AnatomyBrowser: A Framework for Integration of Medical Information

P. Golland[*], R. Kikinis[†], C. Umans[†],
M. Halle[†], M.E. Shenton[†], J.A. Richolt[†]

Abstract

In this paper we present AnatomyBrowser, a framework for integration of images and textual information in medical applications. AnatomyBrowser allows the user to combine 3D surface models of anatomical structures, their cross-sectional slices, and the text available on the structures, while providing a rich set of cross-referencing and annotation capabilities. The 3D models of the structures are generated fully automatically from the segmented slices. The software is platform independent, yet is capable of utilizing available graphics resources. Possible applications include interactive anatomy atlases, image guided surgery and model based segmentation. The program is available on-line at **http://www.ai.mit.edu/projects/anatomy_browser**.

1 Introduction

With recent developments in MRI technology it has become possible to obtain high quality medical images that are extremely useful for clinical studies, surgical planning and other applications. This has spurred rapid development of a family of information extraction and analysis algorithms using medical imagery. These include segmentation and registration, shape and deformation modeling, etc. Importance of visualization of the results, especially 3D data, became apparent as well.

This paper presents AnatomyBrowser, a visualization and integration framework for medical applications. It provides visualization capabilities for slice sets and 3D surface models of anatomical structures, as well as a useful set of cross-referencing and annotation features. It can also be used as an aid tool for the model driven segmentation.

[*]Corresponding author. Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139

[†]Surgical Planning Laboratory, Brigham and Women's Hospital, Boston, MA 02115

The system is written as a Java applet, which makes it platform independent. It can be readily run using any web browser that supports Java. In fact, the system is available on-line [1].

Although no special hardware or software is required to run AnatomyBrowser in its basic setup, it is capable of utilizing specialized graphics resources (hardware/software). For this purpose, an additional 3D viewer was created that generates dynamic renderings of the 3D surface models using rendering capabilities of the graphics software available.

This paper is organized as follows. The next section reviews the differences between AnatomyBrowser and other systems available in the field. Then AnatomyBrowser is discussed in detail and its interface is described. After that, uses of AnatomyBrowser in various application areas are reported. It is followed by a discussion of possible extensions to the system, which include merging the dynamic and the static versions of the 3D rendering components of the system.

2 Related Work

Interactive anatomy atlases are one important example of possible applications for the visualization and integration system we propose. In fact, it was an original inspiration for this work.

The work on digital atlases has been pioneered by Höhne *et al.* [4]. A digital brain atlas integrating several image modalities with the information on anatomy hierarchy was proposed in [6]. The goal of the work presented in this paper was to develop a system for visualization of segmentation results (both in 2D and 3D) and their integration with original scans and text information available on the structures. We do not restrict the application of the system to atlas generation. It can be used on any data set, including clinical cases.

In addition to anatomy studies, digital atlases are used for model driven segmentation. The atlas data is treated by the algorithm as a reference template, to which the input data set is registered. Several registration algorithms (using both rigid and elastic matching) have been developed by different groups [2, 3, 10]. Another approach is to extract anatomical knowledge from the atlas and use it to build a prior model of the input data [5]. In both cases, AnatomyBrowser can serve as an aid tool in the model based segmentation 'loop': segmented images are used to accumulate knowledge of anatomy, which is used, in turn, for segmentation of the new data sets.

In the visualization field, there are several packages available for rendering of 3D surface models (Inventor by SGI, VTK by GE, and many others), but to the authors' best knowledge, AnatomyBrowser is the only integration environment that provides extensive referencing and annotation capabilities. This is extremely important in clinical applications, when a user wants to establish a clear correspondence between all types of pictorial information available. Another significant difference between AnatomyBrowser and other visualization

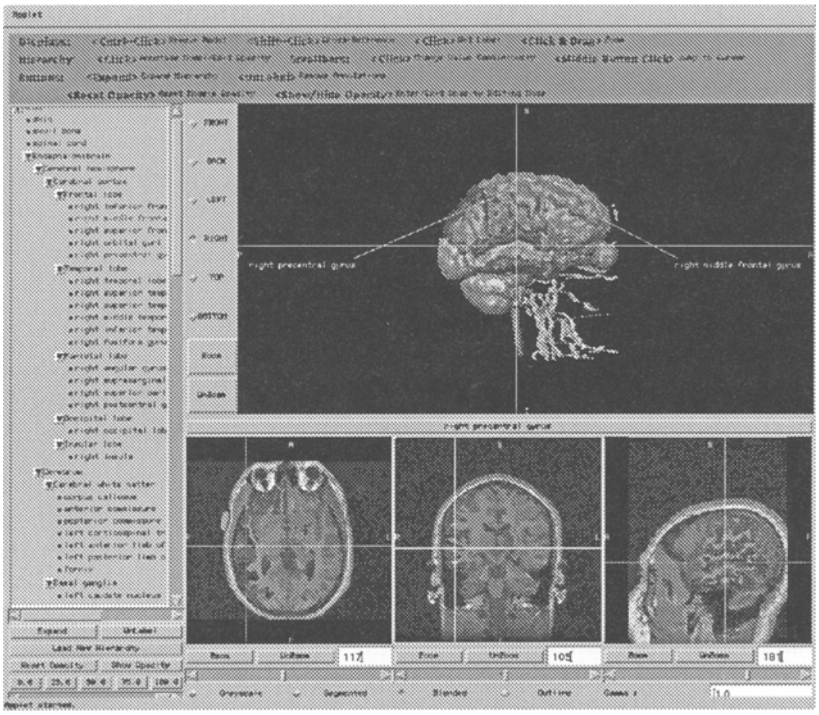


Figure 1: An example of AnatomyBrowser for a brain data set. The cross-hair positions correspond to the same 3D point. Some of the structures are annotated on the 3D display. The sagittal slice is slightly zoomed in.

packages is its portability. It can be a great advantage, considering the target applications and the user community.

3 AnatomyBrowser

In this section the user interface of AnatomyBrowser is described, and the design considerations are introduced and discussed.

AnatomyBrowser integrates three main types of information: the slice sets, the 3D models and the text information on the structures (structure names and general text notes on some structures). Accordingly, its interface consists of three main components, namely, the slice displays, the 3D display and the hierarchy panel (Fig.1). In addition to visualization capabilities, AnatomyBrowser provides cross-referencing among all types of displayed information. The cross-referencing among the images is different in its nature from the correspondence between image and text information. While the former can be established au-

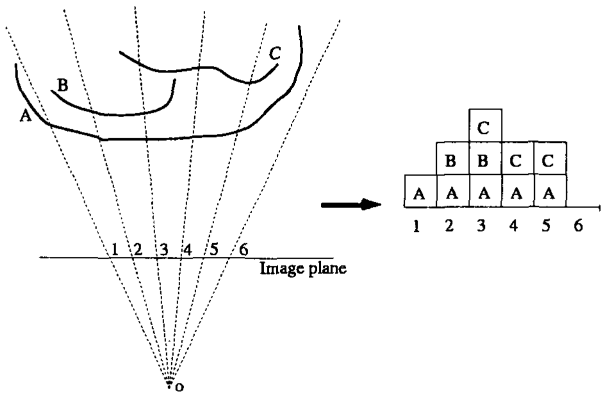


Figure 2: A multi-layer image. The 3D scene consists of three surfaces (A,B and C). The view rays and the stacks in the multi-layer image are shown for six different pixels.

tomatically, the latter requires some user input.

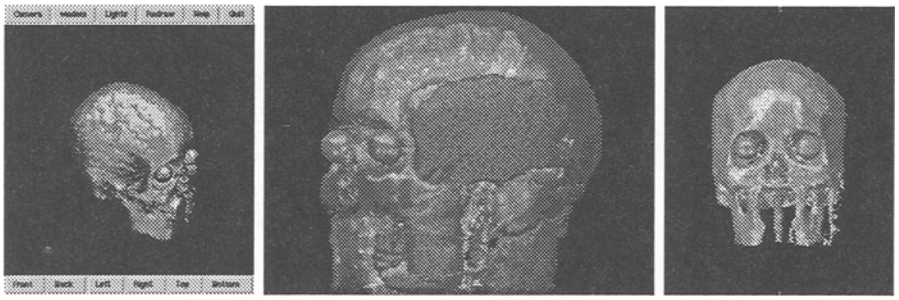
The sections below discuss each interface component, the information it provides to the user and its interaction with other components of the system.

3.1 3D Model Generation

We used the traditional approach of polygon (triangle) meshes for representation of 3D surface models. We use the Marching Cubes algorithm [7] to construct a triangle approximation of a surface from a set of segmented slices. We assume the segmentation to be available. Section 4 discusses connections between AnatomyBrowser and segmentation algorithms used for the medical applications.

To build a surface model, the algorithm first considers each border voxel of the segmented structure independently of its neighbors. In this phase, a planar patch of a surface is built for every border voxel. Next, patches from neighboring voxels are merged to form a single mesh that approximates the surface of the structure. In order to decrease the number of polygons (triangles) in the representation and improve the surface appearance, some amount of decimation and smoothing is performed on the mesh under constraints on the accuracy of the representation.

The model construction process uses a coordinate system naturally defined by the input slice set, and therefore the models can be straightforwardly registered to the pixels in the original slices. This greatly simplifies implementation of cross-referencing between the 3D models and the cross-sectional views.



(a) Dynamic viewer. (b) An oblique view. (c) Front view.

Figure 3: Different examples of the 3D display. (a) The dynamic viewer allows one to render any view of the models. (b,c) Additional examples of the static views. Skin was made fully transparent, skull was made partially transparent, and in (c) muscles were removed.

3.2 3D Model Visualization

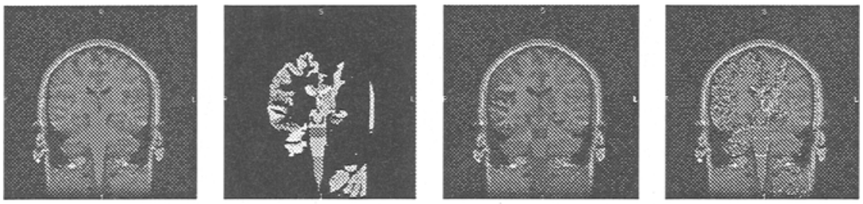
The viewer program for visualization of 3D surface models was created using the Visualization Toolkit (VTK) [8], a software package that uses available GL libraries (e.g. openGl, xGl, Mesa, etc.) to take advantage of graphics resources of a particular machine. Any other visualization library that provides dynamic rendering (such as Inventor by SGI) could be used for implementation of the viewer.

The viewer provides controls for the scene's global parameters, such as the observer's position relative to the scene and the light locations and intensities, as well as the surface properties of the models (color, transparency and reflectivity). Fig.3a shows the viewer interface with the surface models for the brain data set.

Since the goal of this work was to produce an easily portable, platform independent system, a static version of the viewer was implemented as well. The static viewer does not require any special graphics hardware or software, but it is restricted to a set of static views rather than allowing the user to choose a view dynamically. In the current implementation, a user specifies several views of the 3D models, which are rendered off-line and then displayed by AnatomyBrowser. The user still has control over the surface properties of the model (color and transparency), but the properties that would require re-rendering the entire scene (for example, lighting) are kept constant. The AnatomyBrowser interface shown in Fig.1 uses the static version of the viewer.

We use so-called *multi-layer images* [9] to save the information necessary for rendering and depth recovery for the static viewer. The idea of the multi-layer image is the following. For each pixel in the image, a view ray is constructed (Fig.2).

This ray might intersect several surfaces in the scene. For each surface that



(a) Greyscale (b) Segmented (c) Blended (d) Outline

Figure 4: Slice display. The same coronal slice from the brain data set is displayed using four different ways of combining greyscale and segmented images.

is intersected by the view ray, a tuple of the surface index, the intensity and the depth of the intersection point is saved. Thus each pixel in the multi-layer image is represented by a (possibly empty) stack of <index,intensity,depth> tuples, which are used later for rendering and establishing relative location of the structures.

Note that in a proper representation the multi-layer images are almost as easily compressed as the segmented images. Since most of the models are smooth, each of the tuple channels contains (hopefully large) uniform patches. If the channels are compressed separately, high compression rates can be achieved.

Since the pixel value is computed at rendering time by composing the surface points that project onto that pixel, surface transparency and color can be easily handled by the static version of the viewer. Fig.3c shows the brain data set with the skin made fully transparent and the skull being only partially transparent.

3.3 Cross-Sectional Slices

Axial, coronal and sagittal[1] cross-sections are displayed on the three slice displays. The cross-sectional slices are computed from the original scan by resampling it along the three orthogonal directions.

Each slice display has a set of original greyscale images and a set of segmented slices associated with it. AnatomyBrowser provides four different ways to view the slices. The user can choose to view just the greyscale images, or just the segmented set, or blend them together, or view the greyscale images with the outlines of the segmentation. Fig.4 demonstrates all four choices for one slice from the brain data set.

3.4 Cross-Reference Among the Displays

An important capability provided by AnatomyBrowser is cross-referencing among the displays. If a user clicks on one of the display areas (<*Shift-Click*>), a

[1]These are the standard orthogonal cross-sections used in the medical community. They correspond to bottom-to-top, front-to-back, left-to-right cross-sections respectively.

cross-hair appears on all four displays at locations corresponding to the same 3D point.

Even though all four displays exhibit very similar behavior, the cross-referencing definition and implementation must be different for the 3D and the slice displays. To understand why this is so, let's consider a mapping from the display coordinates to the physical volume defined by the input set of slices.

First consider a slice display. Any pixel in the slice display is fully defined by three numbers: the slice index and the pixel coordinates in the slice. These three numbers are also sufficient to compute the 3D coordinates of the corresponding point in the volume defined by the slice set. Therefore, the mapping between pixels in the slice set and physical points in the 3D volume is one-to-one and can be readily computed from the physical dimensions of the voxel.

This is not true for the 3D display. Each pixel in the 3D display is described by only two coordinates. The user has no means to specify nor to obtain depth values from the 3D display, and therefore the depth dimension is lost for the images on the 3D display. The natural one-to-one correspondence between the pixels of the 3D display and the entities in the 3D space is the one from pixels to view rays and *vice versa*. This construction provides us with a useful interpretation of the cross-referencing for the 3D display.

If a cross-reference is initiated from the slice display, the corresponding 3D point can be uniquely computed. For the 3D display, we place the cross-hair on the pixel corresponding to the selected 3D point, but we might not see the point itself on the display. Instead, we will see a surface that occludes the selected 3D point. The cross-hair on the 3D display should be interpreted as a view ray along which the selected 3D point lies.

In a similar way, by clicking on the 3D display, a user may specify a view ray rather than a single 3D point. We use the depth information available to AnatomyBrowser to disambiguate this problem. We assume that the point of interest lies on the intersection of the view ray with the closest visible surface. This decision is justified by a natural notion of pointing[2]: when interpreting one's pointing, an observer follows the imaginary ray defined by the pointer until it meets an object. This object is assumed to be the answer.

3.5 Model Hierarchy

Textual information is a third important component of AnatomyBrowser. This includes names of anatomical structures and text notes associated with some of the structures.

AnatomyBrowser uses a directed acyclic graph (DAG) to model the anatomical hierarchy. The text file that defines the hierarchy is essentially a list of internal nodes that represent groups of structures, and their children, each of

[2]One can argue that this is a very simplistic interpretation of the human interaction by pointing, and the authors would fully agree with that.

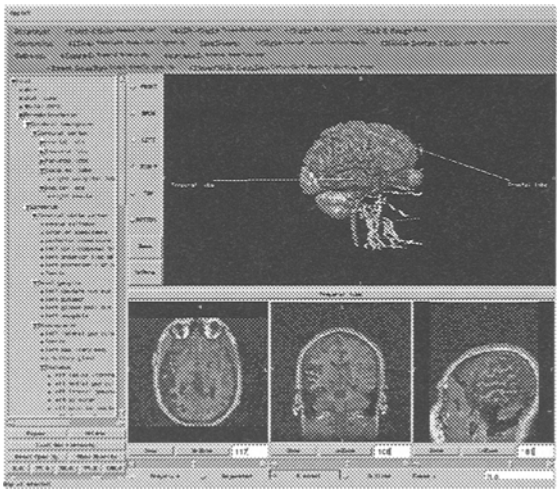


Figure 5: AnatomyBrowser for the brain data set with the frontal lobe collapsed into one node. Compare to Fig.1 for the differences in the hierarchy panel, the colors of the structures and annotation labels.

which can be either another group or a single structure. The hierarchy file is parsed by AnatomyBrowser and displayed in the hierarchy panel.

We use a tree model to display the hierarchy, because it is more suitable than a DAG for visualization purposes. If a particular group (or structure) is shared by several parents, several instances of the corresponding subgraph (or leaf) are generated and displayed under each of the parents. Since the internal representation uses a DAG, and all the instances of a group or a structure point back to the same element in the representation, the element properties can be changed using any of the instances. This implies that the changes will affect all the instances. For example, after a certain instance of a structure in the hierarchy panel was used to change the structure transparency, all the instances of this structure will report the new value for transparency.

A hierarchy tree (graph) is a natural, systematic way to classify and present anatomical structures. It also allows the user to set a desirable level of details for the 3D models and segmented slices. Any subtree in the hierarchy panel can be collapsed into a single node (leaf). This causes all the structures in the sub-tree to be re-rendered using the color of the new collapsed node and to be treated as one structure that point forward (Fig.5).

This feature can be very useful for studying anatomy with the help of AnatomyBrowser. A user can start with a very coarse hierarchy tree and expand the group nodes gradually, rather than try to learn the whole tree at once.

3.6 Integrating Text and Images

AnatomyBrowser also provides annotation capabilities, which can be viewed as cross-referencing between text and images. Annotation of the structures can be invoked from any of the main components of the AnatomyBrowser interface.

Implementation of the annotation using the hierarchy panel is fairly straightforward. Clicking on a node name in the hierarchy tree causes this name to appear on the title bar below the 3D display. If this structure – which can be a group that was collapsed into a single structure – can be seen on the 3D display, it is also labeled with a pointer and the structure name (Fig. 5,6).

Using images for annotation requires additional effort. We use segmented slices for the slice displays and multi-layer images for the 3D display to determine for each pixel what structure, if any, contains this pixel. When a user clicks on any of the displays, the corresponding structure is located in the hierarchy graph and used for annotation.

As was mentioned before, AnatomyBrowser allows text notes to be attached to a particular structure. We are using a title bar with the name of the structure as a 'hypertext' link to a collection of text notes. Clicking on it brings up a text window with all the text information available on the structure.

It is worth mentioning that both annotation and text documentation can be used for all leaves in the hierarchy panel, whether they represent real structures or groups collapsed into a single node.

4 Applications

AnatomyBrowser provides a framework for visualization of medical image information and integrating it with text available on the structures in the images. Anatomy studies are the most obvious application for such a system. AnatomyBrowser can effectively replace a conventional anatomical atlas book, as it provides at least as much information as the book. Students can see anatomical structures, view an MR or CT scan of those structures, read attached description and study the hierarchy of the structures. Visualization and full cross-referencing make AnatomyBrowser a very useful interactive teaching tool.

We have generated several atlases using AnatomyBrowser. They include brain (Fig. 1), knee, abdominal and inner ear data sets. They are available on-line [1].

The atlas can also be used for model driven segmentation. We can view the process of model based segmentation as a loop, in which the results of the segmentation of the previous scans are used in segmentation of the new data sets. And the digital atlas is an intermediate step in this loop, where the knowledge about anatomy is accumulated in a particular representation, visualized for the user, and possibly processed for the future use by the segmentation algorithm. In a way, its purpose is to 'close the loop' from one stage of the segmentation

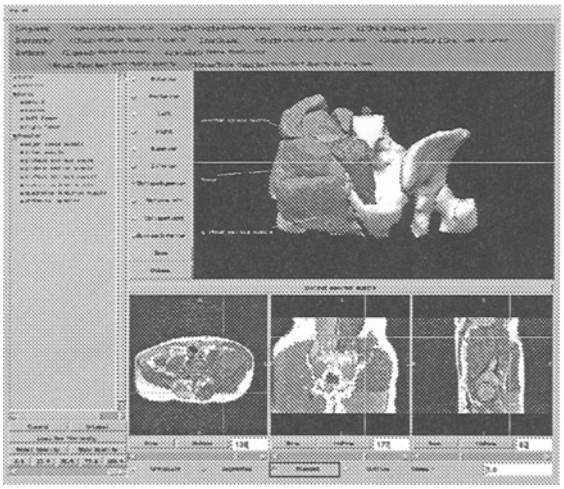


Figure 6: Clinical case example. AnatomyBrowser for the pelvic tumor data set demonstrating cross-reference and label correspondence.

to the next one, which uses the result of the first stage.

We want to point out that there is a significant difference between AnatomyBrowser and a standard anatomy atlas, and it is in the data. An atlas is usually one data set – real or phantom – that is used as a 'normal' reference. AnatomyBrowser, however, is a software package that takes images and text as input and produces a user-friendly interface for viewing and cross-referencing all the pieces of data. It can be used on any data set, and not necessarily on the reference case that would be used for an atlas. This naturally brings up another application for AnatomyBrowser. It can be used as a visualization and annotation tool for clinical cases, when the analysis has to be performed on a per-case basis.

We tested AnatomyBrowser on several tumor cases in different anatomical areas: pelvis (Fig. 6), sacrum, brain, and kidney. Again, all of the cases are available on-line [1]. The tool was tested for both surgical planning and image guided surgery. During the planning stage, the surgeons used AnatomyBrowser to familiarize themselves with the 3D structure of the tumor and structures around it and to plan the directions of the cuts. Then AnatomyBrowser, with the set of views specified by the surgeon, was used as a reference tool during the surgery.

We would like to mention that in two of those cases AnatomyBrowser allowed the surgeons to significantly improve their performance. In case of the sacral tumor, the visualization system made it obvious that certain vertebra were covered by the tumor on the right side, but not on the left side of the sacrum, which was of a great help in planning the surgery. For the pelvic tumor case, the initial plan made by the surgeon, was changed after the analysis of the 3D models in AnatomyBrowser had demonstrated that the originally planned cut would not be feasible.

The surgeons' response to the system was extremely positive. It provided them with much greater visualization capabilities than they had before, while establishing clear correspondences between the 3D models and the more commonly used cross-sectional slice sets.

5 Extensions

In addition to further testing of AnatomyBrowser for clinical use, we are working on merging the dynamic and the static versions of the 3D viewer under the same framework.

In the current implementation, the dynamic viewer is a separate program that requires various pieces of software to be available in the system. It is, therefore, not easily portable. On the other hand, the static version of the viewer is an integral part of the Java applet that implements the AnatomyBrowser interface. It can be viewed using any web browser that supports Java.

We are currently working on an implementation of the dynamic viewer in Java using Java3D, a package that provides an interface between Java and the GL libraries. The dynamic version of AnatomyBrowser will still depend on the graphics hardware and GL support, but this will eliminate its dependency on intermediate 'layers' of software (VTK, tcl/tk) used for visualization of the 3D models. The users with graphics support will be able to run AnatomyBrowser with no extra effort, while the users of the low-end machines with no graphics capabilities will still use the static version of the system.

6 Conclusions

We have presented a framework for generating and combining different modalities of medical imaging data. 3D models generated from the segmented slices provide the user with useful visualization capabilities, while the full cross-reference among all the sources of information allows flexibility in viewing modes with the possibility of bringing the displays in correspondence.

The software is easily portable to various platforms and doesn't require special resources. However, if the fast graphics hardware is available, the system can take advantage of it in a straightforward way.

AnatomyBrowser has been used to generate an interactive atlas for several data sets. The digital atlas can be used as an interactive teaching tool, as well as for model based segmentation. Surgical planning and image guided surgery are another example of the applications that can greatly benefit from using AnatomyBrowser. Different applications proved certain subsets of features to be useful. While hierarchical representation of the textual information and annotation capabilities were very important for the teaching tool, 3D visualization combined with cross-referencing were valuable for the clinical applications. The system as a whole combined a set of capabilities rich enough for both purposes.

References

[1] AnatomyBrowser URL. http://www.ai.mit.edu/projects/anatomy_browser.

[2] R. Bajcsy and S. Kovacic. Multiresolution elastic matching. *Comp. Vision, Graphic and Image Processing*, 46:1–21, 1989.

[3] D.L. Collins, C.J. Holmes, T.M. Peters, and A.C. Evans. Automatic 3D model-based neuroanatomical segmentation. *Human Brain Mapping*, 3:190–208, 1995.

[4] K.H. Höhne *et al.* A 3D anatomical atlas based on a volume model. *IEEE Computer Graphics Applications*, 2:72–78, 1992.

[5] M. Kamber *et al.* Model-based 3D segmentation of multiple sclerosis lesions in dual-echo mri data. *SPIE Visualization in Biomed. Computing*, 1808:590–600, 1992.

[6] R. Kikinis *et al.* A digital brain atlas for surgical planning, model driven segmentation and teaching. *IEEE Transactions on Visualization and Computer Graphics*, 2(3), 1996.

[7] W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics*, 21:163–169, 1987.

[8] W. Schroeder, K. Martin, and B. Lorensen. *The visualization toolkit : an object-oriented approach to 3D graphics.* Prentice-Hall, NJ, 1996.

[9] C. Umans, M. Halle, and R. Kikinis. Multilayer images for interactive 3d visualization on the world wide web. Technical Report 51, Surgical Planning Lab, Brigham and Women's Hospital, Boston, MA, September, 1997.

[10] S. Warfield *et al.* Automatic identification of gray matter structures from mri to improve the segmentation of white matter lesions. In *Proceedings of Medical Robotics and Computer Assisted Surgery (MRCAS)*, pages 55–62, 1995.