3-D object visualization, surface rendering, volume rendering, contour rendering

Piotr LEWANDOWSKI[*], Arkadiusz TOMCZYK[*],
Piotr S. SZCZEPANIAK[*,**]

VISUALIZATION OF 3-D OBJECTS IN MEDICINE – SELECTED TECHNICAL ASPECTS FOR PHYSICIANS

In the paper, selected visualization methods are described such as: surface rendering, volume rendering and the simplest approach basing on texture mapping. This work, however, does not aim at a detailed description of those methods but at a friendly presentation of possible ways of visualization and their features which can be of importance for a physician who has some expectations or needs, such as gaining insight into a 3-D object. Those expectations can be satisfied for example by making incisions or using surface transparency option which significantly improves visualization effect and can be of use in diagnostic process.

1. INTRODUCTION

The development of 3-D image segmentation methods entailed the need to represent 3-D images on flat 2-D displays in such a way as to achieve a photorealistic effect. This, in turn, resulted in the emergence of 3-D visualization methods. The object is shown in such a way as to give the user an idea about its structure. Until recently, interactive visualization of 3-D objects was difficult to perform due to very high cost of specialized equipment. However, now it has become possible, even using the cheapest home computers. The method in question is widely applied in medicine, particularly in diagnosing, surgery planning, therapy monitoring (e.g. radiotherapy). It is also very helpful in acquiring knowledge about anatomical structures that enables insight into an examined object. It can be also applied in the examination of bone and soft tissue morphology and detection of pathologies in organ structures.

In a sense, visualization is the opposite of image analysis. While the aim of image analysis is to find a shortened description of object's features, in visualization the shortened description of an object is a starting point for creating a 3-D image [1].

There are two ways to render an object in 3-D, namely surface rendering (when the image of the surface of an object is generated) and volume rendering (when the image of the whole object together with its 3-D inside is generated). Surface rendering requires a description of the edge of an object and its properties (e.g. by previous segmentation of an object) and only the edge is rendered here. On the other hand, rendering of the whole object, shows all the voxels that make up the object and assign each of them a colour and a transparency value.

[*] Institute of Computer Science, Technical University of Lodz, Wolczanska 215, 90-924 Łodz, Poland
[**] Systems Research Institute, Polish Academy of Sciences,Newelska 6, 01-447 Warsaw, Poland

3-D visualization encompasses many various topics, e.g. from the domain of computer graphics, such as lighting and shading (providing photorealism), projection of a 3-D object onto a view plane, etc. The techniques used include displacement mapping, texture mapping, and texture filtering, to name but a few.

The present paper does not offer a detailed description of 3-D visualization methods. It only outlines the basic rules of 3-D visualization and associated problems. In other words, the main purpose here is not to analyze 3-D visualization methods in great detail, but to explain them in a brief physician-oriented way. More detailed information can be found in [2,3,4,5,6,7,8,9,10].

Contemporary methods of visualization used in medicine, such as computed tomography or magnetic resonance, provide satisfactory visual quality. These are images with high display resolution and contrast that enable a precise construction of a 3-D solid figure that is an equivalent of a real object. As mentioned above, there are two basic ways to render an object: surface rendering and volume rendering. They are described in sections 2 and 3, respectively. Another approach basing on texture mapping and presenting 2-D images on parallel planes is presented in section 4. Finally, section 5 focuses on the problem of 3-D contour rendering which can be very helpful in diagnostic systems using 3-D image visualization. The paper concludes with the summary of the presented approaches.

2. SURFACE RENDERING

In surface visualization methods it is very important to find the edge of an object first. This is done by means of image segmentation, e.g. edge detection method [3]. For example, one can use a 2-D method that divides a 3-D image into slices and detects the edge of each of them separately, or a method which directly detects a 3-D object’s edges (e.g. the *marching cubes algorithm* described in [11]). The former method results in a set of contours built of layers, rendered in the same way as 2-D contours. The latter produces a triangle mesh, which is then shaded and sometimes also textured.

Note that sometimes the roles of visualization and segmentation can be shifted – then segmentation is not just an aiding mechanism, on the contrary – 3-D visualization methods help to evaluate the results of segmentation. As a result, one can avoid preliminary segmentation or use simple low level segmentation methods, which, with the help of the user, may suffice to construct a 3-D image

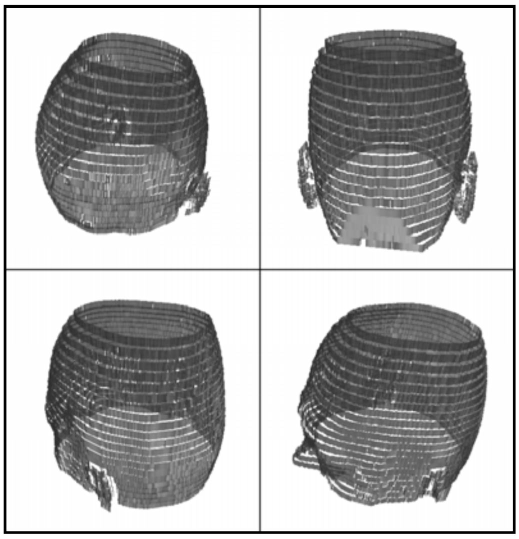

Fig.1. Surface rendering – different images of human skull

In order to render the edge of an object, a modified method of brightness thresholding was used. The modification consists in the fact that from the voxels exceeding the established threshold only those located nearest to the edge are chosen. This method is threshold-sensitive (especially in the presence of noise – see Fig.2), which means that the user has to define precisely the threshold, in order to achieve satisfactory results. However, this method enables quick and easy finding of the object's edge.

The thresholding stage is followed by the proper visualization – since the edge is detected on separate flat displays, it is rendered in the same way as 3-D contour.

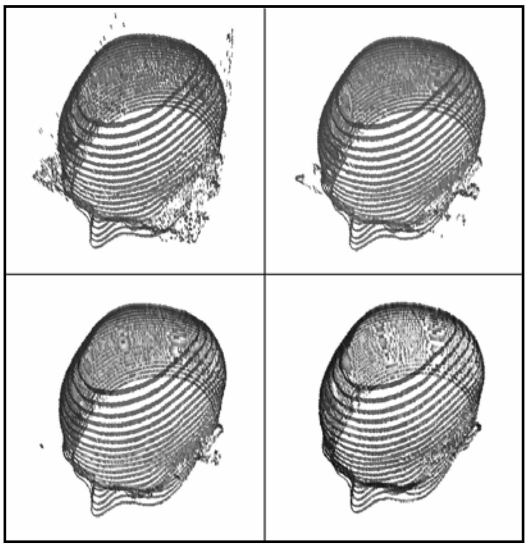

Fig.2. Sensitivity of the method to the threshold value. The noise in the image creates unwanted objects when the threshold is not set properly. In the presented, successive images the threshold value was increased until proper visualization quality was reached.

The visualization method implemented, based on surface rendering, was enriched with the possibility to look inside the object through incisions previously made in the image. Owing to this modification, the method gained a holistic character – rendering the object's contents.

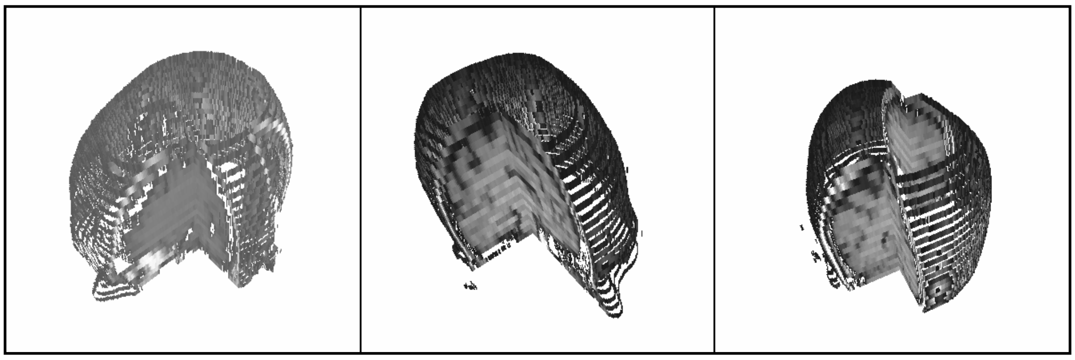

Fig.3. Incisions in the object's surface.

3. VOLUME RENDERING

Every voxel is assigned a colour (or a shade of grey) and a transparency value and is represented as a cube (or, more generally, a cuboid, depending on vertical scaling of the image). Transparency value and colour are chosen according to the user's needs. Transparency of the voxels can be constant for the whole image or dependent on the brightness level of each voxel. Material are defined and divided into groups according to their properties and voxels are classified into material groups according to their colours [2]. However, the image has to be previously segmented.

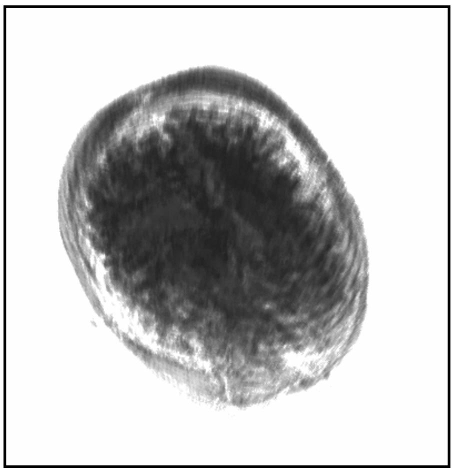

Fig.4. V*olume rendering*. Transparency mechanism – brain surface visible under the skull.

Transparency is a useful tool for showing and hiding selected areas of 3-D image. The voxels are assigned transparency value 1, which renders them as invisible and gives the viewer insight into deeper layers of the object (see Fig. .4).

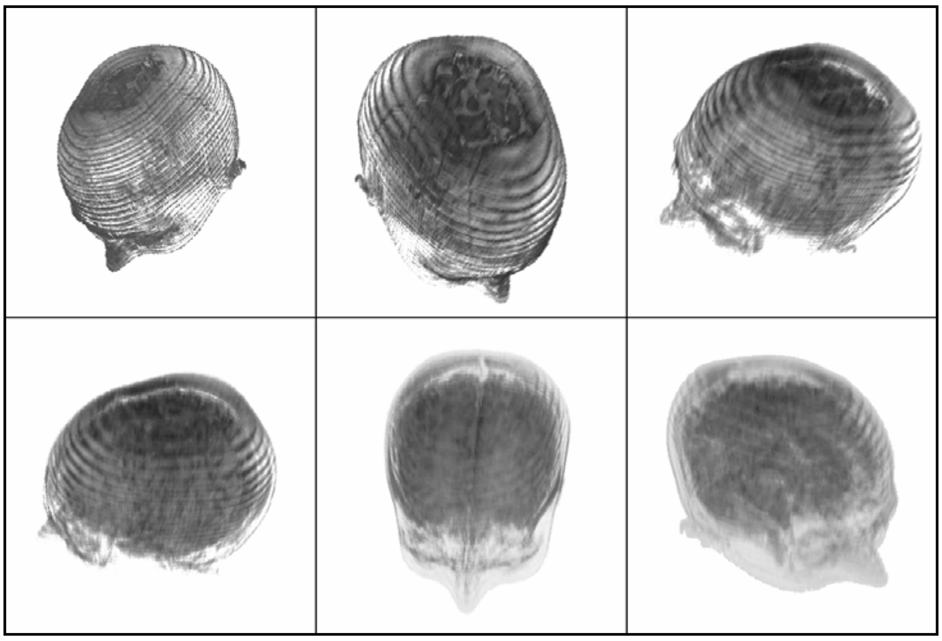

Fig.5. V*olume rendering* of a human skull. Successive images take higher and higher transparency values.

It is worth mentioning that rendering of the whole 3-D image is time-consuming and occupies much storage space, because it requires storing all the voxels together with their properties (colour, transparency) – it can occupy more than 1 gigabyte, depending on the resolution. For the purpose of limiting the number of voxels in a 3-D scene, the visualization proper was preceded by brightness thresholding and all the voxels that did not exceed the set threshold value were rejected. This helped to get rid of the background voxels at the very start and minimize storage occupancy. In this sense, rendering of the surface exclusively is a far better method – representing only the surface occupies little storage. Moreover, hardware used in 3-D graphics is designed to process triangle meshes, which provides greater efficiency of *surface rendering*.

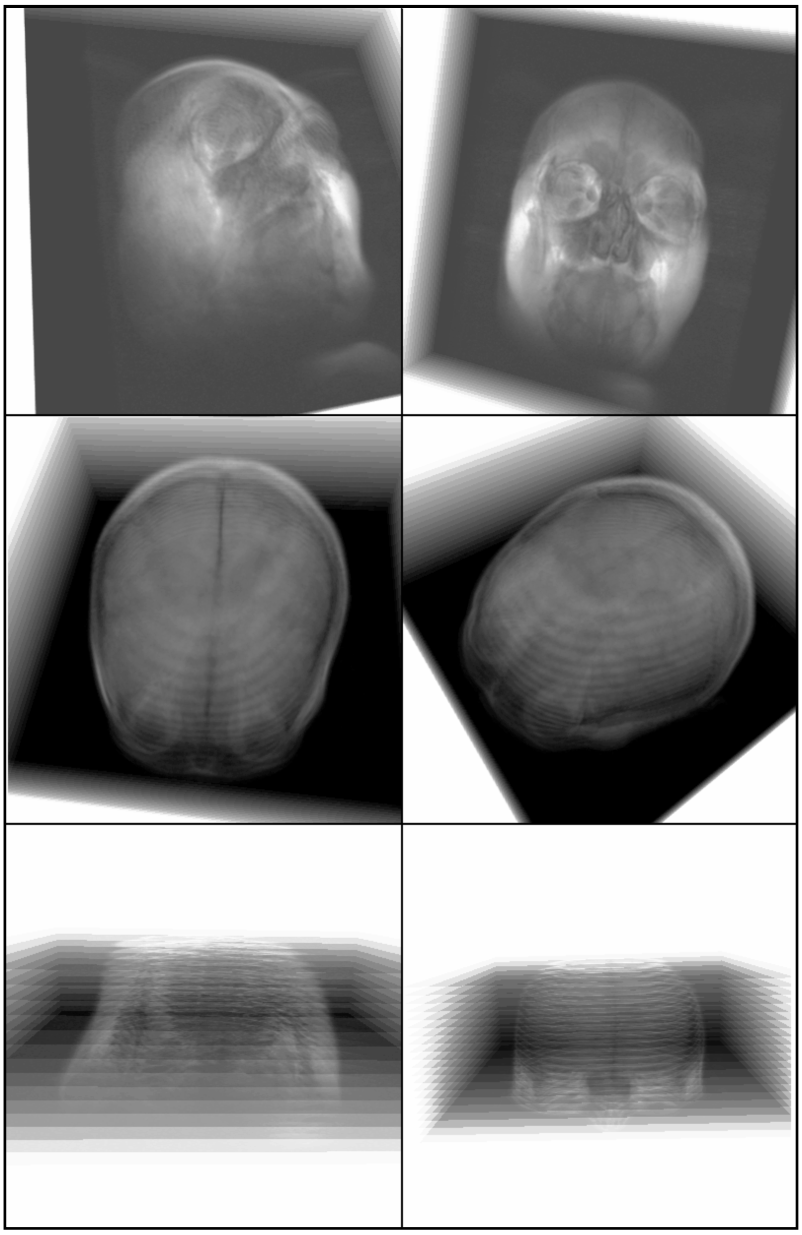

Fig.6. Texture mapping where simply partly transparent images are rendered on parallel planes. The last image shows the lacking side surface when looking at skull from perpendicular perspective which can be a disadvantage of that approach.

4. TEXTURE MAPPING

In another method, every of the flat images can be shown on parallel planes, one above another. In other words, a texture map is applied to each surface. Although it is the simplest possible visualization technique, with the transparency tool applied, it enables the object to be viewed quite well (see Figure .6). The advantage of this method is very high efficiency – it is very quick, occupies little storage (each 2-D image has only 4 vertices) and does not require preliminary edge detection. The disadvantage is the lack of side surface, which makes the object practically invisible, if looked at from perpendicular perspective.

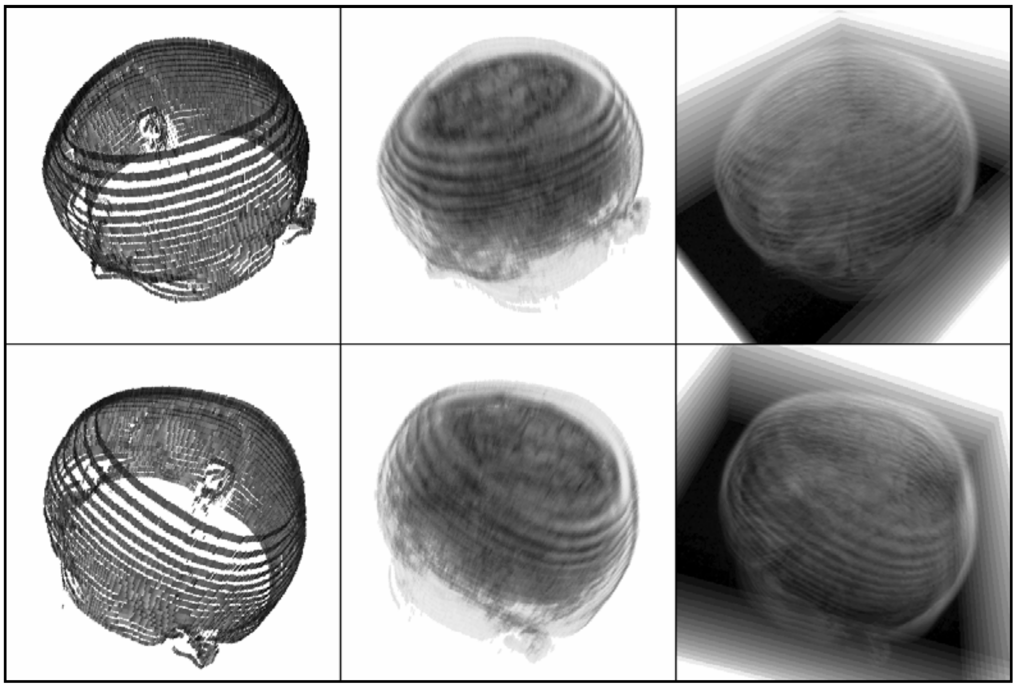

Fig.7. Comparison of the three methods presented in the present paper: surface rendering, volume rendering and a set of flat textures.

5. RENDERING OF CONTOURS

3-D contours can be rendered by the same method as 3-D images. The only difference is that in the case of contour visualization only surface rendering is applied. V*olume rendering* cannot be applied here as contours do not have an inside. In the case of a pile of 2-D contours, visualization is not difficult. Each of the contours is rendered separately as a continuum of polygons, in this case rectangles, every of which corresponds to one space in interpixel chain (in the case of Brownian Strings contour [12]). The fact that each of the 2-D contours is treated separately results in spaces (“holes”) in the object, because neighbouring contours are not connected with each other (see Fig.8).

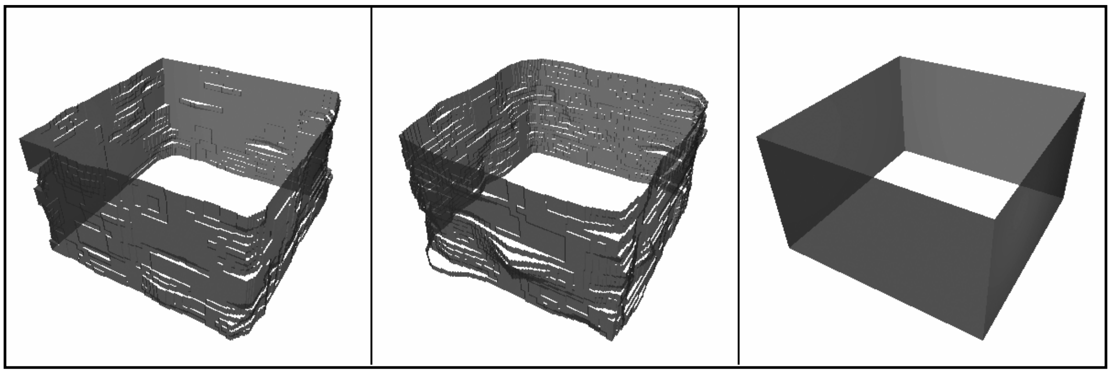

Fig.8. 3-D contour rendered as a collection of separate flat contours. The bigger the difference between the neighbouring contours, the most visible the empty spaces between them. When all the contours are identical (image on the right) the generated solid figure is uniform (without any holes).

"Top" and "bottom" of the contour are not generated, because they do not have any influence on its shape. They would constitute additional vertices, increasing storage occupancy. Moreover, a "covered" contour would obscure the object's inside.

There are algorithms that enable the constructing of a 3-D mesh of vertices on the basis of a set of flat contours. The mesh created in this way is treated as one solid body. The algorithms connect neighbouring contour layers to an appropriate number of triangles (surface triangulation) – the optimal mesh is constructed using graph theory. A detailed description of such an algorithm can be found in [7]. It was not implemented here, because the visualization of contour in the present paper was to give the user only a general idea about its shape and the degree of adjustment to the image. Rendering all the details of the contours and of the solid figure was not considered as relevant here. However, what was very important was the efficiency of the algorithm used to represent a contour, as the contour is rendered repeatedly during the optimisation process (in order to present the progress). Thus, time is a very important factor here.

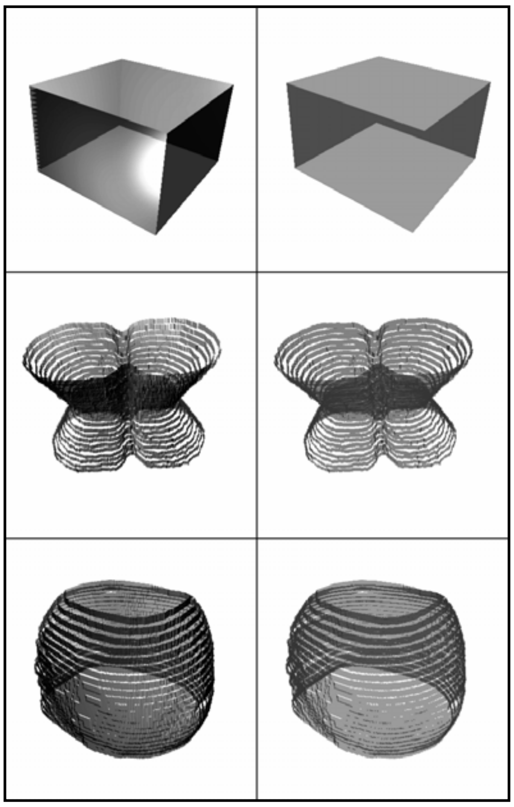
Fig.9. Comparison of contour rendering with shading (left column) and without shading (right column). Without shading, the object looks flat, mat and as if it was made of plastic.

In the implemented method the contour is semi-transparent so that it does not obscure the image. Lighting and shading helps to render the impression of depth (see Figure 9).

6. SUMMARY

Efficient visualization of 3-D objects can be a very time-consuming process. It also requires much storage space. Moreover, the user has to adjust the parameters manually, as full automatization is impossible due to the presence of noise and non-homogenous fragments in the structure of the image.

On the other hand, visualization methods are expected to be highly accurate, detailed, cheap, automated and flexible. Meeting all those expectations is extremely difficult. The present paper focused only on the simple methods of image and contour visualization. It was directed at physicians who want to expand their knowledge about possible visualization methods, their variability and associated problems.

7. ACKNOWLEDGMENT.

All the medical images used in the present paper were rendered on the basis of pictures that were made available by Norbert Barlicki Memorial Teaching Hospital in Lodz, Poland.

BIBLIOGRAPHY

[1] TADEUSIEWICZ R., KOROHODA P.: *Komputerowa analiza i przetwarzanie obrazów*. (*Computer Analysis and Processing of Images*). Wydawnictwo Fundacji Postępu Telekomunikacji, Kraków 1997.

[2] NIKOLAIDIS N., PITAS I.: *3-D Image Processing Algorithms*. 2001.

[3] PITAS I., Digital Image Processing Algorithms and Applications, Wiley, 2000.

[4] STYTZ M. R., FRIEDER G. FRIEDER O., Three-Dimensional Medical Imaging: Algorithms and Computer Systems, *ACM Computing Surveys*, vol. 23, no. 4, pp. 476-477, December 1991.

[5] JINXIAO PAN, TIE ZHOU, YAN HAN, MING JIANG: Variable Weighted Ordered Subset Image Reconstruction Algorithm. *International Journal of Biomedical Imaging*, vol. 2006, February 2006.

[6] KRIETE A., ed..: Visualisation in Biomedical Microscopies, VCH Weinheim, 1992.

[7] FUCHS H., KEDEM Z. M., USELTON S. P.: Optimal surface reconstruction from planar contours. *Communications of the ACM*, vol. 20, no. 10, pp. 693-702, October 1977.

[8] BOISSONAT J. D.: Shape reconstruction from planar cross sections. *Computer Vision, Graphics and Image Processing*, vol. 44, no. 1, pp. 1-29, 1988.

[9] KAUFMAN A., ed.: *Volume Visualization*. IEEE Computer Society Press, 1991.

[10] LEVOY M.: Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, vol. 8, pp. 29-37, 1988.

[11] LORENSEN W. E., Cline H. E.: Marching cubes: A high resolution 3D surface construction algorithm, *Computer Graphics*, vol. 21, no. 3, pp. 163-169, July 1987.

[12] GRZESZCZUK R., LEVIN D. N.: Brownian Strings: Segmenting Images with Stochastically Deformable Contours. *Proc. Visualization in Biomedical Computing*, pp. 72-89, Bellingham, Wash: Int'l Soc. Optical Eng., 1994