Correspondence

Volume Rendering of Segmented Image Objects

Elizabeth Bullitt* and Stephen R. Aylward

***Abstract*—This paper describes a new method of combining ray-casting with segmentation. Volume rendering is performed at interactive rates on personal computers, and visualizations include both "superficial" ray-casting through a shell at each object's surface and "deep" ray-casting through the confines of each object. A feature of the approach is the option to smoothly and interactively dilate segmentation boundaries along all axes. This ability, when combined with selective "turning off" of extraneous image objects, can help clinicians detect and evaluate segmentation errors that may affect surgical planning.**

We describe both a method optimized for displaying tubular objects and a more general method applicable to objects of arbitrary geometry. In both cases, select three-dimensional points are projected onto a modified z buffer that records additional information about the projected objects. A subsequent step selectively volume renders only through the object volumes indicated by the z buffer.

We describe how our approach differs from other reported methods for combining segmentation with ray-casting, and illustrate how our method can be useful in helping to detect segmentation errors.

***Index Terms*—Blood vessels, MR, segmentation, volume visualization.**

I. Introduction

Both computer-assisted surgical systems and computer-assisted diagnosis software often employ segmented image data. For clinicians to base decisions upon the displayed images, the segmentations must be consistently and sufficiently accurate. However, no segmentation method can guarantee accurate object definition under all circumstances.

The traditional method of displaying segmented objects is by surface rendering. Finding an intuitive method to display the degree of uncertainty of an object's surface within a three-dimensional (3-D) display system is difficult. One model for presenting this kind of uncertainty is available in programs used for Radiation Oncology planning. Schlegel's group, for example, evaluates dose distributions in both two-dimensional and 3-D [1], using isodose ribbons in a 3-D model of the patient's anatomy. However, such methods do not give confidence margins and the display of isodose ribbons in 3-D can be confusing.

An alternative method is to display the actual image data via volume rendering (ray casting), and to let the clinician make decisions on the basis of the actual image data shown. However, traditional volume rendering is performed through a block of space, requires more computational power than surface rendering, cannot selectively eliminate anatomical objects in order to better visualize an object of interest, and does not permit query or manipulation of individual anatomical objects.

Manuscript received April 14, 2001; revised June 5, 2002. This work was supported in part by the National Institutes of Health (NIH) under Grant R01CA67812 NIH-NCI and Grant P01CA47982 NIH-NCI and in part by Intel and Microsoft equipment awards. The Associate Editor responsible for coordinating the review of this paper and recommending its publication was M. Viergever. *Asterisk indicates corresponding author.*

*E. Bullitt is with the Division of Neurosurgery, University of North Carolina-CH, Chapel Hill, NC 27599 USA (e-mail: bullitt@med.unc.edu).

S. R. Aylward is with the Department of Radiology, University of North Carolina-CH, Chapel Hill, NC 27599 USA.

Digital Object Identifier 10.1109/TMI.2002.803088

This paper describes a method of combining segmentation with ray casting. We present both a specific method optimized for display of tubular objects and a more general approach applicable to objects of arbitrary shape. A preliminary description of the tubular object method was presented at MICCAI 2001 [2].

The proposed approach displays the actual image data, allows the user to "turn off" any obscuring object, and has the capacity to dilate segmentation margins so that the surrounding image data can be seen if the segmentation definitions are in doubt. The method can either display a "shell" of image data at each object's surface, or can volume render selectively through each object. Explicit polygonization of an object's 3-D surface is not required. Ray-casting can be combined with surface rendering. Without specialized hardware, readily interactive display rates are achieved on standard laptop computers, even under perspective projection.

II. Background

As stated by Zuiderveld [3], one means of classifying visualization methods is by division into "image-order" and "object-order" rendering. With object-ordered rendering, objects in 3-D space are projected onto the viewplane. A common approach employs a set of segmented structures, coats each structure's surface with a set of connected tiles, and then projects these surface tiles. The approach is fast, allows individual objects to be turned "on" or "off," and permits a variety of display options. However, the segmentation margins are hard-edged and the actual image data are not shown. If the surface definitions are in error, the displayed surfaces will be erroneous. The strengths of the approach are that display is fast and that individual objects can be manipulated. The weakness of the approach is that the utility of the display is dependent upon the accuracy of both the segmentation and of the surface polygonization, which the clinician may not trust.

Image-ordered rendering does not require prior segmentation. Instead, the process of visualizing medical image data often involves casting a ray from the ray source to each viewplane pixel, with calculation of the value of each pixel's intensity based upon its ray's intersections with a 3-D block of image data. The advantage of the approach is that the actual image data are shown. The disadvantages of the approach are its large computational requirements, and that individual anatomical objects cannot be manipulated directly.

It would be advantageous to combine the strengths of the two methods. Zuiderveld [3] does so by employing a predominantly image-ordered method, using segmentation to speed the process. Given a set of segmented objects, he prelabels each voxel with the minimum distance to the next voxel of interest, and thus, during ray casting, avoids time-consuming processing of uninteresting voxels. This is a highly flexible method that permits a variety of visualizations, while also dramatically increasing ray-casting speed. However, relabeling space may be required if a request is made to view more of the actual image data around all margins of a segmented object.

The "shell" rendering method of Udupa, Odhner and colleagues [4] employs image data to enhance a predominantly object-ordered approach. The method first constructs a tiled surface for each segmented object, and then collects image data at an arbitrary depth about each object's surface and in the direction normal to each tile. The image data can then either be splatted onto the viewplane or texture mapped to the surface tiles. Once the objects' surfaces are defined and the surface data collected, rendering is extremely rapid, and can greatly exceed even the

rates provided by dedicated hardware boards [4]. This approach appears to work particularly well for tubular objects [5]. It also permits at least partial dilation of segmentation boundaries, via adjustment of the depth over which image data is collected in the direction orthogonal to each tile's surface.

Three disadvantages of the Udupa method, however, are that (as discussed later) the process of explicitly defining a surface mesh may itself produce errors, that it is not possible to cast a ray selectively through an object of arbitrary geometric configuration, and that the collection of image data for more than a short distance across a facet's surface may produce discontinuities in the textures mapped to adjacent tiles. Moreover, as the method of dilating object boundaries applies only to the direction normal to each facet, the portions of space imaged will change as the object is rotated, with additional information seen predominantly in the current depth (z) direction rather than in the more easily perceived in-plane (x–y) directions. It would be preferable to provide a more uniform method of boundary dilation that applies equally along all axes.

This paper describes a third approach to combining segmentation with ray-casting. Our method combines object-ordered and image-ordered rendering techniques. An object-ordered approach is used to project selected points onto the viewplane, followed by an image ordered process to cast selected rays through a defined volume of space.

When compared with the implementations of Zuiderveld or Udupa, our technique is more computationally complex. However, our approach permits clinically useful visualizations that are not directly supported by the other two methods, and runs at readily interactive rates on a Pentium III, 850-MHz computer under Windows 2000 and without specialized hardware.

III. Methods

A. Image Acquisition

The visualization methods described in this paper are applicable to any type of 3-D image data. For this paper, we include objects extracted from 3-D, time-of-flight, magnetic resonance angiograms (MRA) of the head, gadolinium-enhanced magnetic resonance image of the head, contrast enhanced computed tomography (CT) of the abdomen, and a 3-D rotational digital subtraction angiogram (3-D-DSA) of the head that was donated by Siemens.

The size of these datasets varied. Most images were 256×256 with approximately 100 slices (40 to more than 300). Two MRA studies were obtained at high resolution, and each contained $512 \times 512 \times 108$ voxels ($0.5 \times 0.5 \times 1$ mm resolution).

B. Segmentation

Two visualization approaches are described in this paper: a general method applicable to segmented objects of arbitrary configuration, and a specific method optimized for tubes. The general visualization method is independent of the type of segmentation employed. Instead, it takes as input a "mask" file, in which uninteresting voxels are labeled "0," voxels belonging to object 1 are labeled "1," objects belonging to object 2 are labeled "2," etc. Any segmentation approach that produces this kind of output file can be employed.

For this paper, abdominal organs were segmented from contrast-enhanced CT using connected components and mathematical morphology [6]. The brain tumors were extracted using a partially manual program that provides polygon drawing and filling on orthogonal cuts through an image volume. The time to segment the organs or groups of organs shown in this paper ranged from 5 min to over 1 h.

Tubular objects were segmented using the tubular extraction method of Aylward [6], post-processed to provide vessel "trees" [7]. Vessel segmentation involves three steps: definition of a seed point, automatic extraction of an image intensity ridge representing the vessel's central skeleton, and automatic determination of width at each skeleton point. Tests of the vessel extraction method using phantom data, a variety of input parameters, and varying degrees of noise, indicate it to provide consistent extractions with subvoxel accuracy and with relative insensitivity to the input parameters, even under noisy conditions [6]. The tree-creation process requires the user to select one or more vessel "roots," and then automatically builds a set of trees from the segmented vessels, using both distance and image intensity information to determine the connections. In a recent clinical study, two radiologists analyzed seven vascular trees with comparison to DSA data obtained from the same patients. Of the 140 connections evaluated, only one was judged by both radiologists to be incorrect [7]. It takes approximately 15 min to define a set of vessel trees from an intracranial MRA, and approximately 5 min to define the independent vessel trees supplying and draining the liver from abdominal CT.

The output of the vessel definition processes is a set of vessels in which each four-dimensional "vessel point" is represented as an (x, y, z) spatial position along the vessel's medial axis, with an associated radius at each point.

C. Overview of Rendering Methods

Both our general rendering method and our tube-specific rendering method depend upon a modified z buffer. This z buffer stores not only the depth of each projected point, but also the object's identification number, the identification number of the point within the object, and a "radius" associated with the projected point. Of note, a byproduct of the approach is that fast pick operations can be performed upon displayed objects, since all necessary object information is already stored for each pixel.

We view all 3-D points as spherical, and assume that the projections of these points are circular upon the viewplane. The rendering method begins by projecting the spatial location of selected points upon a modified z buffer. If two points superimpose on projection, the one closer to the viewplane is given priority. Given the depth coordinate of each 3-D point, knowledge of that point's radius, and knowledge of the viewing geometry, the radius of each projected point can then be calculated on the viewplane. Using a second modified z buffer, a fast Bresenham circle algorithm [8], modified to provide filled circles, is used to define the associated viewplane pixels to which rays should project. If another object has already been identified as projecting to a portion of a filled viewplane circle, the object closest to the ray source is given priority. Rays are subsequently cast only to labeled viewplane pixels and only through the 3-D sphere projecting to that pixel, using the radius value stored in the z buffer to help determine the length of the ray segment over which image data should be collected.

D. Rendering of Arbitrary Objects

For rendering arbitrary objects, the "mask file" given as input to the program is of the same dimensions as the image data file. In the mask file, all uninteresting voxels are labeled "0," all voxels belonging to object 1 are labeled "1," all voxels belonging to object 2 are labeled "2," etc. On load of a mask file, the program determines the "boundary voxels" for each segmented object. A "boundary voxel" is defined as one that has a nonzero value and which either lies on a surface of the total volume block or possesses a face abutting a voxel of a different value. The boundary voxels are collected and stored for each object.

Two rendering modes are supported: "superficial," and "deep." The superficial mode visualizes a shell of image data at the margin of segmented objects. The deep mode casts rays through the confines of each object. Fig. 1 provides an example of the difference between these two

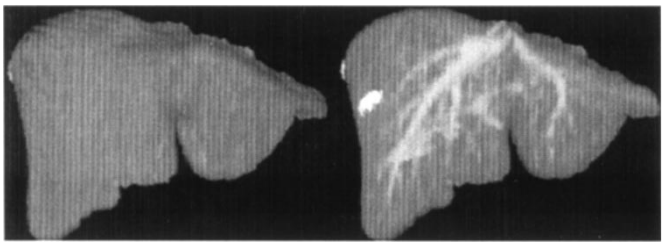

Fig. 1. Superficial rendering of a liver (left) and deep rendering (right). With deep rendering under MIP, brighter vessels or pathological entities can be seen within the segmented object's parenchyma.

visualizations, using the example of a liver defined from CT and as shown by maximum intensity projection (MIP). All extraneous objects have been "turned off" so as to better view the liver.

For superficial rendering, the centers of the boundary voxels are projected upon a modified z buffer, which stores a. default "radius" of half the diagonal of a voxel. Filled viewplane circles are then defined upon a second modified z buffer, and rays are subsequently cast only to labeled viewplane pixels and only through the closest 3-D sphere (voxel) of interest projecting to that pixel.

Deep volume rendering requires knowing, for each arbitrarily shaped object, where each ray should "start" and "stop." In addition to the front z buffer, a back buffer is defined for each object, and rays are cast selectively for each marked pixel. Image intensity information is then calculated for each ray from the positions indicated by the corresponding pixels on the two z buffers and for the object reported by the front buffer.

Arbitrary dilation of segmentation margins can be achieved by increasing the 3-D "radius" value stored in the z buffers. Increasing this "radius" will not only increase the diameters of the circles drawn upon the viewplane, but will also extend the length of the ray segment over which image intensity information is collected. Boundary dilation can be applied both to the superficial and to the deep rendering methods.

E. Rendering of Tubular Objects

Objects with circular cross-sections can be rendered particularly efficiently because each tube point is already associated with an (x, y, z) spatial location and a radius. For superficial vessel rendering, the vessel skeleton locations are projected upon the modified z buffer in the same fashion that voxel centers are projected for arbitrary objects. Each vessel point's own radius is used as the default radius stored in the z buffer, and used during viewplane circle calculation. During ray casting, image data is collected along the ray segment centered upon the z plane of the vessel skeleton point, and extending for a distance "radius" on each side of that point. Arbitrary boundary dilation is achieved by multiplying the radius value stored in the z buffers by an arbitrary amount.

A problem can arise, however, when tortuous vessels are volume rendered under conditions of radius dilation. Since closer points overwrite more distant points, the drawing of empty space surrounding an "expanded radius" vessel point may produce dark pixels that overwrite high intensity vessel points more distant from the ray source. The result can produce an apparent discontinuity in the rendered vessel (Fig. 2). A solution to the problem is to provide an alternate viewing method, similar to the "deep" rendering method described for general objects. In addition to a front buffer, a back buffer is provided for each object. For each pixel of interest, rays are then cast not for a distance $\pm$ radius around a central skeleton point, but rather from the locations indicated by the corresponding pixels on the two buffers (Fig. 2).

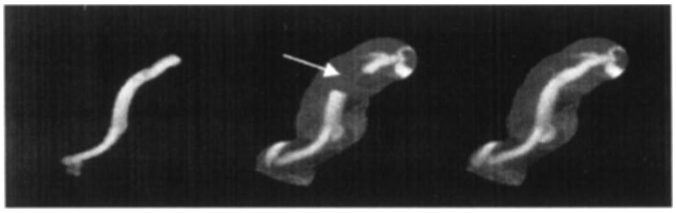

Fig. 2. Tubular objects with dilated segmentation boundaries. (left) A single vessel "superficially" volume rendered using the boundaries defined by segmentation. (center) Visualization of the same vessel with radius dilation by a factor of 4.5. Closer points overwrite more distant points, leading to an apparent discontinuity in the vessel (arrow). (right) Visualization of the same vessel with radius dilation by a factor of 4.5 and using "deep" volume rendering.

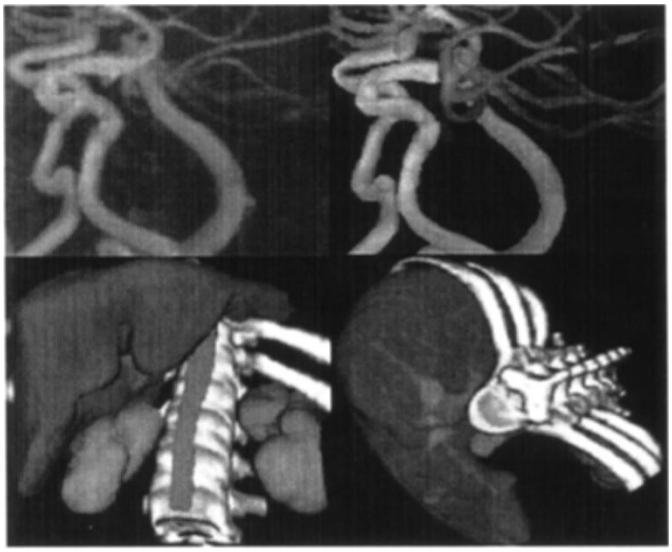

Fig. 3. Depth information on individual frames. The use of MIP when combined with traditional "block" volume rendering can produce confusing results (upper left), since depth information is lost and brighter objects always appear closer to the observer. The described approach provides depth information.

IV. Results

A. Image Quality and Informational Content

In common with other approaches that combine ray-casting with segmentation, our method allows the use of MIP while preserving depth information. Fig. 3 provides two examples. The upper pair of images compare a traditional "block" volume rendering of a portion of the intracranial circulation defined from MRA (left) with our approach (right). The lower image pair shows abdominal organs segmented from CT and displayed from two different points of view. Depth information is clearly available on each frame.

As previously noted, our approach facilitates "pick" operations, since each pixel is labeled with the object projecting to it. Individual objects or groups of objects can, thus, be rapidly selected and manipulated. Fig. 4 provides an example of how the vessel tree structure can be used to clarify feeding patterns to an arteriovenous malformation (AVM; a tangle of abnormal blood vessels). A single point and click can turn off an entire subtree, thus reducing projection overlap and aiding study of the 3-D vascular distribution patterns. In this example, the vessels were defined from 3-D-DSA.

One advantage of the approach is that it does not require explicit construction of a surface mesh for segmented objects. Such construction can sometimes produce errors of its own. For example,

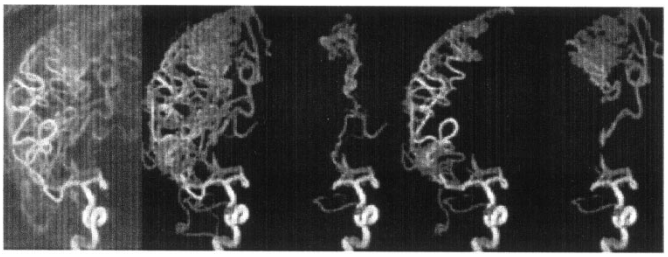

Fig. 4. User-directed manipulation of segmented, volume-rendered objects. (left) A standard block MIP volume rendering of an AVM is shown. The vasculature is complex, overlapped on projection, and difficult to understand. The adjacent pictures depict volume rendering via segmented objects. User-directed "point and click" directives to hide or display vessel subtrees can clarify the vascular supply to the lesion.

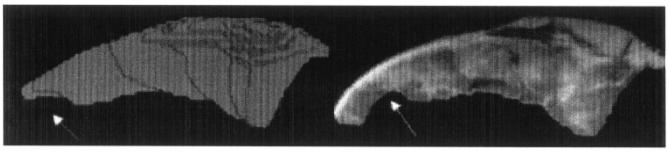

Fig. 5. Error produced by surface definition of a brain tumor using marching cubes. The thin tail of the tumor is missing (arrow at left). (right)Our approach includes segmented objects of a voxel's thickness.

the "marching cubes" algorithm [9] is an effective and widely used method for defining object surfaces. A limitation, however, is that voxel thick structures may fail to be included in the surface definitions.

Fig. 5 provides an example of a brain tumor with a thin "tail." This tail was omitted during a marching-cubes implementation of surface definition [10]. If surgical or radiation therapy planning had been based upon the displayed surfaces, an error might have resulted. The proposed approach does not suffer from this limitation, and will correctly visualize single voxels included in the segmentation.

B. Detection of Segmentation Errors

Our approach permits both the "turning off" of obscuring image objects and interactive dilation of segmentation boundaries so that more of the actual image data can be shown. Such dilation can be done either by superficial or deep volume rendering. Fig. 6 provides three examples of how this approach can be used to detect possible segmentation errors that might affect therapeutic intervention.

In the first case (upper left in Fig. 6), the segmentation of a distal middle cerebral artery aneurysm failed to disclose an aneurysm neck. The question was raised whether such a neck might actually exist. Dilation of the segmentation margins failed to disclose a neck, as was subsequently confirmed by angiography. The initial segmentation was, therefore, correct. However, the ability to expand segmentation margins and to view more of the actual image data without obscuration by other objects can be profoundly reassuring to the clinician. Moreover, the ability of the proposed approach to provide "deep" volume rendering under conditions of radius dilation avoids problems such as that illustrated by Fig. 2.

The upper central image in Fig. 6 illustrates a kidney whose initial segmentation did not include the hilum. With simultaneous elimination of obscuring image objects and with dilation of the kidney's segmentation boundaries, the hilum is correctly shown (lower central image, arrow).

The upper right image in Fig. 6 depicts a complex intracerebral AVM, whose nidus was defined by connected components and which is here shown by deep volume rendering. The vessels leading to and draining the nidus were also extracted and are displayed. Expansion of

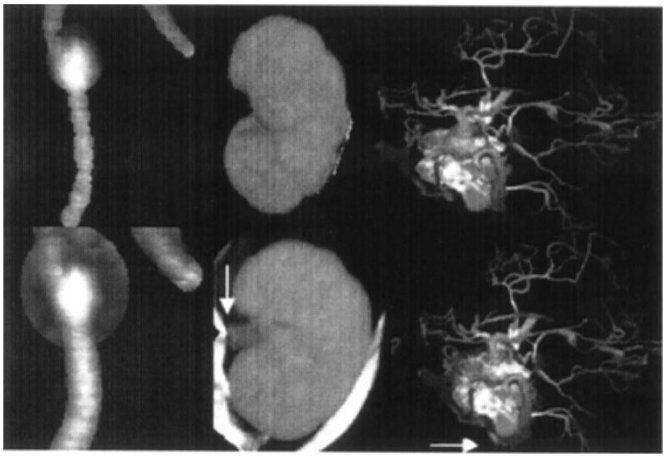

Fig. 6. Use of segmentation boundary dilation to detect segmentation errors.

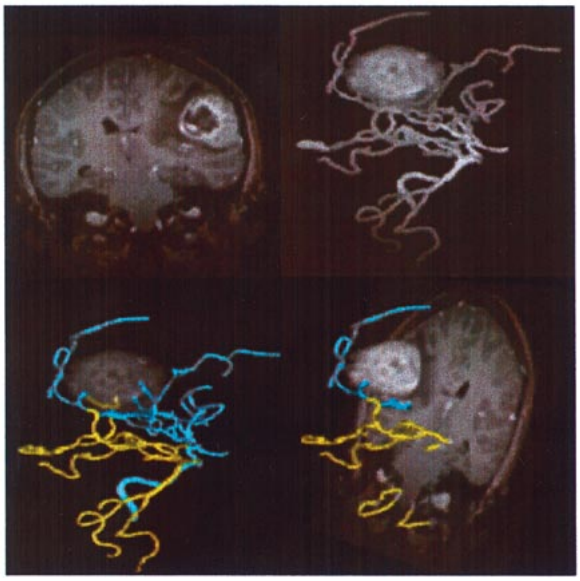

Fig. 7. Combining surface and volume rendering to aid perception of the relationship between objects.

the nidus' segmentation boundaries shows a vessel (lower right image, arrow) that was missed during the initial vessel segmentation.

When 3-D image data are displayed for clinical planning purposes, it is often important not only to know the precise delineation of relevant objects, but also to understand the relationships between objects. A disadvantage of ray-casting is that the use of only a greyscale palette can make it difficult to visualize the relationship between adjacent or interpenetrating objects of similar image intensities. The use of color and variable opacity can often be helpful in clarifying these relationships. It, therefore, should be noted that our approach can be combined with colored, surface renderings by modifying the surface rendering protocols to write to the same modified z buffers used by the volume rendering technique.

Fig. 7 provides an example. The patient had a brain tumor (upper left) whose vascular feeding pattern was unclear. With volume rendering alone, it is difficult to determine which of two major vascular subtrees enter the tumor, as the vessel and tumor intensities are similar (upper right). The figure at lower left shows the same two vascular subtrees, now surface rendered in different colors, and the tumor volume rendered at partial opacity. When seen in a 3-D display, it is evident that the gold posterior circulation sends an abnormal branch into the tumor.

The figure at lower right illustrates the simultaneous display of colored, surface rendered vessels, a volume rendered tumor at full opacity, and a coronal slice also set to full opacity.

C. Rendering Speed

The approach described is currently implemented only in software and does not yet even employ Open GL's texture mapping capabilities [11], [12]. Nevertheless, the method runs at easily interactive rates. As previously reported, we can render segmented MRAs of the whole head ($256 \times 256 \times 100$ voxels) at five frames/s under perspective projection and into a 500×500 display window, using a personal computer (850-MHz, Pentium III) without specialized hardware [1]. Superficial rendering of arbitrary objects operates at comparable rates. These rendering times are over an order of magnitude faster than traditional "block" volume rendering of the same image data when only a software approach to ray casting is employed. The display rate of five frames/s permits clinical users to employ the programs interactively [13].

V. Discussion

This paper describes a new method of displaying anatomical objects through a combination of volume rendering and segmentation. We believe that the proposed approach can offer useful visualizations not easily possible by other techniques. In particular, the method permits rapid and selective dilation of segmentation boundaries along all axes, with the option of viewing image data collected either as a shell about an object's surface or via "deep" ray casting, in which the start and stop positions of each ray are defined relative to the object's 3-D boundaries.

Several limitations of the approach should be noted, however. A first, obvious limitation is that dilation of segmentation boundaries is not an efficient means to search for distant objects that should have been segmented but were not. The method, therefore, should not be relied upon to seek for missing structures in unknown positions. Nevertheless, all clinical applications, except those used for screening purposes, tend to focus upon a particular set of structures or problems. Our approach can be helpful in reassuring the clinician that the initial segmentation was indeed correct, or in warning the clinician that the initial segmentation may have missed important information.

A second and perhaps more important limitation is that the current implementation employs boundary limited maximum intensity projection. When "deep" volume rendering is used, this visualization allows perception of higher intensity vessels or lesions within a structure whose parenchyma is of lower intensity (Fig. 1). However, this approach will not easily depict the margins of lower-intensity entities lying within higher-intensity parenchyma, unless such structures have been previously segmented and are shown in another way (Fig. 7).

Several points should be made about rendering speed. We have not attempted to compare the rendering time of our own technique with the techniques of other investigators who have also combined segmentation with volume rendering. State-of-the-art approaches to increasing volume rendering speed often incorporate hardware optimizations into their solutions, and our own approach is currently implemented only in software. It is, therefore, not possible to make a fair comparison between the two methods. Similarly, although the Udupa shell mapping approach is likely to require more set-up time than our own, once the set-up is complete no further ray-casting is required, and so the method will perform much more rapidly than the approach described here.

In summary, we present a new method for combining segmentation with volume rendering. The program is cross-platform, runs well on the PC, and, even without optimization, provides an effectively fast means of visualizing the actual image data in a variety of potentially useful ways that would be difficult to implement by other approaches. We hope and believe that the proposed method will be useful to clinicians to detect and evaluate segmentation errors during clinical planning.

References

[1] R. Bendl, J. Pross, A. Hoess, M. Keller, K. Preiser, and W. Schlegel, "VIRTUOS—A program for VIRTUal radiOtherapy simulation and verification," in *Proc 11th Int. Conf. Use of Computers in Radiation Therapy*, Manchester, U.K., 1995, pp. 226–227.

[2] E. Bullitt and S. R. Aylward, "Volume rendering of segmented tubular objects," in *Lecture Notes in Computer Science*, W. J. Niessen and M. A. Viergevier, Eds. New York: Springer-Verlag, 2001, vol. 2208, MICCAI 2001, pp. 161–168.

[3] K. Zuiderveld, "Visualization of multimodality medical volume data using object-oriented methods," masters thesis, Universiteit Utrecht, Utrecht, The Netherlands, 1995.

[4] G. Grevera, J. Udupa, and D. Odhner, "An order of magnitude faster isosurface rendering in software on a PC than using dedicated, general purpose rendering hardware," *IEEE Trans. Visual. Comput. Graphics*, vol. 6, pp. 335–345, Oct./Dec. 2000.

[5] T. Lei, J. Udupa, P. Saha, and D. Odhner, "Artery-vein separation via MRA—An image processing approach," *IEEE Trans. Med. Imag.*, vol. 20, pp. 689–703, Aug. 2001.

[6] S. R. Aylward and E. Bullitt, "Initialization, noise, singularities and scale in height ridge traversal for tubular object centerline extraction," *IEEE Trans. Med. Imag.*, vol. 21, pp. 61–75, Feb. 2002.

[7] E. Bullitt, S. R. Aylward, K. Smith, S. Mukherji, M. Jiroutek, and K. Muller, "Symbolic description of intracerebral vessels segmented from MRA and evaluation by comparison with X-ray angiograms," *Med. Image Anal.*, vol. 5, pp. 157–169, 2001.

[8] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice*. Reading, MA.: Addison-Wesley, 1997, pp. 83–87.

[9] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface reconstruction algorithm," in *Computer Graphics, (Proc. of SIGGRAPH)*, vol. 21, 1987, pp. 163–169.

[10] W. Schroeder, K. Martin, and B. Lorensen, *The Visualization Toolkit*, New Jersey: Prentice Hall, 1998. [Online]. Available: http://www.kitware.com/vtk.html, Open source software.

[11] P. Hastreiter, C. Rezk-Salama, K. Eberhardt, B. Tomandl, and T. Ertl, "Functional analysis of the vertebral column based on MR and direct volume rendering," in *Lecture Notes in Computer Science*, S. L. Delp, A.M. DiGioia, and B. Jaramaz, Eds. New York: Springer-Verlag, 2000, vol. 1935, MICCAI 2000, pp. 412–421.

[12] B. Cabral, N. Cam, and J. Foran, "Accelerated volume rendering and tomographic reconstruction using texture-mapping hardware," in *Proc. ACM Symp. Volume Visualization*, 1994, pp. 91–98.

[13] E. Bullitt, S. R. Aylward, E. Bernard, and G. Gerig, "Computer-assisted visualization of arteriovenous malformations on the home PC," *Neurosurgery*, vol. 48, pp. 576–583, 2001.