Surface Area Estimation in Practice

Guy Windreich[1], Nahum Kiryati[1], and Gabriele Lohmann[2]

[1] Dept. of Electrical Engineering–Systems, Tel Aviv University
Tel Aviv 69978, Israel
nk@eng.tau.ac.il
[2] Max-Planck Institute of Cognitive Neuroscience
Stephanstr. 1a, 04103 Leipzig, Germany
lohmann@cns.mpg.de

Abstract. Consider a complex, convoluted three dimensional object that has been digitized and is available as a set of voxels. We describe a fast, practical scheme for delineating a region of interest on the surface of the object and estimating its original area. The voxel representation is maintained and no triangulation is carried out. The methods presented rely on a theoretical result of Mullikin and Verbeek, and bridge the gap between their idealized setting and the harsh reality of 3D medical data. Performance evaluation results are provided, and operation on segmented white matter MR brain data is demonstrated.

Keywords: Surface area estimation, digital geometry, voxels objects, morphometric measurements, segmented white matter.

1 Introduction

Consider a three dimensional object that has been digitized and is given as a set of voxels. How can one delimit a region of interest on the surface of the object and estimate its area? Estimating the area of specific regions in the highly convoluted cortical surface is a challenging instance of this generic problem.

The cortex is the thin outermost layer of grey matter in the brain; cortical surface area is likely to be related to functional capacities [15]. The interesting problem of topologically-correct brain segmentation in MR images is beyond the scope of this paper. Given the segmented cortical voxel set, a useful surface area measurement process should include three non-trivial steps: Tracing the boundary of the region of interest on the surface, identifying the region surrounded by the boundary, and estimating its area.

Marking a boundary contour on a convoluted surface is not straightforward, because parts of the intended curve may not be visible. To overcome this limitation, the user should be able to select a sequence of visible key points, and have them connected automatically to form the boundary. This calls for an efficient algorithm for geodesic path generation, i.e., for finding shortest paths between points on a surface. As to region identification, in the continuous world Jordan's theorem ensures that a simple closed contour encloses a region and separates the interior and exterior. In the discrete domain, identifying a region of interest by

I. Nyström et al. (Eds.): DGCI 2003, LNCS 2886, pp. 358–367, 2003.

its outline is prone to paradoxes, since the discrete version of Jordan's theorem does not generally hold and different definitions of digital connectivity must be used for the region and its boundary [12]. Estimating the continuous area of a surface that is available only in digital form is fundamentally difficult. Different continuous surfaces, with different surface areas, may have the same digital representation. Furthermore, the voxel representation of smooth continuous surfaces is generally jagged, so the total area of exposed voxel faces is usually much greater than that of the original continuous surface.

Transforming the digital object and its surface from their original voxel representation to a triangle-based polyhedral representation has certain advantages. An efficient algorithm for geodesic path generation on triangulated domains is available [8], facilitating key point based boundary generation. Moreover, on the triangulated domain the paths found are continuous, so the boundary outlines the region of interest in a well defined way. Two difficulties however arise. First, when using the marching cubes algorithm [13] to create the triangulated representation, topological ambiguities may occur and holes may be generated [10]. Second, the surface area estimate produced by summing up the area of the resulting triangles does not converge to the true surface area as the resolution increases [7]; this follows from the *locality* of the marching cubes algorithm. Klette and Sun [11] suggest that surface area estimators that converge to the true surface area can be obtained by using a *global* polyhedrization method. More efficient algorithms are required in order to make their method practical for large, high resolution data sets; see also [2].

This research follows a different approach: the original voxel representation of the object is maintained and no polyhedrization is carried out. This simplifies the overall system design and avoids difficulties, ambiguities and distortion that may arise due to the application of a polyhedrization process to a complex, convoluted surface. A voxel-based surface area estimator was presented by Mullikin and Verbeek [14]. Extending the planar perimeter estimation methodology [3,5], their estimator is designed to be unbiased and minimize the mean square error (MSE) for planes, and its operation is evaluated with spheres. The estimator of Mullikin and Verbeek [14] is at the core of the method presented here. However, as will be discussed, it cannot be directly applied to complex convoluted surfaces and various difficulties need to be addressed. The creation of a complete voxel-based surface area estimation method, applicable to surfaces as complex as that of the brain, is the focus of this research. Preliminary results were presented in [16].

2 Delimiting the Region of Interest

2.1 Border vs. Boundary

Given a 3D object that is represented as a set of voxels, one can easily identify the voxels that are 6-connected to the background and view them as the border of the object. The border set can be represented as a graph; once the user defines keypoints on the border, they can be automatically connected using the algorithm of [9] to obtain a closed contour that encloses the region of interest.

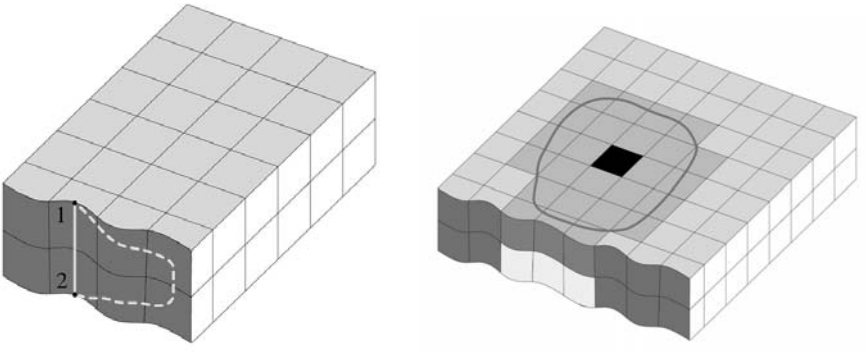

Fig. 1. *Left:* The shortest path generated (solid) between border voxels '1' and '2' does not follow the intended contour (dashed), even though the path is constrained to border voxels. *Right:* A region grown from the seed voxel (black) within the contour will include not only the intended region of interest (darker voxels in the top layer) but also other voxels in the border set to which they are connected (bottom layer).

Delimiting the region using the voxel-chain contour is inadequate. Consider for example the object detail shown in Fig. 1 (left). The voxels marked '1' and '2' both belong to the border set. When connecting them (as part of the contour generation process) using an algorithm like [9], the connection (solid) will not follow its intended path (dashed). Once a closed contour within the border set has been created, Fig. 1 (right) demonstrates that the contour does not properly enclose the region of interest. A region grown from the seed voxel (black) within the outline will include not only the intended region of interest (darker voxels in the top layer) but also other voxels in the border set to which they are connected (bottom layer).

The difficulties associated with using the border voxel set can be eliminated by keeping track of the *boundary* of the object, the set of voxel *faces* that separates the object from the background [1]. This is easily verified for the pathological cases shown in Fig. 1.

Marking the region of interest requires the following steps. First, detection of the border-set and the boundary. Second, selection of key-faces on the boundary and creation of a chain of faces that connects the key-faces on the surface. Third, seed selection within the region of interest, growing the region of interest on the boundary and associating it with the border-set. Maintaining the border-set representation is crucial, since the surface area estimation algorithm, based on [14], operates on the border-set.

2.2 Algorithms

Border and Boundary Detection. The straightforward approach to simultaneous detection of the border-set and the boundary is to visit each voxel in the 3-D image and determine whether it is 6-connected to a background voxel. Each surface voxel detected is inserted to a border hash table and each boundary

face to a boundary hash table. In practice, for the brain images used in this research, this operation took only 7 seconds on a 350MHz PC. Thus, sophisticated alternative algorithms were not needed.

Constructing the Boundary Adjacency Graph. Following the detection of the border and the boundary, the boundary faces adjacency graph is constructed. For each boundary face, adjacent boundary faces are determined. A simple closed surface can be represented as a directed graph with indegree and outdegree two [1]. Here, we extract a region of interest on the closed surface of an object, i.e., a surface patch. The directed graph representation is not suitable for the surface patch: there will be nodes in the graph with indegree (or outdegree) smaller than two. For example, consider an object that consists of a single voxel. Suppose the region of interest contains five faces out of the six that form the surface of the voxel. Only one of these five faces has four neighbors (two of them are adjacent faces in the sense of [1], that share the outgoing edges with the source face. The other two share the incoming edges). Each of the other four faces has only *three* edges shared with other faces. Clearly, this simple region of interest cannot be described using the directed graph representation. Thus, to find the boundary faces adjacent to each face, we modify the method of [1] to allow four adjacent faces for each boundary face, one for each of its edges.

Connecting Key-Points on the Boundary. Kiryati and Székely [9] described an efficient algorithm for finding shortest paths on voxel surfaces represented as graphs. Here, given the boundary graph representation, a similar algorithm can be devised to find reasonably short paths between the keypoints defining the region of interest on the surface. As in [9], the sparsity of the boundary adjacency graph allows very efficient search. Unlike [9], where different spatial adjacency relations (link types) between voxels induce different weights for arcs in the surface graph, here all four arcs connecting a boundary face to adjacent faces are equally weighted. This algorithm requires $O(N \log N)$ time, where N is the number of boundary faces.

Growing the Region of Interest on the Object Boundary. In the boundary adjacency graph, the degree of each node is between one and four. Like [1], we use a breadth first search algorithm for graph traversal that begins at an arbitrary node. Unlike [1], we have the list of boundary faces in memory so we do not have to detect the boundary, but only to mark the nodes (boundary faces) that are within the region of interest. To allow the search to stop at the borderline of the region of interest, all the nodes representing the outline of the region of interest are marked as they are generated by the shortest-path algorithm. Thus, they are already marked when surface growing starts.

3 Surface Area Estimation

3.1 The Estimator of Mullikin & Verbeek [14]

Mullikin and Verbeek [14] extended the theory of 2-D perimeter estimation to 3-D surface area estimation. Their algorithm begins by detecting all surface

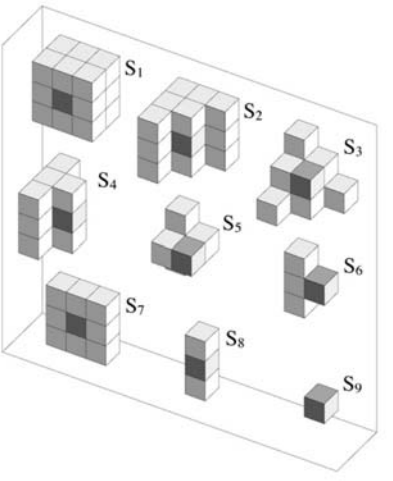

Fig. 2. The nine unique surface voxel classes (after [14]). A voxel under consideration (dark) is classified according to the arrangement of adjacent surface voxels (lighter grey and white). Only voxels of types S_{1-3} appear in a planar surface.

voxels, i.e., object voxels that are 6-connected to background voxels. Each voxel is classified into one of nine possible classes, and the surface area is estimated as a linear combination of the class membership values $\{N_i\}$:

$$\hat{S} = \sum_{i=1}^{9} W_i N_i$$

Each surface voxel is classified according to the number and configuration of its faces that are exposed to the background. Up to rotation and mirroring, there are exactly nine unique voxel classes (Fig. 2), denoted S_{1-9}. Only voxels of types S_{1-3} appear in digital planes. Voxel types S_{4-6} are found in curved border regions. Voxel types S_{7-9} exist in extreme situations, where the object is a plane, line or point respectively.

Having defined the voxel classification scheme, Mullikin and Verbeek determined the weights W_{1-3} associated with voxels in classes S_{1-3}, to make the surface area estimate *unbiased* for random plane orientations and to minimize the mean square error. These weights are $W_1 \approx 0.894$, $W_2 \approx 1.3409$ and $W_3 \approx 1.5879$; the coefficient of variation ($CV = \sigma/\mu$) for planes is 2.33%. Clearly, an unbiased estimator for planes will have very small errors when operating on curved surfaces, where local estimation errors, obtained at differently oriented patches, essentially cancel out.

This methodology does not determine the weights W_{4-9} associated with classes S_{4-9}. Following the spatial grid method [4], Mullikin and Verbeek set $W_4 = 2$, $W_5 = 8/3$ and $W_6 = 10/3$. No weights were assigned by Mullikin and Verbeek to voxel types S_{7-9}. Experimental performance evaluation with spheres

revealed some bias related to the radius, that can be alleviated by averaging the surface area of the object with that of the background.

The surface area estimator of [14] is local. While it does not exhibit multigrid convergence, it operates directly on voxels, is easy to implement, very fast to compute and achieves very reasonable accuracy. Note that multigrid convergence is related to surface area estimation accuracy with resolution approaching infinity (and surface curvature approaching zero). This is not the case in present MR brain images, where the cortical surface curvature is high. For an alternative voxel-based surface area estimation method, see e.g. [6].

3.2 Voxel Types in Brain Surfaces

Table 1 shows the frequency of the nine surface voxel types in a $160 \times 200 \times 160$ segmented white matter MR brain image (in the grey-white matter interface). It is seen that all nine voxel classes are represented, and that voxel types S_{4-9} constitute 3.23% of the 187,567 surface voxels. About 1.35% of the surface voxels are of types S_{7-9}.

Table 1. The frequencies of surface voxels types in the grey-white matter interface of a segmented $160 \times 200 \times 160$ MR brain image.

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	total
No. of voxels	65878	46532	37218	1151	2547	1383	468	923	725	187567
Frequency (%)	42.1	29.6	23.7	0.73	1.62	0.88	0.30	0.59	0.46	100

The presence of voxels types S_{4-9} in brain data necessitates the assignment of weights to these classes, but these are not provided by the design methodology, that is based on digital planes. However, the fairly low frequency of these voxels means that the overall surface estimation accuracy is not too sensitive to the weights selected. As discussed above, weights for classes S_{4-6} were already proposed in [14]. For voxel classes S_{7-9}, we suggest the following.

For S_7, with two opposite faces exposed to the background, we take the weight to be twice the weight of voxel type S_1 (that has only one face exposed to the background), i.e., $W_7 = 1.79$. A voxel in S_8 has two pairs of adjacent faces exposed to the background; we can take its weight as twice that of voxel type S_2 (that has only one pair of adjacent faces exposed to the background), i.e., 2.68. Alternatively, one can argue that the weight should be 4/5 of the weight of S_6 (that has 5 faces exposed to the background). This gives an almost identical weight of 2.67. Thus, for all practical purposes we can take $W_8 = 2.68$. As to S_9, with all 6 faces exposed to the background, the weight can be taken to be the sum of the weights of S_6 and S_1 (4.23), or twice that of S_4 (4) or the sum of the weights of S_5 and S_2 (4.01). The difference between these values is insignificant considering the low frequency of these voxels. We take the average, $W_9 = 4.08$.

Consider a flat, thin object consisting of a single layer of S_7 voxels. Suppose that the boundary region of interest is on one side of the object. Each of the S_7

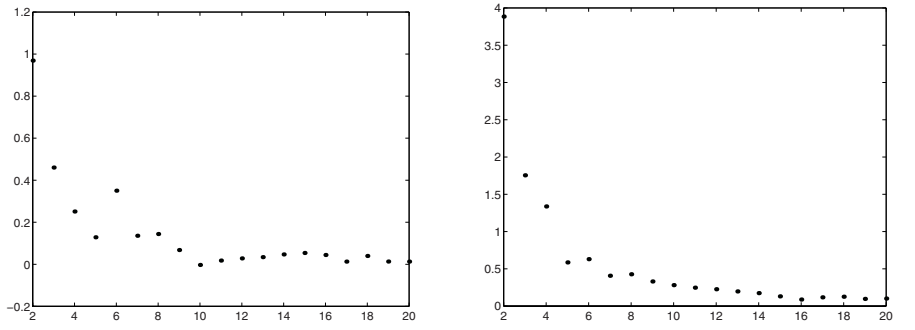

Fig. 3. *Left:* The relative mean estimation error (percent) in estimating the surface area of spheres, as a function of sphere radius (with object-background averaging). *Right:* The corresponding coefficient of variation (percent).

voxels has two faces exposed to the background, but only one of them belongs to the region of interest! In this case only half of the weight W_7 should contribute to the surface area estimate. Generally, for a voxel with P faces exposed to the background, of which p faces belong to the region of interest on the boundary, we take

$$W_i^p = \frac{p}{P} \cdot W_i$$

where W_i is the voxel class weight. In most cases $p = P$, so $W_i^p = W_i$.

3.3 Performance Evaluation

Mullikin and Verbeek [14] evaluated the performance of their surface area estimator using simulated spheres and cylinders. Here we report on our simulation results, with synthetic spheres and ellipsoids. Small spheres represent objects with high surface curvature, that deviate mostly from the planar surface model used in the design of the estimator. Large spheres, with their uniformly distributed surface normals, can demonstrate the unbiasedness of the estimator. Unlike spheres, surface normal directions on ellipsoids are not uniformly distributed. Testing the surface area estimator on ellipsoids, suggested in [7], indicates the sensitivity of the estimation error to nonuniformity of the normal directions distribution. Note that planar objects, having a single normal direction (or two opposite directions if both sides are considered), are generally the worst case for the estimator: the coefficient of variation (the standard deviation divided by their mean) is 2.33% for randomly oriented planes.

Fig. 3 (left) shows the relative mean estimation error (percent) in estimating the surface area of spheres (average of object and background surface areas), as a function of sphere radius. Each point in the graph is based on 50 spheres, whose center points are uniformly distributed within the unit voxel. It is seen that the relative mean estimation error is less than 1% even for spheres of radius 2, that it rapidly decreases as the radius is increased, and is practically zero for radii larger than 10. The coefficient of variation of these measurements is presented

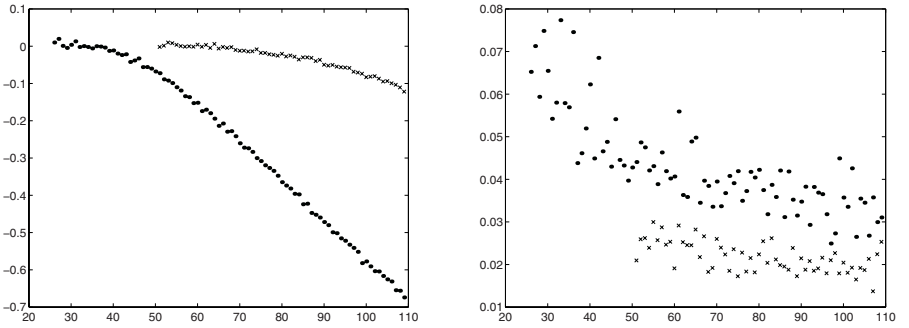

Fig. 4. *Left:* Relative mean estimation error (percent) as a function of ellipsoid main semi-axis a (using object-background surface area averaging). The dots refer to the ellipsoid family $(a, 26, 25)$; the x's to $(a, 51, 50)$. *Right:* The corresponding coefficients of variation (percent).

in Fig. 3 (right). It rapidly decreases with the sphere radius, from about 4% with sphere radius 2, through 0.5% with radius 5 to negligibly small values at larger radii. These results are similar to those obtained with spheres in [14]; they demonstrate the outstanding accuracy of the estimator even when the surface curvature is very high. Note that surface curvature radii between 2 and 5 are common in the segmented MR white matter brain data used in this research.

Consider an ellipsoid with semi-main axes a, b and c parallel to the coordinate system axes. Each data point in Fig. 4 corresponds to surface area estimation of 50 such ellipsoids, with center points uniformly distributed within the unit voxel. The dots in Fig. 4 (left) refer to the ellipsoid family $(a, b = 26, c = 25)$, and shows the relative mean surface area estimation as a function of a. The x signs refer to the ellipsoid family $(a, b = 51, c = 50)$. As expected, the error is almost zero for nearly spherical ellipses; it slowly grows as a increases and the ellipses become elongated. The respective coefficients of variation are shown in Fig. 4 (right); they are very small.

4 Application to Brain Data

The methods presented in this paper have been implemented as a C++ program named *Surf3D*, for Unix platforms. *Surf3D* receives as input 3D binary images, visualizes the data, and allows the user to interact with the surface using the mouse. A graphical user interface allows easy object rotation, synthetic illumination, etc. Using the mouse, the user marks a set of key points that indicate the region of interest on the surface of the viewed object. The keypoints are connected automatically to create the surrounding contour. From a seed point selected by the user, *Surf3D* grows the region of interest and estimates its area.

A typical work session with *Surf3D* is illustrated in Fig. 5. The input data was a $160 \times 200 \times 160$ segmented white matter MR brain image, with 558,363 cubic object voxels, of which 156,825 were surface voxels, having 300,130 boundary faces (larger data sets are readily accommodated). The session begins by loading

an image containing segmented white matter MR brain data. The border and the boundary of the object are detected and displayed. The user defines the contour of the region of interest by selecting keypoints on the surface. When ready, the program is instructed to close the contour. The user then clicks on a surface point within the region of interest, from which the program grows the region and estimates its surface area. With a 350MHz PC running Linux, the computing time spent in each of the steps is a few seconds or less.

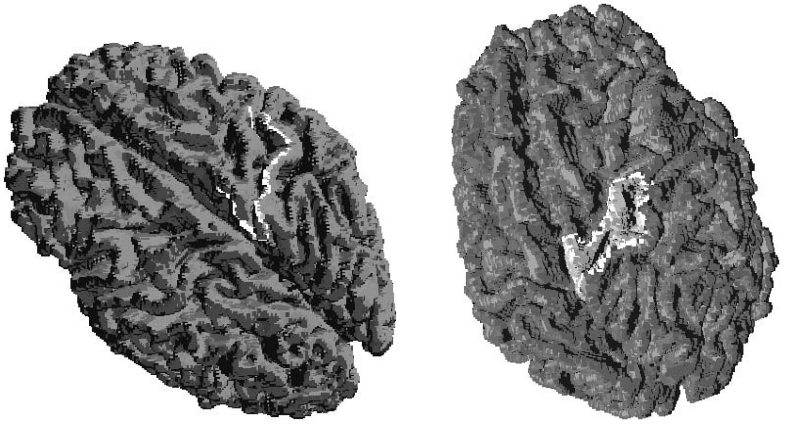

Fig. 5. Interactive definition of the region of interest on the surface. *Left:* Automatic connection of key points provided by the user. *Right:* Closure of the contour surrounding the region of interest.

5 Conclusions

This research provides a fast, accurate and convenient scheme for estimating the surface area of regions of interest on the surface of digital objects. The input is a 3D binary digital image, i.e., a set of voxels. The voxel representation is maintained and no triangulation is carried out. The suggested technique bridges the gap between the theoretical results of Mullikin and Verbeek [14] and the reality of complex medical data. In particular, the method is well suited for highly convoluted surfaces. The accuracy is verified using synthetic surfaces: simulation results reported in [14] are corroborated, and augmented by results on ellipsoids. Operation is demonstrated on segmented white-matter MR brain data.

Acknowledgments

This research was supported by a grant from the G.I.F., the German-Israeli Foundation for Scientific Research and Development.

References

1. E. Artzy, G. Frieder and G.T. Herman, "The Theory, Design, Implementation and Evaluation of a Three-Dimensional Surface Detection Algorithm", *Computer Graphics and Image Processing*, Vol. 1, pp. 1-24, 1981.
2. D. Couerjolly, F. Flin and O. Teytaud, "Digital Surface Area Estimation", abstract, in Dagstuhl Seminar Report no. 339, Schloss Dagstuhl, Germany, 2002.
3. L. Dorst and A.W.M. Smeulders, "Length Estimators for Digital Contours", *Computer Vision Graphics Image Processing*, Vol. 40, pp. 311-333, 1987.
4. U. Hahn and K. Sandau, "Precision of Surface Area Estimation Using Spatial Grids". *Acta Stereologica*, Vol. 8, pp. 425-430, 1989.
5. J. Koplowitz and A.M. Bruckstein, "Design of Perimeter Estimators for Digitized Planar Shapes", *IEEE Trans. Pattern Analysis Machine Intelligence*, Vol. 11, pp. 611-622, 1989.
6. J. Lindblad and I. Nyström, "Surface Area Estimation of Digitized 3D Objects using Local Computations", Proc. DGCI'2002, *Lecture Notes in Computer Science*, Vol. 2301, pp. 267-278, 2002.
7. Y. Kenmochi and R. Klette, "Surface Area Estimation for Digital Regular Solids", Technical Report CITR-TR-62, Computer Science Department, University of Auckland, New Zealand, 2000. available online.
8. R. Kimmel and J. Sethian, "Computing Geodesics on Manifolds", *Proc. National Academy of Sciences*, Vol. 95, pp. 8431-8435, 1998.
9. N. Kiryati and G. Székely, "Estimating Shortest Paths and Minimal Distances on Digitized Three-Dimensional Surfaces", *Pattern Recognition*, Vol. 26, pp. 1623-1637, 1993.
10. R. Klette, F. Wu and S. Zhou, "Multigrid Convergence of Surface Approximations", Technical Report CITR-TR-25, Computer Science Department, University of Auckland, New Zealand, 1998. Available online.
11. R. Klette and H.J. Sun, "A Global Surface Area Estimation Algorithm for Digital Regular Solids", Technical Report CITR-TR-69, Computer Science Department, University of Auckland, New Zealand, 2000. Available online.
12. G. Lohmann, *Volumetric Image Analysis*, Wiley, Chichester, UK & Teubner, Stuttgart, Germany, 1998.
13. W.E. Lorensen and H.E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm", *ACM Computer Graphics*, Vol. 21, pp. 163-169, 1987.
14. J.C. Mullikin and P.W. Verbeek, "Surface Area Estimation of Digitized Planes", *Bioimaging*, Vol. 1, pp. 6-16, 1993.
15. X. Zeng, L.H. Staib, R.T. Schultz and J.S. Duncan, "Segmentation and Measurement of the Cortex from 3-D MR Images Using Coupled-Surfaces Propagation", *IEEE Transactions on Medical Imaging*, Vol. 18, pp. 927-937, 1999.
16. G. Windreich and N. Kiryati, "Voxel-based Surface Area Estimation", abstract, in Dagstuhl Seminar Report no. 339, Schloss Dagstuhl, Germany, 2002.