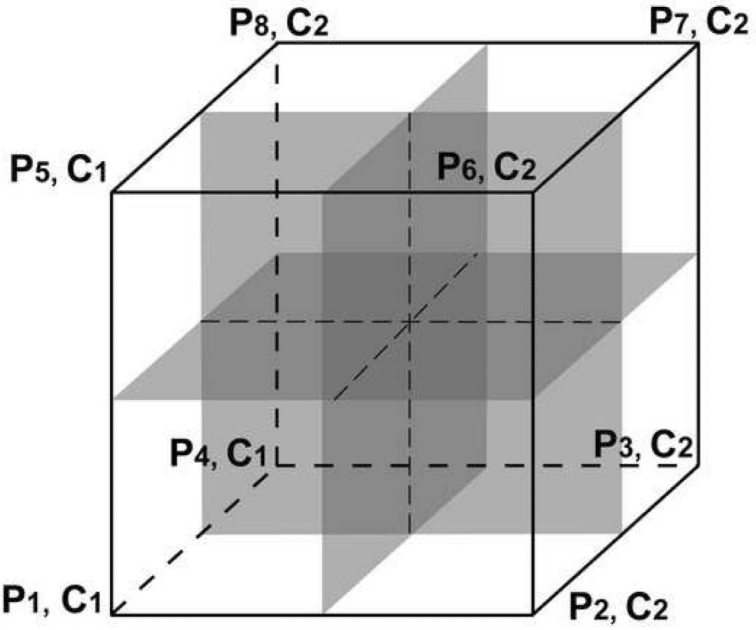

FIGURE 2. Subdivision of Unit Cubic Space

A unit cubic space enclosed by eight adjacent voxel centers is divided into eight disjoint sub-volumes. The division happens at three planes perpendicular to each other and each cuts the cubic space into two equal-sized halves.

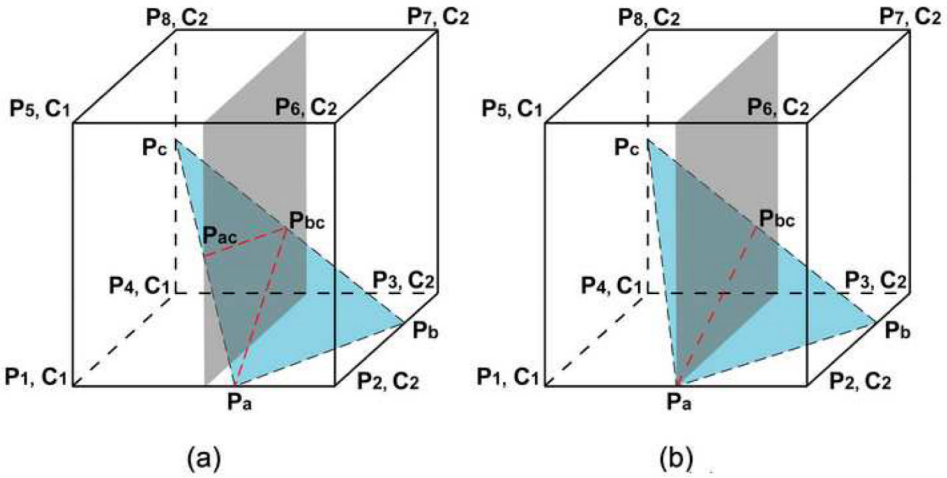

FIGURE 3. Subdivision of a boundary Triangle

A triangle is subdivided according to its intersections with the three planes that subdivide the cubic space containing the triangle. (a) a plane subdivides triangle (P_a, P_b, P_c) into three smaller triangles: (P_a, P_b, P_{bc}), (P_a, P_{bc}, P_{ac}), and (P_c, P_{ac}, P_{bc}); (b) a plane subdivides triangle (P_a, P_b, P_c) into two smaller ones: (P_a, P_b, P_{bc}) and (P_a, P_{bc}, P_c) since vertex P_a coincides with the intersecting plane.

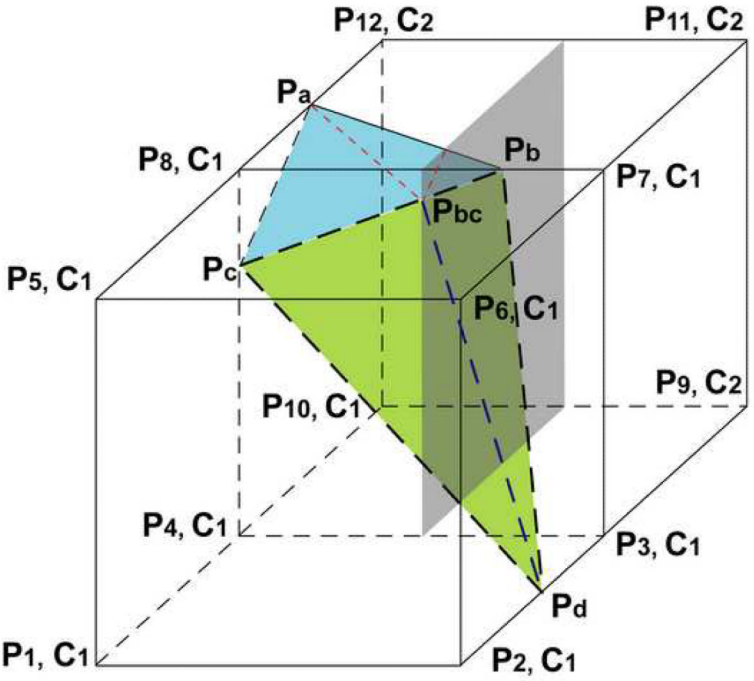

FIGURE 4. Re-tessellation of a non-boundary Triangle

Triangle (P_b, P_c, P_d), though enclosed by voxels (P_1, P_2, …, P_8) having the same categorical value C_1, has to be splitted into triangles (P_c, P_{bc}, P_d) and (P_d, P_{bc}, P_b) since it is adjacent to a boundary triangle (P_a, P_b, P_c) and the common edge between them, (P_b, P_c) has been subdivided.

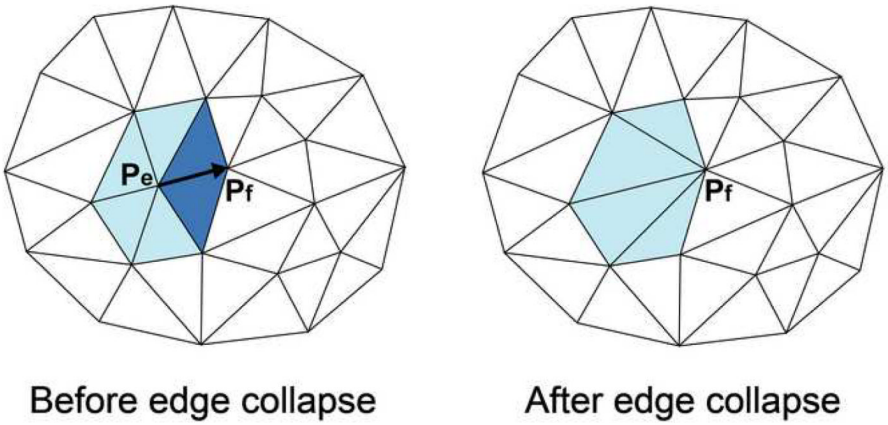

FIGURE 5. Surface Simplification through Edge Collapse

An edge (P_e, P_f) is removed where vertex P_e is merged with vertex P_f that results in the removal of vertex P_e and the removal of two triangles (colored in dark blue) sharing the edge (P_e, P_f).

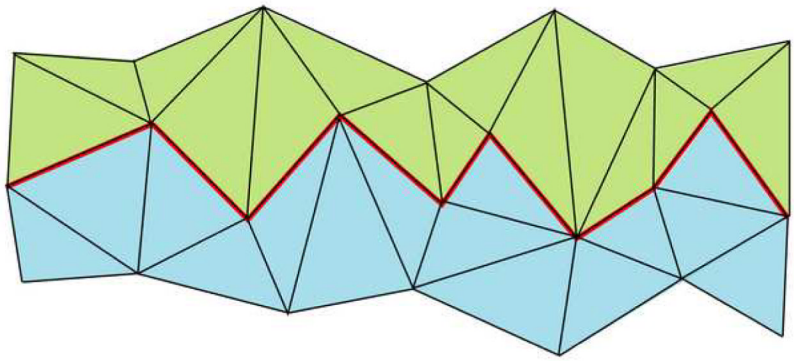

FIGURE 6. Region Boundary

A region boundary is a series of joint line segments, each being an edge shared by two triangles that belong to different categories. Here the red line segments represent the boundary between two regions marked by blue and green respectively. The boundary has a zigzag appearance due to the discrete nature of surface, and thus discrete curve representation.

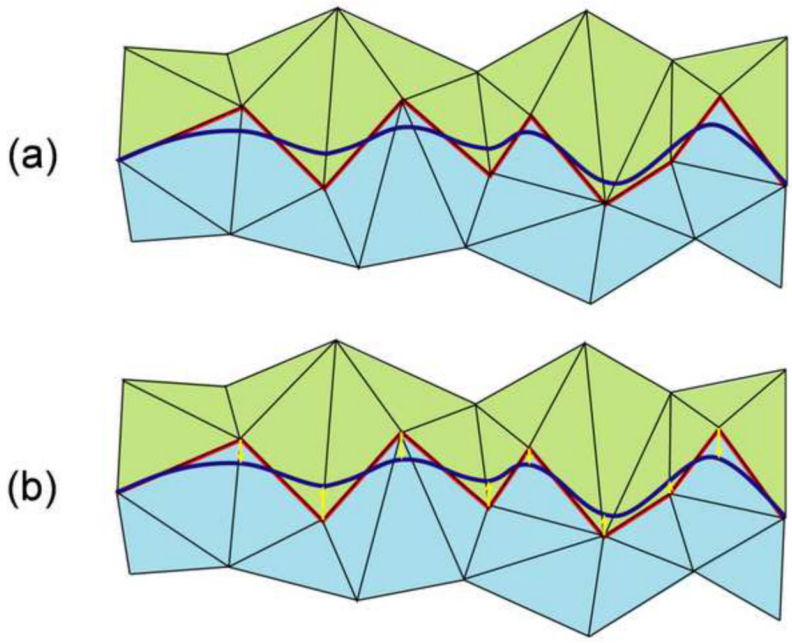

FIGURE 7. Perturbation of Boundary Vertices' Positions

Vertices are moved toward the B-spline fitted curve of a regional boundary. (a) B-spline fitting (curve in dark blue) of the boundary line segments (in red) and boundary vertices are moved toward (yellow arrows) the B-spline curve; (b) Boundary vertices are now on the B-spline curve and the boundary (in red) is much smoother after the perturbation.

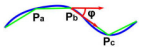

FIGURE 8. Turning angle at a Boundary Vertex

Boundary vertices P_a, P_b, and P_c are consecutive end points along a boundary line segments (in green) that discretely sample a smooth boundary curve (in blue). The turning angle φ at vertex P_b is the angle between left tangent line (P_aP_b) and right tangent line (P_bP_c, both lines in red).

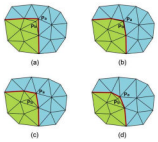

FIGURE 9. Adjustment of non-boundary Vertex Position

After perturbation of a boundary vertex, its neighboring non-boundary vertices' positions get adjusted to improve smoothing results in the next smoothing iteration. Two different regions are shown in blue color and green color respectively. Red line segments mark the regional boundary. (a) Before perturbation of vertex P_a. (b) After the perturbation, P_a is close to a non-boundary vertex P_b and cannot move further without flipping the orientation of any triangle. (c) The repulsion of P_b away from P_a makes room for further movement of P_a. (d) After another iteration of smoothing, P_a moves closer to its desirable location, and the boundary curve (in red) is much smoother than the curve in (b).

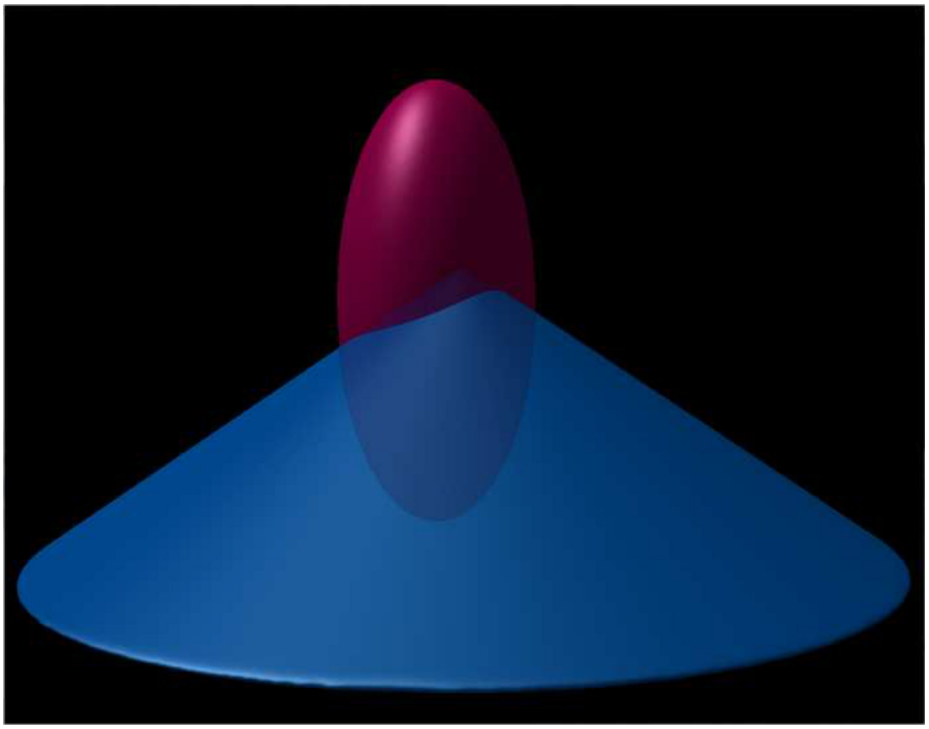

FIGURE 10. Simulated Dataset

An ellipsoid is generated for algorithm evaluation. The top half of the ellipsoid is outside the cone and is classified as region one. The bottom half of the ellipsoid is inside the cone and classified as region two. A boundary curve that defines different regions is thus delineated on the surface of the ellipsoid.

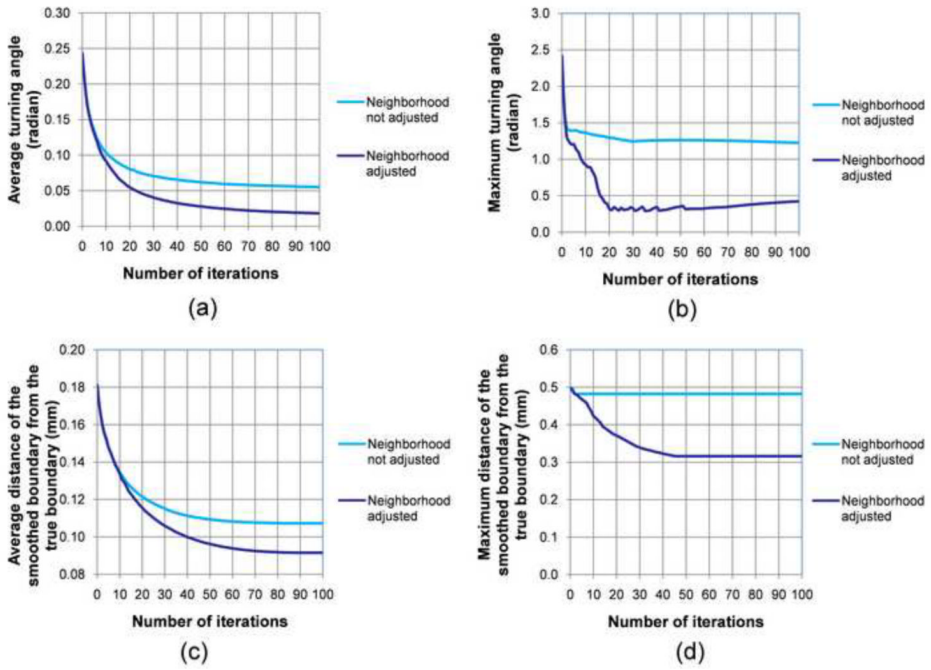

FIGURE 11. Parcellations on the Simulated Dataset at 1mm Resolution

Different measures are compared between approaches of smoothing with and without adjusting neighborhood vertices, as functions of smoothing iterations: (a) Average turning angles (in radian); (b) Maximum turning angle (in radian); (c) Average distance of the smoothed boundary from the true boundary (in mm); (d) Maximum distance of the smoothed boundary from the true boundary (in mm).

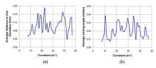

FIGURE 12. Curvature-Related Parcellation on the Simulated Dataset at 1mm Resolution

The results are generated using smoothing with neighborhood adjustment with 50 iterations. (a) Average distance from true boundary is displayed as a function of curvature; (b) Average turning angle is displayed as a function of curvature.

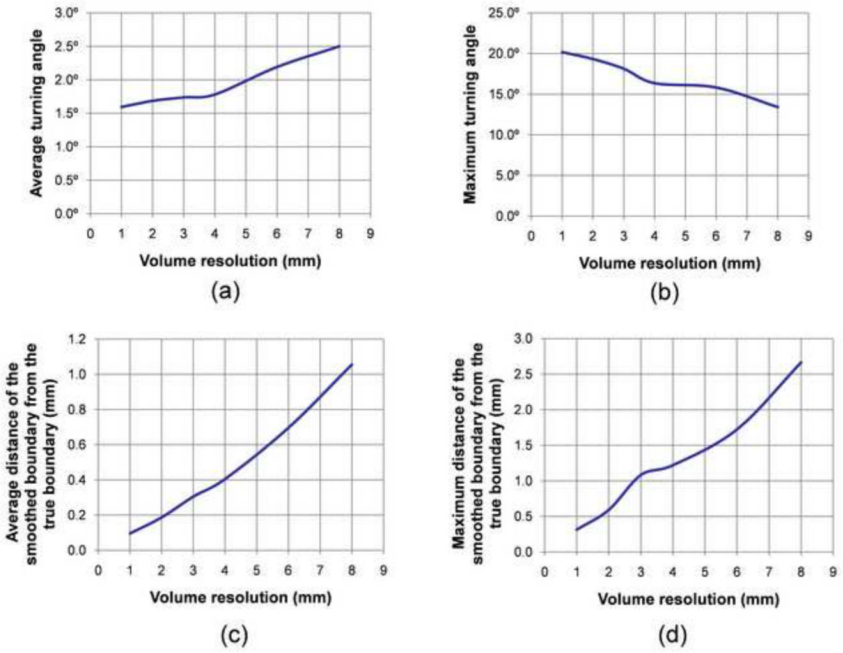

FIGURE 13. Comparing Parcellations on the Simulated Dataset at Varying Resolutions

Different measures are compared as functions of volume resolutions. The results are generated using smoothing with neighborhood adjustment with 50 iterations. (a) Average turning angles (in radian); (b) Maximum turning angle (in radian); (c) Average distance of the smoothed boundary from the true boundary (in mm); (d) Maximum distance of the smoothed boundary from the true boundary (in mm).

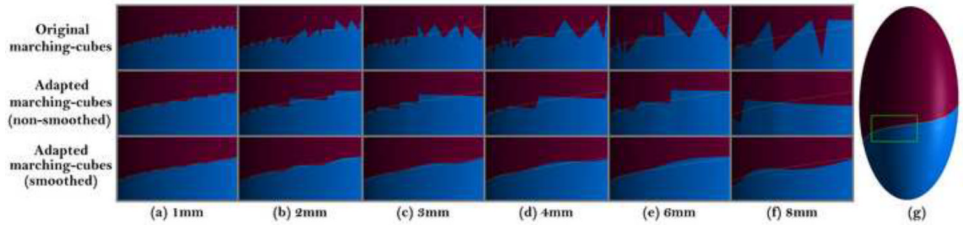

FIGURE 14. Comparing Regional Boundaries Generated by Different Methods at Varying Resolutions

Top row: boundary generated by original marching-cubes algorithm; middle row: boundary generated by adapted marching-cubes algorithm without smoothing; bottom row: boundary generated by adapted marching-cubes algorithm with smoothing. (a) 1mm resolution; (b) 2mm resolution; (c) 3mm resolution; (d) 4mm resolution; (e) 6mm resolution; (f) 8mm resolution; (g) Ellipsoid with selected region whose enlarged view shown in (a)–(f) marked.

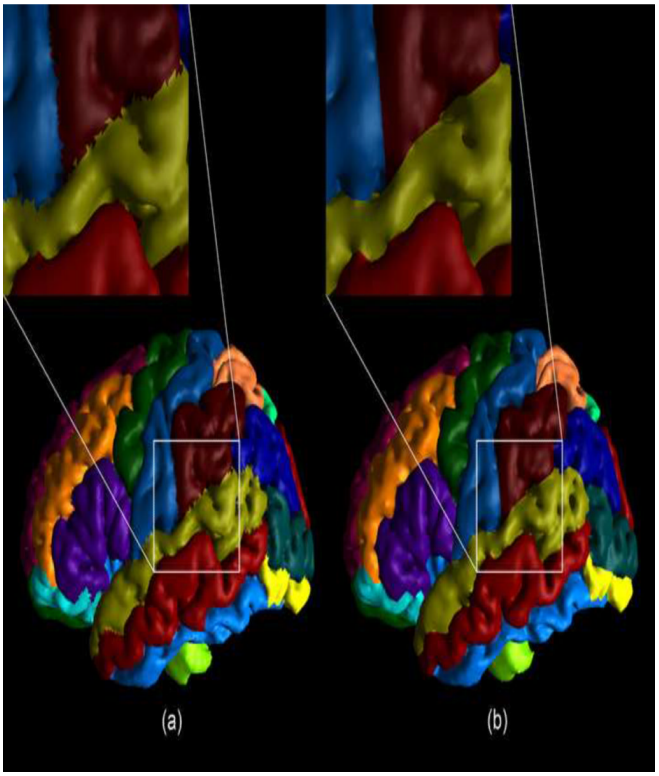

FIGURE 15. Comparing Parcellations on a Cortical Surface

The cortical surface is displayed as colored regions. (a) Surface generated by original marching-cubes algorithm with label assigned to each triangle from the voxel that contains the largest portion of the triangle; (b) Surface generated using our adaptive subdivision with boundary-preserved simplification and B-spline smoothing. Both surfaces have equal number of triangles. On the top left side of each image is an enlarged fragment of the view.

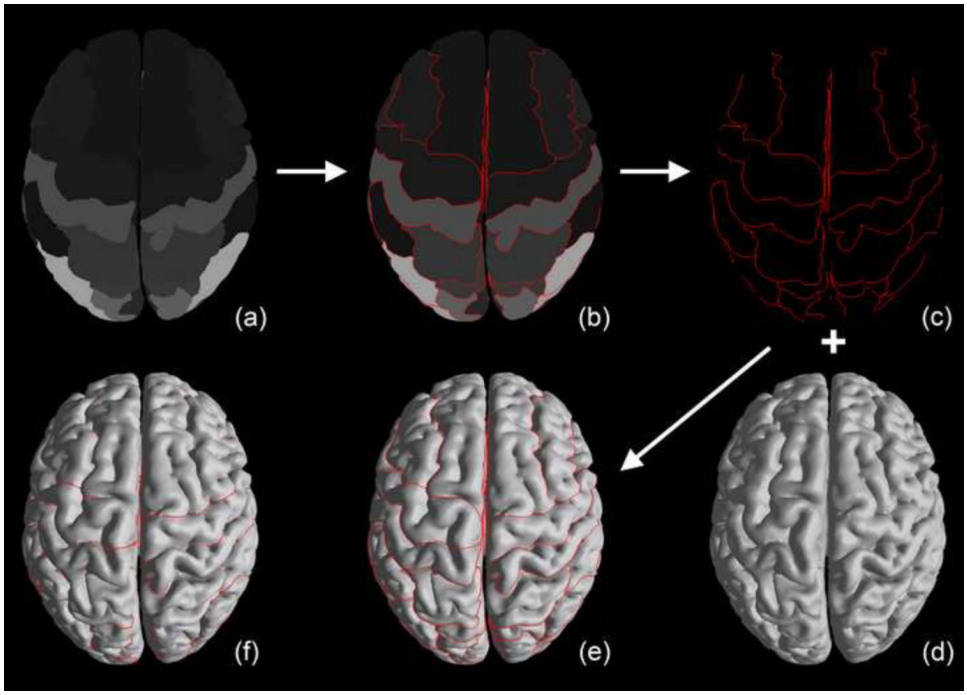

FIGURE 16. Delineation Display using Double-Snapshot Method

(a) the first snapshot generates an image from labeled markers of triangles without lighting; (b) region boundaries (in red) are traced from the above image by pixels that are on the boundary of regions with different intensity values; (c) region boundaries alone; (d) second snapshot of the lighted surface, at the same viewing angle as in (a); (e) region boundaries from (c) are super-imposed onto image from (d) to give the final visualization of delineation results; (f) result of traditional display, in which some of the boundary curves are buried inside sulci and invisible.

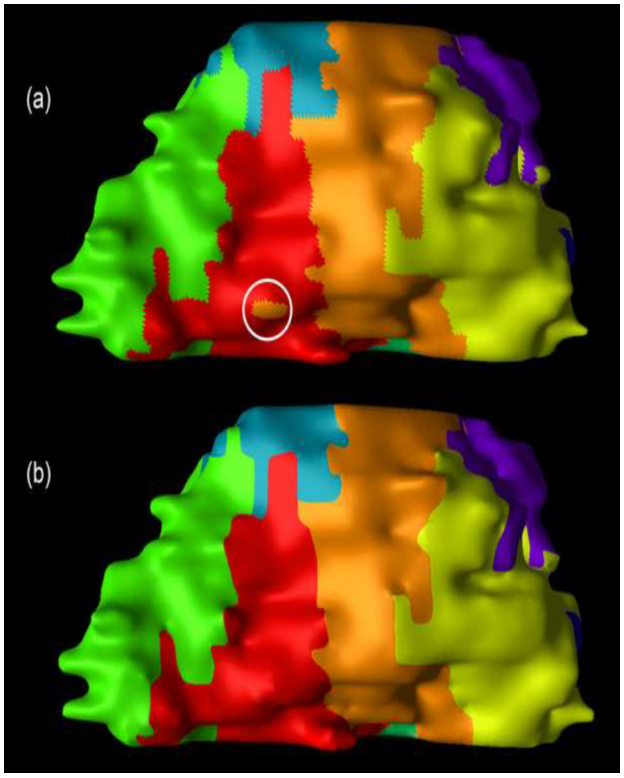

FIGURE 17. Comparing Parcellations of the Right Thalamus

The surface of right thalamus is displayed as colored regions. (a) Surface generated by original marching-cubes algorithm with label assigned to each triangle from the voxel that contains the largest portion of the triangle. White circle marks a misclassified region; (b) Surface generated using our adaptive subdivision with boundary-preserved simplification and B-spline smoothing. Both surfaces have equal number of triangles.

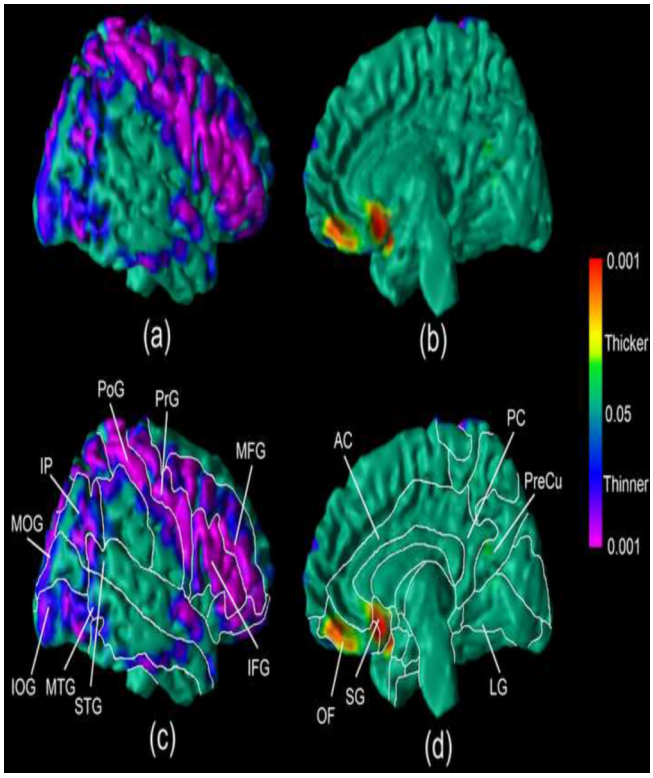

FIGURE 18. Maps of Group Differences in Cortical Thickness

Statistical significance of differences in cortical thickness of right hemisphere across groups (high vs. low risk in major depressive disorder) in participants from 2nd and 3rd generations are color coded, with warm colors (yellow, orange, and red) representing significantly thicker cortices in the high risk group and cooler colors (blue and purple) representing thinner cortices in that group. The color bar indicates the color-coding of p-values for testing of statistical significance at each point in the brain. Upper panel: results displayed without region boundaries. Lower panel: results shown with region boundaries. Lateral views: (a, c). Medial views: (b, d).

AC: anterior cingulate
IOG: inferior occipital gyrus
IP: inferior parietal lobule
LG: lingual gyrus
MFG: middle frontal gyrus
MOG: middle occipital gyrus
MTG: middle temporal gyrus
OF: orbitofrontal cortex
PC: posterior cingulate
PoG: post-central gyrus
PreCu: precuneus
PrG: pre-central gyrus
SG: subgenual cortex
STG: superior temporal gyrus

Table 1

Comparison of results on simulated dataset at various resolutions using our adapted marching-cubes algorithm with those obtained by original marching-cubes algorithm, in terms of average distance from true boundary, maximum distance from true boundary, average turning angle along the boundary curves, maximum turning angle along the boundary curves, and percentage of misclassified area.

Resolution	Measurement	Original marching-cubes	Adapted marching-cubes	
			Non-smoothed	Smoothed
1 mm	Average distance from true boundary (mm)	0.389	0.181	0.096
	Maximum distance from true boundary (mm)	0.994	0.499	0.316
	Average angle (radian)	1.616	0.243	0.028
	Maximum angle (radian)	2.995	2.417	0.352
	Percentage of misclassified area	0.206%	0.103%	0.056%
2mm	Average distance from true boundary (mm)	0.801	0.368	0.187
	Maximum distance from true boundary (mm)	1.957	0.997	0.594
	Average angle (radian)	1.618	0.321	0.029
	Maximum angle (radian)	3.005	2.629	0.338
	Percentage of misclassified area	0.388%	0.209%	0.099%
3 mm	Average distance from true boundary (mm)	1.184	0.567	0.304
	Maximum distance from true boundary (mm)	2.889	1.497	0.833
	Average angle (radian)	1.723	0.292	0.030
	Maximum angle (radian)	3.131	2.407	0.318
	Percentage of misclassified area	0.527%	0.318%	0.123%
4mm	Average distance from true boundary (mm)	1.449	0.696	0.405
	Maximum distance from true boundary (mm)	3.844	1.984	0.921
	Average angle (radian)	1.601	0.379	0.031
	Maximum angle (radian)	2.967	2.272	0.283
	Percentage of misclassified area	0.689%	0.363%	0.194%
6mm	Average distance from true boundary (mm)	2.291	1.065	0.696
	Maximum distance from true boundary (mm)	5.633	2.975	1.721
	Average angle (radian)	1.708	0.373	0.038
	Maximum angle (radian)	3.059	2.287	0.276
	Percentage of misclassified area	0.944%	0.564%	0.268%
8 mm	Average distance from true boundary (mm)	2.911	1.476	1.055
	Maximum distance from true boundary (mm)	7.211	3.916	2.664
	Average angle (radian)	1.638	0.306	0.044
	Maximum angle (radian)	3.079	1.603	0.237
	Percentage of misclassified area	1.08%	0.793%	0.387%

Table 2

Comparison of results on a cortical surface using our adapted marching-cubes algorithm with those obtained by original marching-cubes algorithm, in terms of triangle count, maximum distance, and average distance of boundary vertices to their nearest region boundary plane as defined in section 2.1.1, as well as the average turning angle along the boundary curves.

		Initial triangles	Triangles after simplification	Maximum distance (mm)	Average distance (mm)	Average angle(rad)
Original marching-cubes		611056	300852	0.900	0.252	1.636
Adapted marching-cubes	Non-smoothed	1440376	300852	0.000	0.000	0.908
	Smoothed	1440376	300852	0.493	0.130	0.271

Table 3

Comparison of results on a right thalamus surface using our adapted marching-cubes algorithm with those obtained by original marching-cubes algorithm, in terms of triangle count, maximum distance, and average distance of boundary vertices to their nearest region boundary plane as defined in section 2.1.1, as well as the average turning angle along the boundary curves.

		Triangles	Maximum distance (mm)	Average distance (mm)	Average angle (rad)
Original marching-cubes		74804	0.334	0.054	1.825
Adapted marching-cubes	Non-smoothed	74804	0.000	0.000	0.707
	Smoothed	74804	0.221	0.034	0.154