# Visualizing Inner Structures in Multimodal Volume Data

ISABEL H. MANSSOUR[1,2], SÉRGIO S. FURUIE[3], SILVIA D. OLABARRIAGA[1], CARLA M.D.S. FREITAS[1]

[1]UFRGS–Universidade Federal do Rio Grande do Sul, Instituto de Informática,
Caixa Postal 15064, 91501-970 Porto Alegre, RS, Brasil
[2]PUCRS– Pontifícia Universidade Católica do Rio Grande do Sul, Faculdade de Informática -
Av. Ipiranga 6681, Prédio 30, Bloco 4, 90619-900 Porto Alegre, RS, Brasil
[3]InCor–Instituto do Coração, HC-FMUSP, Unidade de Pesquisa e Desenvolvimento -
Av. Dr. Enéas de Carvalho Aguiar, 44, 05403-000 São Paulo, SP, Brasil
{manssour,silvia,carla}@inf.ufrgs.br, Sergio.Furuie@incor.usp.br

**Abstract.** With the evolution of medical image acquisition techniques, the capacity and fidelity of image-based diagnosis were extended. The current trend is to acquire information using multiple sources to help medical diagnosis, but the integration of the multivariate data into a single 3-D representation is non-trivial. Techniques for the visualization of multimodal volume data have been developed with the goal of finding suitable strategies to integrate characteristics of multiple data sets into a single visual representation. Likewise, several techniques are dedicated to the exploration of different ways of incorporating seeing-through capabilities into volume rendering techniques. This paper presents a new approach to visualize inner structures in multimodal volume data, which is based in the utilization of cutting tools.

## 1 Introduction

Volume visualization techniques are related with the extraction of meaningful information from complex 3-D data sets, providing different ways to exhibit and explore the interior of volume data, and enabling the identification of its inner regions and structures. The approaches for seeing-through volumes described in the literature assume that (a) we can see through transparent voxels; or (b) we remove voxels that are in front of the region where the user is interested.

Several image acquisition modalities have been used to facilitate the medical diagnosis, e.g. Positron Emission Tomography (PET), Computed Tomography (CT), and Magnetic Resonance Imaging (MRI). While modalities such as PET help to determine how the body functions, images from CT and MRI aid in the identification of anatomic structures. These modalities show different, complementary and/or partially overlapping aspects of the examined organ [12].

Despite the increasing use of information acquired from multiple sources, the combination of multivariate data into a 3-D single representation of the patient is extremely difficult, time-consuming and error-prone. The proper integration of images from multiple modalities therefore requires enhanced visualization techniques that explicitly address the combination of data from different volumes into a single view. For example, it would be useful to visualize simultaneously two volumes (MRI and PET) representing the same patient organ. The inspection of both volumes could reveal more information than the individual volumes, increasing the confidence of the observers in the location of a functional abnormality (PET) in relation to the anatomy (MRI) [20]. Some examples of applications that might benefit from multimodal data visualization are: analysis of regional brain activity in patients suffering from schizophrenia [11]; radiotherapy treatment planning [18]; and surgical planning [20].

Although the goal of physicians usually resides in visualizing and quantifying isolated structures inside volume data, there is a lack of tools for selecting the region-of-interest (ROI) and allowing for the visualization of inner structures in multimodal volumes. In this paper we describe a new approach for the visualization of inner structures in multimodal data sets as an extension of the ray casting algorithm, which allows for the simple integration of images from multiple modalities and the incorporation of several cutting tools as a seeing-through capability. The goal is to provide a powerful and general set of tools for interactive inspection of inner structures in multimodal volume data. The integration of these tools into the ray casting framework is straightforward and simple to implement via the manipulation of a few parameters.

The paper is organized as follows: Section 2 describes the basics of volume rendering using the ray casting algorithm, and presents existing techniques for seeing through volumes and for visualizing multimodal volume data. The proposed approach is described in Section 3. Section 4 presents some applications of the developed tools and a comparison with some existing techniques. Conclusions and future work are presented in Section 5.

## 2 Visualizing Volumes

Direct volume rendering denotes a set of techniques used to directly display volume data, where the images are generated through the transformation, shading, and projection of 3-D voxels onto 2-D pixels [5]. A subset of direct volume rendering techniques is based on the ray casting algorithm introduced by Levoy [8] and briefly described in Section 2.1.

### 2.1 Ray Casting

The ray casting algorithm starts spanning the image window pixels, and fires rays from the observer into the volume data, one ray per pixel. The main elements are the volume data and their properties, the viewing parameters, the image window and the rays where the sample points are processed. The volume data $V(\mathbf{x})$ refer to values $\mathbf{v}$ associated to discrete sampling positions $\mathbf{x} = [x, y, z]$ often regularly spaced in 3-D, with $x \in [1, N_x]$, $y \in [1, N_y]$ and $z \in [1, N_z]$. The values $\mathbf{v}$ corresponds to one or more data properties (e.g. $V(\mathbf{x})$ is a scalar representing density in CT).

The viewing parameters traditionally specify the observer's position, the projection type, the direction of projection, the projection plane, and the 3-D clipping planes. In the context of volume rendering, the direction of projection $\overrightarrow{DOP}$ specifies the orientation of rays cast into the volume data. The projection plane is mapped onto a window $I$ in the image plane, which refers to the intensity values associated to the pixels position $\mathbf{u} = [u, v]$, with $u \in [1, N_u]$ and $v \in [1, N_v]$ (Figure 1).

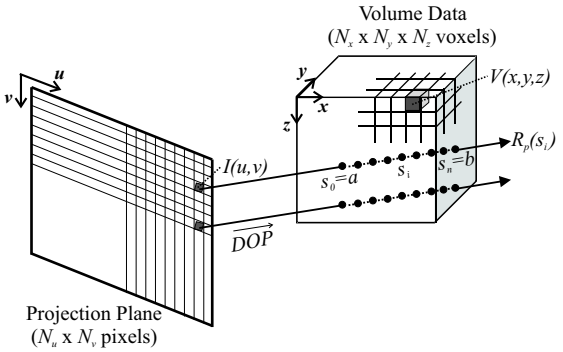


Figure 1: General scheme of a ray casting algorithm.

The intensity for each image pixel $I(\mathbf{u})$ is specified by casting a ray $\mathcal{R}_p$ from that pixel along the viewing direction $\overrightarrow{DOP}$. Samples $s$ are processed along the ray and a classification process assigns optical properties (color and opacity) to each volume sample. The optical properties are combined to determine the final pixel color. We represent this process as:

$$I(\mathbf{u}) = g(\mathcal{R}_p) \tag{1}$$

where $g$ is a function that determines the pixel intensity based on the samples of $V$ along the ray $\mathcal{R}_p$.

In this work a parametric notation is adopted for the ray $\mathcal{R}_p$. We denote as $s_i$ each sampled position along the ray path when navigating from pixel $I(\mathbf{u})$ towards $V$, with $i \in [0, n]$, and $n + 1$ being the total number of samples. The volume samples are therefore denoted by $V(\mathbf{x}_{s_i})$. Note that some interpolation strategy should be used for samples that do not belong to the discrete volume grid (for example, trilinear interpolation [10]).

Usually (e.g. [8]), $g$ is implemented as an accumulation function:

$$g(\mathcal{R}_p) = \sum_{s_i=a}^{b} f(\mathbf{x}_{s_i}), \tag{2}$$

where $s_0 = a$ is the first sample (front) that contributes for $I(\mathbf{u})$, $s_n = b$ is the last sample (back), and $f(\mathbf{x}_{s_i})$ is the function that specifies the contribution of each sample to the final pixel intensity based on the corresponding optical properties of sample $V(\mathbf{x}_{s_i})$.

In Figure 1 and Equation 2 we considered that the ray casting algorithm processes samples in a *front-to-back* order, from sample point $a$ towards $b$. However, it is also possible to do the computation from $b$ to $a$, in a *back-to-front* order.

In the RGB$\alpha$ approach introduced by Levoy [8], $f$ determines the RGB values by mapping voxels values into color components and opacity and combining them with a simple illumination model using a back-to-front order. In a later work [9], however, Levoy showed that the calculation of color and opacity values can be expressed recursively using a front-to-back composition. The function $g$ would then be implemented as:

$$g(\mathcal{R}_p) = C(\mathbf{x}_{s_n})/\alpha(\mathbf{x}_{s_n}) \tag{3}$$

with

$$C(\mathbf{x}_{s_i}) = \begin{cases} f_C(\mathbf{x}_{s_i}) f_\alpha(\mathbf{x}_{s_i}), & i = 0 \\ C(\mathbf{x}_{s_{i-1}}) + f_C(\mathbf{x}_{s_i}) f_\alpha(\mathbf{x}_{s_i})(1 - \alpha(\mathbf{x}_{s_{i-1}})), & i > 0 \end{cases} \tag{4}$$

$$\alpha(\mathbf{x}_{s_i}) = \begin{cases} f_\alpha(\mathbf{x}_{s_i}), & i = 0 \\ \alpha(\mathbf{x}_{s_{i-1}}) + f_\alpha(\mathbf{x}_{s_i})(1 - \alpha(\mathbf{x}_{s_{i-1}})), & i > 0 \end{cases} \tag{5}$$

where $C(\mathbf{x}_{s_i})$ and $\alpha(\mathbf{x}_{s_i})$ are, respectively, the pixel color and opacity values accumulated up to $s_i$, $f_C(\mathbf{x}_{s_i})$ and $f_\alpha(\mathbf{x}_{s_i})$ are the transfer functions that determine color and opacity based on voxel density, and $C(\mathbf{x}_{s_n})$ and $\alpha(\mathbf{x}_{s_n})$ are the final pixel color and opacity.

### 2.2 Visualizing Multimodal Data

Visualization techniques that integrate multimodal data volumes for the generation of one image are concerned with the proper combination of different, complementary and superposed features existent in each data volume. The first

and fundamental step to generate images from multimodal volumes consists of bringing the involved modalities into spatial alignment, a procedure called registration. After registration, a fusion step is required for the simultaneous display of the two data sets. The terms fusion and "data intermixing"[2] are used here to refer to the process of visualizing multimodal volumes.

Considering the classical ray casting technique (Section 2.1), in the visualization of multimodal data there is more than one value **v** associated with each sample point $s_i$. These values correspond to several data properties, each one belonging to a different volume $V$. The function $f$ used to process each pixel intensity $I(\mathbf{u})$ (Equation 2) has a different implementation, since color and opacity values are specified considering two or more sample values. For example, it is possible to use one volume as a parameter to the opacity transfer function to represent the anatomy, and the other one as a parameter to the color transfer function. Several approaches for this kind of visualization have been proposed and some of them are briefly described below.

In the *Linked Feature Display* [7, 20], there is a correlation between a 3D location and the equivalent position in the original 2D image of each acquisition modality. The images from each modality are presented in separate windows, and a linked cursor indicates corresponding locations in the image slices of different modalities. For example, as a mouse-driven line-cursor is moved through a 3D model, two cross-sectional images (e.g. MRI and PET) of the corresponding section are updated on the screen.

*Surface Texturing and Mapping* techniques integrate the information by mapping parts of the functional information from the volume onto a surface. Zuiderveld [23] and Stokking [20] implemented an algorithm using this technique for integrated visualization. In the "Normal Projection Technique" [23], the functional values were mapped onto a surface extracted from an anatomical data volume, e.g. MRI. The "Normal Fusion Technique" [20], also included the use of the Hue-Saturation-Value (HSV) color model, allowing for the manipulation of color tables by users according to their perception and preferences.

The *Spectral Volume Rendering* technique is based on direct volume rendering, and uses the full color spectra and physically realistic light/matter interaction models [16]. The difference between this technique and traditional ray casting algorithms is that it simulates light interaction with the materials inside a voxel to calculate the final color. In this approach, a material density is assigned to each voxel value, instead of a RGB color and opacity. Several materials can be used for multimodal images.

In the *Data Intermixing* technique [2], data integration can be done in different levels of the direct volume rendering pipeline based on the classical ray casting technique. The data flow in the rendering pipeline through three different stages: (1) geometric transformation; (2) integration-in-depth, which is composed by several steps (traversing, sampling and interpolation, illumination, accumulation, and pixel intensity and opacity determination); and (3) mapping. Data intermixing may be performed in different steps in the second stage: image level, when two rendering images are merged; accumulation level, when sample values are calculated in each volume along a ray and their visual contributions are mixed; or illumination model level, which consists in opacity and intensity calculation at each sampling point directly from a multi-volume illumination model.

### 2.3 Visualizing Inner Structures

Many efforts described in the literature are devoted to exploring different ways of incorporating seeing-through capabilities into volume rendering techniques. Two main approaches are identified in the existing techniques: controlling voxel transparency during the classification step and removing the voxels that are not in the region-of-interest.

In the first approach, *controlling voxel transparency*, the general idea is to reduce the visibility of structures that are not interesting and enhancing those that are relevant for a given study. This basically means designing adequate transfer functions (Equation 2) to assign high transparency levels to the structures outside the ROI - see Lichtenbelt et al. [10]. The design of transfer functions, however, is usually a difficult task - see a discussion in [17]. Since the trial and error approach is not suitable for identifying the best transfer function, automatic and semi-automatic techniques are being investigated (e.g. [6]). Another problem is that sometimes it is not suitable to visualize a structure of interest. Since transparency is usually defined as a function of voxel values, it is not possible to see a structure of interest if it has the same value as an obstructing structure [4].

The use of cutting techniques, i.e. *removing uninteresting voxels*, is another alternative for visualizing inner structures. In a complementary way, this approach can be treated as a selection task, where just the voxels that are inside the ROI are selected. The task of removing (or selecting) voxels can be done basically with two different types of techniques: image segmentation and volume cutting [22].

Image segmentation, i.e. subdividing an image into its constituent parts or objects until all the objects of interest for a given application are isolated [3], is acknowledged as a very difficult task. Very often it is desirable to avoid this step, or at least postpone it for after a first inspection of the volume data with other techniques that are easier to apply. This is the case of volume cutting techniques, which remove volume portions that we do not want to see, revealing the ROI. For example, a clipping plane can be used to cut through the volume, exposing a new surface and enabling the visualization of internal objects. This concept was in-

troduced as volume slicing and it consists in visualizing the volume data as a single 2-D image slice orthogonal to one of the three major axes or parallel to the view plane [15]. This main idea has evolved into more flexible cutting tools that can be used to remove blocks of voxels in several ways, as discussed in section 3.2.

## 3 Proposed Approach

This section presents a new approach for the visualization of inner structures in multimodal volumes as a generalization of volume cutting tools for multimodal data fusion. Our approach extends the classical ray casting algorithm by combining it with the Data Intermixing technique (Section 2.2) in a straightforward manner. A visualization technique based on this approach was implemented as a C++ class of the framework described in [13].

Two synthetic data volumes with $100 \times 100 \times 100$ voxels and intensity in $[0, 255]$ are used to illustrate our examples. We call them "sphere"(Figure 2a) and "cube"(Figure 2b). The sphere has three different layers (intensities) and a thin square inside it. The cube has several layers with different intensities. Anatomic data (e.g. CT or MRI) can be represented by the sphere, while the cube can be used to simulate a functional volume.

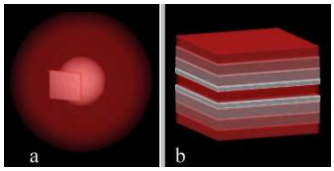


Figure 2: Synthetic volumes: "sphere"(a) and "cube"(b).

### 3.1 Integration of Multimodal Data

The algorithm is based on direct volume rendering with an extension to integrate multimodal data at the accumulation level of the rendering pipeline (Section 2.2). We adopted the recursive approach introduced by Levoy [9] to implement a front-to-back composition.

For the integrated visualization of two previously registered volumes, $V_1$ and $V_2$, two rays are cast, one for each volume. Consequently, two volume samples $V_1(\mathbf{x}_{s_i})$ and $V_2(\mathbf{x}_{s_i})$ are simultaneously taken into account for color and opacity calculation at each sampling position $s_i$. Since we are dealing with different volumes, $\mathbf{x}_{s_i}$ is mapped to $\mathbf{x1}_{s_i}$ and $\mathbf{x2}_{s_i}$ for sampling volumes $V_1$ and $V_2$, respectively. We assume that $V_1$ is used to determine the opacity value, possibly representing anatomy, and $V_2$ is used to obtain the color value, possibly representing function. In this way we perform the data intermixing in the accumulation level [2].

Then, the Equations 4 and 5 are rewritten below to emphasize that the color and opacity transfer functions are based on data from different volumes:

$$C(\mathbf{x}_{s_i}) = \begin{cases} f_C(\mathbf{x2}_{s_i}) f_\alpha(\mathbf{x1}_{s_i}), & i=0 \\ C(\mathbf{x}_{s_{i-1}}) + f_C(\mathbf{x2}_{s_i}) f_\alpha(\mathbf{x1}_{s_i})(1-\alpha(\mathbf{x}_{s_{i-1}})), & i>0 \end{cases} \quad (6)$$

$$\alpha(\mathbf{x}_{s_i}) = \begin{cases} f_\alpha(\mathbf{x1}_{s_i}), & i=0 \\ \alpha(\mathbf{x}_{s_{i-1}}) + f_\alpha(\mathbf{x1}_{s_i})(1-\alpha(\mathbf{x}_{s_{i-1}})), & i>0 \end{cases} \quad (7)$$

Figure 3 illustrates the integrated visualization of the test volumes. In Figure 3a, the sphere corresponds to $V_1$ and determines opacity, representing anatomy, and the cube corresponds to $V_2$ and is used for color, representing function. In Figure 3b the roles of $V_1$ and $V_2$ are swapped, respectively determining color and opacity. Thus, the cube is colored according to the sphere (see left image in Figure 3b where a cutting plane was used). A linear opacity transfer function was used to specify opacity in both images. The color transfer functions were given by different color tables.

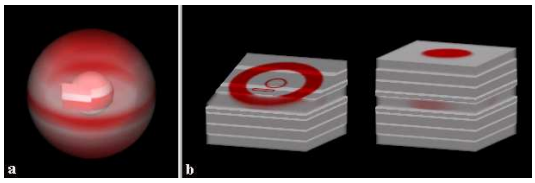


Figure 3: Integrated visualization of the synthetic volumes.

### 3.2 Visualization of Inner Structures in Multimodal Data

Besides transparency control, visualization of inner structures can be obtained by controlling the ROI, eliminating volume portions that are not relevant. This can be done in two different ways: by exclusion or by inclusion. In both cases, the cutting tools allow the user to specify the volume region that should be excluded (or included) in the processing. Below we describe a unified approach for cutting techniques in the context of the ray casting algorithm, presenting examples of how they can be used to visualize the interior of multimodal volume data.

#### 3.2.1 Unified Approach to Cutting Techniques

In Equation 2, the sampling interval along $\mathcal{R}_p$ is given by $[a, b]$ and it determines the voxels that in fact contribute to the calculation of pixel intensity. If no volume cutting tools are available, $a$ and $b$ correspond, respectively, to the intersection points of each ray with the volume data: $r_1$ is the position where it enters the volume data, and $r_2$ is the position where it leaves the volume ($a = r_1$ and $b = r_2$ - see Figure 4).

Cutting techniques can be seen as strategies to modify the sampling interval by determining $a$ and $b$ other than on

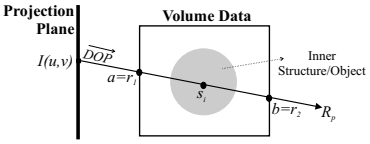


Figure 4: Sampling interval when no cutting tools are used.

the basis of the volume data boundaries. Figure 5 illustrates the use of two cutting planes, frontal and back, to determine the sampling interval by their intersection $c_1$ and $c_2$ with the ray ($a = c_1$ and $b = c_2$).

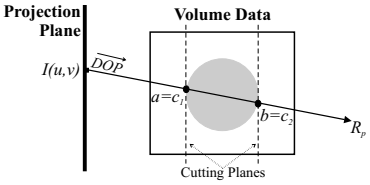


Figure 5: Cutting planes delimiting the sampling interval.

We define general cutting tools in terms of functions $f_a$ and $f_b$ that determine the values of $a$ and $b$:

$$a = f_a(U_a, \mathcal{R}_p, V) \quad (8)$$

$$b = f_b(U_b, \mathcal{R}_p, V) \quad (9)$$

where $U_a$ and $U_b$ correspond to parameters supplied interactively by the user to configure the cutting tool. Different cutting tools can be implemented depending on the choice of user parameters $U$ and the functions $f$.

The parameters $U$ typically correspond to geometrical elements (e.g. a plane or a cube) used to define the subset of voxels to be removed. In some techniques, $U$ can be an interval of voxel density values that belong to the ROI (e.g. range threshold).

The functions $f$ typically compute the intersection between the given object and the ray to determine the position of $a$ and $b$ for a given pixel. If no intersection exists, $r_1$ and $r_2$ are assumed respectively for $f_a$ and $f_b$.

### 3.2.2 Cutting Tools

We initially consider the situation where $a = b$. This is the case of using a cutting plane or surface to obtain a *single section*, planar or non-planar, from the volume data [15, 19]. Thus, just one sample $s_i$ is used to determine each pixel color $I(\mathbf{u})$, resulting in a 2-D visualization of a 1-voxel thick slice of the volume data. The function $f = f_a = f_b$ determines the sampling location by calculating the intersection between $\mathcal{R}_p$ and the geometry information given by user via the configuration of $U = U_a = U_b$.

We now move on to more flexible tools where $a \neq b$, which involve the computation of several sample points for each $\mathcal{R}_p$, resulting, in fact, in a sub-volume. In this case, additional cutting planes may be inserted into the object space (volume data) [10].

A first possibility is to specify a *front cutting plane* to remove all voxels between the observer and this plane [21]. In this case, $U_a$ specifies the plane parameters, in such a way that $f_a = c_1$, and $f_b = r_2$. The user could also specify a *back cutting plane* by configuring the plane parameters given by $U_b$, removing all voxels located thereafter ($f_b = c_2$). Alternatively to planes, it is possible to use one or two *non-planar cutting objects*, e.g. quadric surfaces [19].

The combined use of cutting planes and the integrated visualization of multimodal data (Section 3.1) is shown in Figure 6. In this case, volume $V_1$ determines opacity, and volume $V_2$ specifies color. Note that it is possible to see the inner structures with the colors by identifying their functional values. Figure 6a shows the result of a cutting by exclusion visualization ($f_a = c_1$ and $f_b = r_2$). Figure 6b and Figure 6c are, respectively, the results of cutting by exclusion and by inclusion, both with $f_a = c_1$ and $f_b = c_2$.

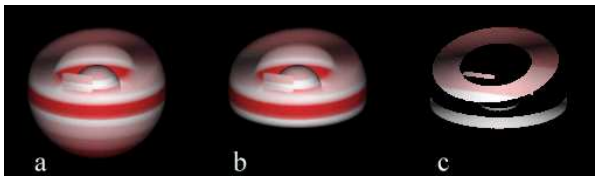


Figure 6: Example of one (a) and two (b) cutting planes by exclusion, and two cutting planes by inclusion (c), used to select the voxels that are under the planes.

A *convex volumetric cutting object* can also be used to remove (or select) all the voxels located in its intersection with the volume, i.e. the voxels that are *inside* the given object determined by $U = U_a = U_b$. In the examples below, we consider a cube as the cutting volume, where $U$ describes the cube dimensions and location.

In the case of cutting by exclusion, the function $f_a$ computes the intersection between the cube and $\mathcal{R}_p$. If no intersection exists, $f_a = r_1$; if there is intersection, $f_a$ assumes the value $c_2$, where the ray leaves the cutting cube. In both cases, $f_b = r_2$. However, if the cutting object is in the middle of the volume data, a special procedure is needed for handling two sampling intervals: $f_a = r_1$ and $f_b = c_1$, corresponding to samples along the ray that are closer to the viewer; $f_a = c_2$ and $f_b = r_2$, corresponding to samples after the cutting object, and far from the user.

In the case of cutting by inclusion, all voxels that are outside the cube are excluded from the computation of the pixel value. The functions $f_a$ and $f_b$ are therefore coupled to determine the first ($c_1$) and the second ($c_2$) intersection points between the cutting object and $\mathcal{R}_p$. An experiment

using a cube as a volumetric object to perform cutting by inclusion and by exclusion is presented in Figure 7.

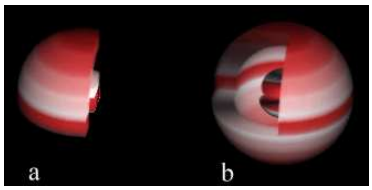


Figure 7: Cutting multimodal data by inclusion and by exclusion with a cube.

In all cutting techniques mentioned so far, $U_a$ and $U_b$ correspond to geometric information. Instead of geometrical elements, voxel properties can also be used to define the subset of voxels that has to be cut or selected. The Confocal Volume Rendering technique (CVR) [14] explores this idea by combining voxel properties with geometric information to specify the visible portion of the volume. This algorithm is used to control a "focal" region using three parameters: (a) See-through band (S-Band), where the user defines a voxel property (e.g. density) that characterizes an structure and a distance (e.g. mm) from the given structure to step-over; (b) Transition band (T-Band), which indicates exactly how much (in terms of depth, e.g. mm) of the structure the user desires to see from the S-Band; (c) Enhance, which configures the opacity scaling function within the selected portion of the volume, allowing the user to enhance particular structures within the ROI.

The CVR technique was also modeled using our approach. The S-Band corresponds to $U_a$ and is determined by the voxel property $v$ and a constant distance $d_1$. The function $f_a$ is defined as:

$$f_a = first(\mathcal{R}_p, v) + d_1 \tag{10}$$

where $first$ is a function that gives the first sample position $s_i$ such that $V(x_{s_i}) = v$.

The T-Band parameter corresponds to $U_b$, which is used to specify $b$ in terms of $a$ and a constant distance $d_2$:

$$f_b = a + d_2 \tag{11}$$

In the CVR technique, therefore, the sampling interval $[a, b]$ is determined by a mixture of voxel properties (in $f_a$) and geometry (distances in $f_a$ and $f_b$).

Examples are presented in Figure 8. The lines in the axial and sagittal planes identify the ROI defined by the S-Band and T-Band parameters interactively specified over volume $V_1$. Two different ROI are selected resulting in images that show different features. We can see that the final images reveal structures that are different from those obtained with the geometrical cutting techniques shown in Figures 6 and 7.

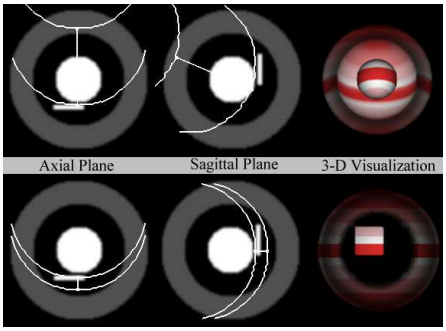


Figure 8: Multimodal data rendered with CVR technique.

This shows that applying cutting techniques based on voxel properties in the visualization of multimodal data provides more possibilities for selecting a ROI. For example, using a front cutting plane, $a$ and $b$ will be the same for both volumes, while in the CVR technique $a$ can be set based on a value $v$ (Equation 10) taken from volume $V1$ or volume $V2$, providing different results.

## 4 Results and Discussion

The integration of multimodal data into a single image requires enhanced visualization techniques that combine data from different nature providing a meaningful representation. For the visualization of inner structures, it is possible to use transfer functions, cutting and segmentation techniques, either as alternative approaches or combined ones. Interactive cutting operations are required for removing obstructing parts of the volume when the user wishes to see different parts of the interior of a data volume [4]. In Section 3 we presented a unified approach for implementing cutting tools in combination with multimodal data volumes, with examples based on synthetic volumes. In this section we present results obtained for the visualization of inner structures in medical images - see Figure 9.

Three datasets were used in the experiments. One volume was acquired by ftp from the University of North Carolina, which gives the anatomy and consists of 128 slices of $113 \times 128$ MRI images. The same volume was manually modified in order to simulate MRI functional data, which shows the activation area due to a hypothetical task performed by the patient. The third volume is the synthetic cube, adapted to match the resolution of the first volume.

The MRI data was integrated with the cube (Figure 9c) and with the simulated MRI functional data (Figures 9a, 9b and 9d). We use the basic formulation presented in Section 3.1, changing $a$ and $b$ values to simulate different cutting tools as described in Section 3.2. Functions $f_a$ and $f_b$ were specified by geometry only in Figures 9a, 9b and 9c, and in combination with voxels properties in Figure 9d, with the goal of revealing the brain. In this case, we use

the anatomical volume to define CVR parameters, and the other one to process the color, extending CVR to deal with multimodal data.

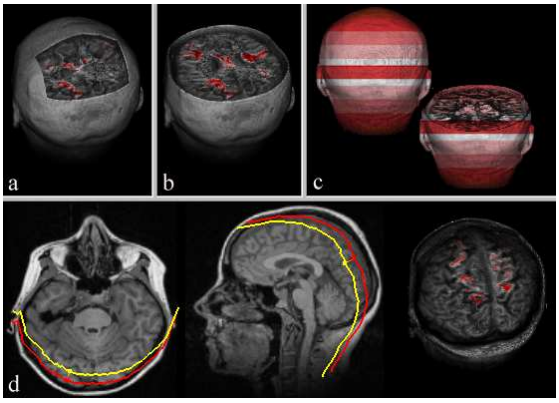


Figure 9: Multimodal medical image visualization.

Figure 9 presents an example where anatomical data (MRI) is used to eliminate (or cut) functional data outside the ROI (in this case, the skull). This example illustrates the power of the proposed approach, which combines the advantages of the CVR [14] (flexible cutting tools) with those of the Data Intermixing [2] techniques (simultaneous visualization of multimodal data). Its main advantages are: (1) allows the analyses of two different information, functional and anatomical, and provides the ability to eliminate regions that do not contribute for the diagnosis, using geometrical objects or voxel information; (2) does not rely on a segmentation step and reduces the image generation time through the reduction of samples' calculation; (3) the unified approach (Section 3.2.1) allows a better understanding of the current cutting tools described in the literature as well as the identification of unexplored alternatives.

The extension of the unified approach to deal with multimodal data volumes enables the combination of voxel properties of more than one volume to be used as parameters for the cutting tools. Such tools are needed for the visualization of un-segmented volume data, but often they are described in a simple way, referred only as slicing tools like in a recent review [1]. However, cutting can be performed based not only on geometry but also on voxel properties, in a more natural and intuitive way to interact with the data, especially when visualizing multiple image modalities.

Other authors have addressed the visualization of multimodal data, some of them including conventional cutting tools as planar sections and volumetric cutting objects. Those cutting techniques can be easily modeled using our unified approach for cutting tools.

The ANALYZE$^{TM}$ system [18, 19] developed at the Mayo Clinic, provides an integrated set of display, manipulation and measurement tools for detailed investigation and evaluation of three-dimensional biomedical images. Some of the provided exploration tools are oblique and curved sectioning and sub-volume selection, which are also available for multimodal data. These tools can be easily modeled using our approach just setting $a$ and $b$ parameters. However, no cutting tools based on voxel property are present.

The 3Dbench software package [16] contains general routines to manipulate and visualize volumes. It provides an interactive slicer and volume renderer for the visualization of several volumes: unprocessed, segmented, multimodal, multi-channel and any combination of them. However, it does not provide cutting tools other than slicing.

Zuiderveld [23] developed a software architecture for the integrated visualization of multimodal volumetric data sets, including a technique to display vascular information, called Closest Vessel Projection (CVP). CVP, which is performed just with one volume, consists in terminating the ray casting when the ray exits the first vessel it encounters. In our general scheme, this tool could be implemented by setting $f_a$ to the first intersection point with the volume ($r_1$), and $f_b$ is specified in terms of the voxel property, in this case the density of the vessel boundary. So, our conceptual framework allows the development of a set of tools to visualize inner structures where CVP can be included.

## 5 Conclusions

This paper provides results in two important aspects of volume visualization: cutting tools for volume inspection tasks and integrated visualization of multimodal data. The use of cutting tools as a mechanism for seeing through multimodal volume data has not been explicitly addressed in the literature. The general approach for volume cutting tools facilitates understanding the weaknesses and strengths of each one and their applicability to interactive volume exploration. The combination of general cutting techniques and the Data Intermixing [2] technique for visualization of inner structures in multimodal data volumes provides new ways for exploring these data sets. For example, the unified approach allowed us to easily integrate CVR [14] with multimodal data visualization, opening new possibilities for case studies with this technique.

In the context of medical imaging, the presented approach provides powerful tools to explore the integrated volume data obtained from different modalities, sometimes of a complementary nature, revealing additional diagnostic information. The generic formulation for cutting tools allowed us, for example, to exhibit the integrated volumes using even a volume from an atlas to set $a$ and $b$ parameters for cutting. It is easy to understand the behavior of each volume cutting tool, and how they can be implemented for prototyping purposes. Moreover, this study revealed that several new tools could be created from the combination of

multiple data volumes and cutting strategies found in existing methods. The exploration of this new space of cutting tools combined with the visualization of multimodal data is our current topic of investigation.

We are interested in the applicability of such tools for inspection of medical volume data, which is likely to require different or even combined techniques for different tasks. The development of an interactive interface and a case study using real medical images, including the integration of opacity transfer functions in our unified approach, is the next topic of investigation.

**Acknowledgements**

This research was partially supported by CAPES, CNPq and FAPERGS.

**References**

[1] K. Brodlie and J. Wood. Recent Advances in Volume Visualization, *Computer Graphics Forum*, v. 20, n. 2, 2001, p. 125–148.

[2] W. Cai and G. Sakas. Data Intermixing and Multi-Volume Rendering. *Computer Graphics Forum*, Cambridge, v. 18, n. 3, 1999, p. C359–C368.

[3] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Reading, MA, Addison-Wesley, 1993, 716 p.

[4] M.W. Jones and M. Chen. Fast Cutting Operations on Three Dimensional Volume Datasets. In: M. Goebel et al. (eds). *Visualization in Scientific Computing*. Berlin, Springer-Verlag, 1995, p. 1–8.

[5] A.E. Kaufman. *Volume Visualization*. Los Alamitos, CA, IEEE Computer Society Press, 1991.

[6] G. Kindlmann and J.W. Durkin. Semi-Automatic Generation of Transfer Functions for Direct Volume Rendering. In: Symposium on Volume Visualization, 1998. *Proceedings...* p. 79–86.

[7] D.N. Levin, X. Hu, K.K. Tan et al. The Brain: Integrated Three-Dimensional Display of MRI and PET Images. *Radiology*, v. 172, n. 3, 1989, p. 783–789.

[8] M. Levoy. Volume Rendering - Display of Surfaces from Volume Data. *IEEE Computer Graphics & Applications*, v. 8, n. 3, 1988, p. 29–37.

[9] M. Levoy. Efficient Ray-Tracing of Volume Data. *ACM Transactions on Graphics*, v. 9, n. 3, 1990, p. 245–261.

[10] B. Lichtenbelt, R. Crane and S. Naqvi. *Introduction to Volume Rendering*. Upper Saddle River, NJ, Prentice Hall, 1998.

[11] G.Q. Maguire Jr., M.E. Noz, H. Rusinek et al. Graphics Applied to Medical Image Registration. *IEEE Computer Graphics & Applications*, March, 1991, p. 20–28.

[12] J.B.A. Maintz and M.A. Viergever. Survey of Medical Image Registration. *Medical Image Analysis*, v. 2, n. 1, 1998, p. 1–36.

[13] I.H. Manssour, S.S. Furuie, L.P. Nedel and C.M.D.S. Freitas. A Framework to Visualize and Interact with Multimodal Medical Images. In: K. Mueller and A. Kaufman (eds). *Volume Graphics 2001*. Wien, Austria, Springer-Verlag, 2001, p. 385–398.

[14] R. Mullick, N. Bryan and J. Butman. Confocal Volume Rendering: Fast Segmentation-Free Visualization of Internal Structures. In: SPIE'00, San Diego, California, 2000. *Proceedings...* p. 144–154.

[15] G.M. Nielson and B. Hamann. Techniques for the Interactive Visualization of Volumetric Data. In: IEEE Conference on Visualization, 1990. *Proceedings...* p. 45–50.

[16] H.J. Noordmans. *Interactive Analysis of 3D Microscope Images*. Netherlands, Universiteit Utrecht, 1997, Ph.D. Thesis.

[17] H. Pfister, B. Lorensen, C. Bajaj et al. The Transfer Function Bake-Off. *IEEE Computer Graphics & Applications*, v. 21, n. 3, May/June 2001. p. 16–22.

[18] R.A. Robb. Visualization Methods for Analysis of Multimodality Images. In: R.W. Thatcher et al. (eds). *Functional Neuroimaging: Technical Foundations*. San Diego, CA, Academic Press, 1994.

[19] R.A. Robb. *Three-Dimensional Biomedical Imaging: principles and practice*. New York, NY, John Wiley & Sons, 1998.

[20] R. Stokking. *Integrated Visualization of Functional and Anatomical Brain Images*. Netherlands, Universiteit Utrecht, 1998, Ph.D. Thesis.

[21] R. Westermann, T. Ertl. Efficiently using Graphics Hardware in Volume Rendering Applications. In: SIGGRAPH 1998, *Proceedings...* p. 169–177.

[22] S. Yi Yen, S. Napel and G.D. Rubin. Fast Sliding Thin Slab Volume Visualization. In: Symposium on Volume Visualization, 1996. *Proceedings...* p. 79–86.

[23] K.J. Zuiderveld. *Visualization of Multimodality Medical Volume Data using Object-Oriented Methods*. Netherlands, Universiteit Utrecht, 1995, Ph.D. Thesis.