Fast and accurate semiautomatic haptic segmentation of brain tumor in 3D MRI images

Masoud LATIFI-NAVID¹, Murat BİLEN², Erhan İlhan KONUKSEVEN¹⁺, Mustafa DOĞAN², Adnan ALTUN³

¹Department of Mechanical Engineering, Middle East Technical University, Ankara, Turkey
²Department of Electrical and Electronics Engineering, Başkent University, Bağlca Campus, Ankara Turkey
³Department of Neurosurgery, Faculty of Medicine, Ondokuz Mayıs University, Samsun, Turkey

Abstract: In this study, a novel virtual reality-based interactive method combined with the application of a graphical processing unit (GPU) is proposed for the semiautomatic segmentation of 3D magnetic resonance imaging (MRI) of the brain. The key point of our approach is to use haptic force feedback guidance for the selection of seed points in a bounded volume with similar intensity and gradient. For the automatic determination of a bounded volume of segmentation in real time, parallel computation on the GPU is used. Automatic segmentation is applied in this adjustable bounded spherical volume with a variable diameter, which is controlled according to the edge map acquired from the gradient map. The haptic force feedback is used in order to guide the user to remain in a volume, where the intensity and gradient change are under a defined threshold range. After each seed point selection, the segmentation algorithm works inside the bounded volume of the ball with an adjusted diameter. The proposed segmentation method based on force and visual feedback with the advantage of adjustable bounded volume is not only accurate and effective in narrow spaces near the boundaries of different layers, but also fast in large homogeneous spaces since the radius of the ball increases in such regions. Parallel programming on the GPU is used for computing gradient change in selected directions, which is needed for the self-adjustment of the sphere diameter. Gradient values are used for calculating the haptic force on the CPU in real time. In this study, two haptic devices are used, one for getting haptic force feedback and the other for camera guidance during 3D visualization. A comparison between manual segmentation of MRI by an expert surgeon and the proposed segmentation algorithm is done. The proposed segmentation procedure is completed 4 times faster than the manual segmentation with similar accuracy.

Key words: 3D MRI processing, haptic segmentation, parallel processing, virtual reality

1. Introduction

Digital radiological imaging techniques have been widely used for the practice of modern medicine. Medical tomographic images like computed tomography (CT) or magnetic resonance imaging (MRI) can provide very useful anatomical and physiological information, which is used in modern diagnosis and therapy. MRI is an effective approach for getting high-resolution three-dimensional information about the structure and function of different tissues of the body. The implementation of complicated algorithms on such a large amount of spatial data is a time-consuming process using common personal computers. The alternative is to use parallel processing on graphical processing unit (GPU). The most important advantage of GPU application compared to
CPUs is its capability to implement high-resolution floating-point problems like 3D MRI image analysis. Castao et al. [1] evaluated the image processing on a GPU with respect to other common processors and showed its efficiency. Colantoni et al. [2] and Ahn et al. [3] did image processing on normal 2D color images. A 2D/3D image registration was done by Kubias et al. [4] and Ansorge et al. [5]. Klüch et al. proposed a GPU-supported haptic device integrated dental simulation environment using some medical CT images [6].

An important part of medical image analysis is segmentation, which includes the classification of the data elements as object or background and the establishment of the relations between the voxels of the MR images and their related organ or tissue in the human body. The derived semantic results of such a segmentation of the large voxel data sets can be used in diagnosis, therapy monitoring, visualization for surgical planning, surface model creation for surgical simulation, and several other medical applications. Accurate and robust segmentation methods are vital in medical image analysis. The existing methods used in segmentation can be classified with respect to the range of operator intervention or the degree of the algorithm’s automation. It is clear that manual segmentation is very time-consuming and tedious because it needs pixel-by-pixel analysis for precise segmentation. Furthermore, there is intraobserver variability in the manual segmentation. These facts made the manual segmentation also error-prone without any reproducibility. Methods that do not need any direct user interaction are known as automatic segmentation methods. There are numerous automatic segmentation methods, e.g., an automatic brain segmentation based on a generic brain model [7]; an algorithm based on statistical properties of different brain regions [8]; a method based on graph cuts, which assumes the volume of the MRI image as a graph in order to execute the segmentation with maximum flow partitioning [9]; the edge-based method, which uses classification of voxels by locating the boundaries between classes [10]; a method based on the watershed algorithm, which step by step assembles finer parts into one segmented area [11]; and an algorithm based on a combination of wavelet decomposition method and watershed algorithm [12]. However, due to the complexity of the required a priori knowledge regarding the target structures, these algorithms can be effective on precisely defined boundaries. This is the reason why it is difficult to develop a robust and reliable fully automated algorithm for segmentation that can satisfy the users for clinical use. Therefore, semiautomatic or interactive methods are preferred [13–15]. The algorithms given in [16–19] are based on minimization of user interaction energy for each segment. A well designed semiautomatic segmentation algorithm is expected to use the ability of the operator in the recognition of the objects in parallel with the help of the computer power in the object’s delineation. In order to get a successful semiautomatic segmentation, the operator’s interaction load should be minimized without disturbing his control of the segmentation process.

A common semiautomatic algorithm can be initiated with the definition of the seed points by the user. Two-dimensional (2D) image segmentation methods do not lead to acceptable results when extended for three-dimensional (3D) image segmentation. One of the main reasons for such an efficiency decrease is the fact that human interaction with 3D images is harder than with 2D ones. The quality of the seed point selection may have a great impact on efficiency of the segmentation. In this study, virtual reality-based haptic feedback is used in order to increase the efficiency of segmentation and facilitate the seeding process, which is very important in segmentation of MR images.

1.1. Haptic segmentation

Application of haptics and virtual reality techniques can improve the segmentation accuracy by facilitating the manipulation of 3D model and seed point selection during segmentation. The haptic feedback allows the operator to feel the boundaries and find the location of surfaces and structures in a volume [20]. A virtual reality-based interaction for semiautomatic segmentation of medical 3D volume data was proposed in [21],
where a haptic device was used in order to extract the centerline of a tubular structure. This method was used to extract the skeleton of a linear structure with a 3-DOF haptic device for a full segmentation of the desired area. The performance improvement while using a visio-haptic interaction tool with haptic feedback during the semiautomatic segmentation was shown in [22]. The effect of haptic feedback and stereo graphics in a 3D target acquisition task was investigated in [23]. The combination of stereo graphics and haptics was proposed in [24] in order to facilitate the seeding procedure that enables the user to trace vessels in a better way during semiautomatic segmentation of MRI images. A haptic interaction technique based on gradient vector flow (GVF) was proposed in [25] for navigation in 3D images. It enables the operator to remain centered inside objects by providing haptic feedback on the boundaries. Application of the haptic interaction in semiautomatic segmentation of the liver from CT scans by using the fast marching algorithm was proposed in [26] with high reproducibility. In order to facilitate efficient 3D interaction during the segmentation of objects with different complexity, by a varying number of live-wire curves, an algorithm using combined stereo graphics and haptics was used in [27]. A method based on using a haptic device for 3D interactive correction of brain segmentation errors was proposed in [28]. The main goal of the method is to actively use the pial surface, marking the potential segmentation errors on it and manipulating and modifying the surface using visual feedback and a haptic device. After such a modification the new corrected surface is used for semiautomatic volume editing. A fully 3D semiautomatic liver segmentation method was implemented in [29] based on region-growing from seeds with a fast marching method and subsequent fitting of a deformable simplex mesh with shape constraints. A haptic-enabled application used for interactive editing in medical image segmentation was developed in [30]. This application combines haptic feedback, stereo graphics, a fast surface rendering algorithm, and morphological image processing operators in order to manipulate and edit the segmented data interactively. A toolkit called WISH—Interactive Segmentation with Haptics was developed in [31] where the stereo graphics and haptic feedback support the well-known image segmentation algorithms, e.g., fast marching, fuzzy connectedness, and live-wire. An approach for providing haptic interaction with MRI data was proposed in [32] that uses a preprocessing step based on knowledge-based tissue classification and surface information extraction. Such an approach can handle noises as well as overlapping scalar tissue values that increase the quality of the haptic rendering, guiding the user and presenting haptic cues about the size and shape of target tissues like tumors.

In most of the haptic segmentation applications a point is used in order to initialize the automated algorithms. Our proposed algorithm is a novel method based on using a ball with a variable diameter. The ball diameter is adjusted according to the edge map acquired from the gradient map and the segmentation process can be bounded with respect to the volume of the mentioned ball. In the proposed method, the diameter of the ball decreases when we want to access the narrow homogeneous space and its diameter increases when we want to initiate the automated segmentation in a wider homogeneous space. The main advantage of this algorithm is to access all of the homogeneous regions to be segmented in the 3D volume without the limitation of small or narrow regions in an effective way. After the automatic adjustment of the virtual tool ball diameter, the user moves easily through the uniform region to be segmented with the help of haptic feedback. After each seed point selection the segmentation algorithm runs inside the volume of the ball limited by the adjusted diameter. When the ball is moved through the empty or uniform region, the force feedback from all directions cancel each other out and the ball diameter starts to become greater until it reaches the nearest nonuniform region. Large-scale diameter usage provides completion of the segmentation algorithm in a short time. The other important contribution of this study is to use the GPU for the adjustment of the mentioned sphere diameter in real time.
2. The theory used in semiautomatic haptic segmentation

The main structure of the software and the theory used in this study are shown in Figure 1. The software contains preprocessing and rendering algorithms combined with haptic rendering in order to develop an accurate and fast region-growing segmentation application. Table 1 shows the load time and Table 2 shows the runtime jobs.

![Figure 1. The structure of the semiautomatic haptic segmentation software.]

<table>
<thead>
<tr>
<th>Load time</th>
<th>CPU</th>
<th>GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Read MRI data</td>
<td>IDLE</td>
</tr>
<tr>
<td>2</td>
<td>Create intensity map</td>
<td>IDLE</td>
</tr>
<tr>
<td>3</td>
<td>Create gradient map</td>
<td>IDLE</td>
</tr>
<tr>
<td>4</td>
<td>Create edge map</td>
<td>Calculate gradients</td>
</tr>
<tr>
<td>5</td>
<td>Create ray direction textures</td>
<td>Calculate edges</td>
</tr>
<tr>
<td>6</td>
<td>Write maps to GPU</td>
<td>IDLE</td>
</tr>
<tr>
<td>7</td>
<td>Init haptic devices</td>
<td>IDLE</td>
</tr>
<tr>
<td>8</td>
<td>Get haptic device position</td>
<td>IDLE</td>
</tr>
</tbody>
</table>

2.1. Rendering

In order to render and generate the 3D model, we used a ray-tracing approach. Ray tracing is a rendering technique in computer graphics based on global illumination. It traces the rays of light through pixels in an image plane. Then the effects of its encounters with virtual objects can be simulated. If the path of light does not encounter any object, the pixel on the path is shaded with background color. The ray-tracing technique handles shadows, multiple specular reflections, refraction, scattering, and chromatic aberration and texture mapping in a simple way and provides a high degree of visual realism in comparison with other methods [33]. However, the computational cost of the ray-tracing technique is not acceptable for real-time applications. In order to overcome this constraint, a parallel rendering technique can be implemented [34]. In our study, a parallel rendering approach based on the ray-tracing method is used. First, the texture of 2D images is allocated in the global memory of the GPU, and then a 3D model is rendered using multiple threads, where each thread corresponds to one ray.
Table 2. Run time jobs in order on both CPU and GPU.

<table>
<thead>
<tr>
<th>Run time</th>
<th>CPU</th>
<th>GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Send current ball position to GPU and call calculate ray lengths</td>
<td>IDLE</td>
</tr>
<tr>
<td>2</td>
<td>Get camera haptic device position and calculate camera transformation and projection matrices</td>
<td>Calculate ray lengths and store them to ray length texture</td>
</tr>
<tr>
<td>3</td>
<td>For each entry in ray length texture find min length and send this value to GPU and call calculate forces</td>
<td>IDLE</td>
</tr>
<tr>
<td>4</td>
<td>Test if button on haptic device is pressed</td>
<td>Calculate forces using ball position, ray direction texture, and radius and store them in forces texture.</td>
</tr>
<tr>
<td>5</td>
<td>Sum each entry in forces texture to generate total force and send this value to haptic device thread</td>
<td>IDLE</td>
</tr>
<tr>
<td>6</td>
<td>Prepare for rendering; if button clicked call region-growing start rendering</td>
<td>Prepare for rendering (clear back buffer, render states, etc.)</td>
</tr>
<tr>
<td>7</td>
<td>Get haptic position and perform application logic</td>
<td>3D rendering</td>
</tr>
</tbody>
</table>

2.2. Processing of the 3D image

At this step, three 3D maps are used. The first map is the intensity 3D map, which was previously mentioned. The second is the gradient map. The gradient map can be derived from the intensity map using a 3D Sobel kernel \(^1\). The gradient map is used for generating the haptic force feedback. The third map is the edge map, which is derived by thresholding the gradient map in order to generate the edges of the 3D model. The edge map is very important for the proposed algorithm for determining the segmentation region.

\[
\begin{bmatrix}
-1 & -3 & -1 \\
0 & 0 & 0 \\
1 & 3 & 1
\end{bmatrix}
\begin{bmatrix}
X - 1 \\
1 & 3 & 1
\end{bmatrix}
\begin{bmatrix}
-1 & -3 & -1 \\
0 & 0 & 0 \\
1 & 3 & 1
\end{bmatrix}
\begin{bmatrix}
-1 & -3 & -1 \\
0 & 0 & 0 \\
1 & 3 & 1
\end{bmatrix}
\begin{bmatrix}
-1 & 0 & 1 \\
-3 & 0 & 3 \\
-1 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
-3 & 0 & 3 \\
-6 & 0 & 6 \\
-3 & 0 & 3
\end{bmatrix}
\begin{bmatrix}
-1 & 0 & 1 \\
-3 & 0 & 3 \\
-1 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
Z - 1 \\
Z \\
Z + 1
\end{bmatrix}
\]

\(^1\) Figure 2 shows samples of the three 3D maps. Figure 2a shows the intensity map read from MR images, Figure 2b shows the gradient map derived from the intensity map, and Figure 2c shows the edge map derived from the gradient map. After generating the 3D maps, the next step is to implement segmentation using adjustable ball diameter and haptic feedback.
2.3. Ball size adjustment

The proposed segmentation algorithm runs inside a ball volume with a variable diameter. The adjustment of the diameter is based on the edge map, which is generated by using the proposed algorithm GPU. The rays in all directions from the center of the ball are traced until each of them reach an edge voxel within the edge map. The ray that gives the minimum length from the center of the ball to the edge voxel designates the ball radius for the instant center position of the ball. The nearest edge voxel to the ball center position bounds the ball radius (Figure 3). Figure 3a shows the ball state before ball size adjustment. If the number of rays in all directions is not enough there is a possibility to pass an edge voxel by the ball as shown in Figure 3b.

In order to obtain a precise homogeneous volume using an adjusted ball diameter and a haptic force feedback in real time, the number of the rays in different directions is increased and parallel processing on GPU is used (Figure 3c).

In order to guarantee access to all the voxels in the ball, we have developed a ray calculation algorithm that calculates all of the unique rays in the ball for different radiiuses. The algorithm finds rays and stores them in a list, and tests if each ray is in the ball radius. Then it tests if the simplified version of the new ray is presented in the list; if not, it adds to the list and continues. In the given example, ray (1,1,1) has the same direction as (2,2,2) and (3,3,3), etc., and thus after finding ray (1,1,1), the algorithm simplifies ray (2,2,2) to (1,1,1) and cancels it. Figure 4 shows in 2D the unique rays in green and the cancelled ones in red. Figure 5 shows only rays in the ball radius for 2D. The 3D ray directions are stored in an A32R32G32B32F texture, in which x, y, and z rays’ normalized vector components are stored in red, green, and blue channels, respectively, as seen in Figure 6.
Figure 4. Voxels that create unique rays in 2D (green: voxels creating unique rays, red: voxels creating cancelled rays).

Figure 5. Voxels that create unique rays in balls in 2D (green: voxels creating unique rays, red: voxels creating cancelled rays).

Figure 6. Texture representation showing how vectors are stored in a 4-channel 4 × 4 A32R32G32B32F texture. Each texel contains four 32-bit floating point channels (red, green, blue, and alpha channels).

In the A32R32G32B32F texture format, A, R, G, and B are the 4 channels per texel as alpha, red, green, and blue; 32 means 32 bits per channel; and F at the end means that all channels are stored as floating point type.

First, the 2D texture is generated using 256 2D images. Then the generated texture is allocated in the global memory. The position of the ball is sent to the GPU in real time in order to calculate minimum ray length. Finally, the texture containing the minimum radius for each ray is sent back to the CPU and the minimum radius is selected as an adjusted ball diameter. Figure 7 and Table 2 show the steps of ball size adjustment using the GPU. The detailed codes are mentioned in the Appendix. All of the GPU code is generated using high-level shading language.

2.4. Segmentation with haptic feedback

As our algorithm is semiautomatic, the segmentation algorithm is implemented when the operator clicks. In addition to the described method, which provides the ability of segmentation in the narrowest regions using a ball, we designed a haptic feedback-based navigation system that can help the operator to start the segmentation in the best location. We used the global effect of the gradient map in order to generate the haptic force feedback. The higher gradient change leads to the greater force feedback from the haptic device.
Gradient change plays an important role in feedback force calculations. Figure 8 shows gradient vectors in red and gradient magnitude in gray scale for each voxel in 3D. As seen from Figure 8, approaching to the edges, the magnitude of the gradient vector increases and the gradient vectors are in the same direction as the opposite of the normal edge. Thus, when we are closer to edges the generated force increases in the opposite direction, which forces us toward the opposing direction to the edge.

![Gradient Map Update](image)

**Figure 8.** Gradients of the selected section of MRI data; partition of brain. Gradient vector directions in red arrows, gradient magnitude in gray scale (black = 0.0, white = 255.0). The original MR image is shown in the upper left corner of the figure.

Eq. (2) expresses the method of such force feedback generation, where \( \vec{F} \) is the force vector. The coefficient \( a \) is the experimental force regulator constant that is between 0 and 1, \( \nabla \vec{m} \) is the gradient vector of the MRI data, and \( \vec{Rayn} \) is related to the current ray. The term \( \vec{Rayn} \cdot \nabla \vec{m} \) is the dot product of the current ray and gradient vector and \( \| \vec{Rayn} \| \) is the magnitude of current ray that is used for normalization of...
the current ray.

\[ \vec{F} = a \times \frac{1}{\text{Number of Rays}} \times \sum_{n=1}^{\text{Number of Rays}} \left( \sum_{m=1}^{\text{Number of Voxels Through Ray}} (\vec{Ray_n} \cdot \nabla \vec{m}) \right) \times \frac{\vec{Ray_n}}{||\vec{Ray_n}||} \] (2)

Inside the volume there would be little change of intensity and thus the total force would converge to a small value, but on the other hand, when reaching the edges, greater gradient difference produces a much greater force. Obviously, the greatest feedback will be generated when our ball reaches an edge voxel. Such a kind of feedback prevents the user from passing the edges, and also notifies the user with a counter force when reaching the edge region. In addition, the operator can feel the edge while passing from one region or tissue to another one because of the gradient change between them, which leads to the change of the haptic force feedback. In the application of this method, an Omni haptic device is used for generating the force feedback, locating the initialization point, and starting the segmentation. The segmentation algorithm is executed when the operator clicks the button on the Omni robot arm.

Another important point in 3D segmentation is to increase the visualization capability. This means that the operator should be easily and clearly able to see all areas of interest and move inside the 3D model. For this purpose, we used another Omni haptic device combined with 3D glasses for stereo vision, which is shown in Figure 9. This haptic device is used for navigation of the camera inside the 3D model. The 3D glasses give the ability of excellent 3D visualization and lead to the 3D segmentation inside the model with high accuracy in a short time. In our study, the region-growing algorithm is developed in C++ and used as segmentation method. This semiautomatic method classifies the pixels or subregions into greater regions with respect to the predefined criteria as given in [35]. In our system, when the button is clicked, the segmentation starts to grow considering the neighboring voxels in the user-defined threshold range of the seed point’s gray level intensity inside the ball.

![Figure 9. Our setup using two Omni haptic devices, one for segmentation and the other for camera guide.](image)

An example of segmentation is shown in Figure 10. Figure 10a shows a sample of the region-growing segmentation in the 3D volume inside the ball. In order to represent the segmentation clearly for debugging purposes, the three slices (horizontal, vertical, and lateral) related to the center point of the ball are shown in this figure. The second image in Figure 10b shows the top view of the segmentation inside the 3D model. One of the advantages of our designed system is the capability to define and change the upper and lower threshold
in real time during the segmentation. In the basic region-growing method, first we define the regions of interest and then we run the algorithm. The operator can change the number of regions of interest and the number of seed points during the segmentation based on visual feedback and force feedback. Figure 11 shows the 3D simulation and segmentation sample generated using the proposed algorithm. The update rate of the haptic devices used in this study is about 1000 frames per second. In this study, we used a computer with the following configuration: Intel® Core™ 2 Quad CPU Q6600 @ 2.40 GHz, 3.25 GB of RAM, Graphic Card Quadro FX 4600. The rendering rate was 60 frames per second.

Figure 10. Sample of the region growing segmentation: (a) segmentation inside the 3D model using haptic force feedback, (b) top view of the 3D segmentation inside the 3D model.

3. Results and discussion
In order to investigate the efficiency of the algorithm, a comparison among manual segmentation, proposed semiautomatic segmentation, and segmentation using commercial MIMICS software was done. For such a comparison, first the brain MR images were segmented by an expert surgeon using the common region-growing method. In each 2D MR image the exact borders of the tumor were recognized and marked manually by an expert surgeon. Then the segmentation was done inside these marked borders. This process was completed within 62 min. Then a nonexpert operator started to segment the tumor inside the 3D rendered MRI set using haptic feedback and the proposed semiautomatic algorithm. This process was completed in less than 15 min. Region-growing segmentation was then done on MR images using commercial software in 26 min. Parameter adjustment in commercial software directly affects the performance of the region-growing segmentation. Therefore, these parameters were adjusted precisely and the best result was selected for comparison. The segmentation by commercial software is shown in Figure 12. The 3D model and segmented 2D images are shown in this figure. Three segmentation results are shown for an example in Figure 13. Manual and
haptic segmentations are shown in Figures 13a and 13b by overlapping the results. In Figure 13c, commercial software segmentation and manual approach results are overlapped.

**Figure 11.** Sample of 3D simulation and segmentation with the proposed algorithm.

**Figure 12.** Region-growing segmentation using MIMICS (3D model and segmented 2D tumor tissue images).

In Figure 13b, the regions segmented by both manual and haptic methods are shown by blue pixels. In Figure 13c, the regions segmented by both manual method and commercial software are shown by blue pixels. The red pixels are used for the regions segmented only by the surgeon. The green pixels are used for the regions segmented only by the haptic interaction or only by commercial software region-growing method. The
numbers of segmented pixels using each method are compared in Table 3. In Table 3 the segmentation results shown in Figure 13 and the segmentation results for all MR images that include tumor tissues are demonstrated separately. According to the results for one MR layer shown in Figure 13, 99.2% of pixels are segmented correctly when haptic segmentation is used, while 75.8% of pixels are segmented correctly when the commercial software region-growing algorithm is used (Table 3). When all segmented MRI images are evaluated, 99.5% of pixels were segmented correctly using the proposed method while 83.4% correct segmentation was obtained using the commercial software region-growing algorithm. Evaluation of results shows that segmentation using the commercial software region-growing algorithm is error-prone in tumor borders, especially in segmenting the first and last MR images. The mentioned values show higher efficiency of the 3D haptic segmentation method in comparison with the segmentation using commercial software even in tumor borders.

![Figure 13. Sample of segmented image (a). Manual and haptic segmentation (b). Manual and region growing algorithm (c).](image)

While implementing the proposed method, several points should be taken into account in order to get better results. In spite of using preprocessing and noise reduction algorithms, the probable noises in the 3D model can lead to unwanted force feedback and speed reduction during the segmentation. Thus, the quality of 2D images in 3D model generation is an important factor that affects the software. As a future work, a kind of biasing filter can be designed considering the continuity of the edge voxels as a force regulation tool.

In order to obtain accurate results, the number of rays should be increased, as mentioned in Figure 3. Application of such a number of rays combined with a 3D model with high resolution and real-time operation on them requires a high amount of computation, which is often outside the capacity of common CPUs. That is why we were motivated to use parallel processing on the GPU during the rendering and seed point selection of the segmentation.

One of the advantages of the proposed algorithm is implementing segmentation locally based on the gradient change. This leads to a decrease in the connectivity problem in the region-growing method. In addition, our ball bounds the region of segmentation and the segmentation algorithm stops in the sphere
boundaries, which provides faster, more accurate, and controllable region-growing algorithm implementation. In other words, we enter the gradient change as an additional parameter for increasing the efficiency of the region-growing algorithm.

Table 3. Comparison of segmentation results related to Figure 13 and all MR images that include tumor tissues.

<table>
<thead>
<tr>
<th>Method</th>
<th># (%) of true pixels that segmented as true</th>
<th># (%) of false pixels that segmented as false</th>
<th># of true pixels that segmented as false</th>
<th># of false pixels that segmented as true</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual segmentation (grand truth)</td>
<td>1463 (99.2%)</td>
<td>209,481 (99.9%)</td>
<td>1,048,284</td>
<td>-</td>
</tr>
<tr>
<td>Semiautonomic haptic segmentation</td>
<td>1452 (99.5%)</td>
<td>209,405 (99.9%)</td>
<td>1,047,848</td>
<td>78</td>
</tr>
<tr>
<td>MIMICS region-growing segmentation</td>
<td>1109 (75.8%)</td>
<td>209,414 (99.9%)</td>
<td>1,047,691</td>
<td>421</td>
</tr>
</tbody>
</table>

4. Conclusion
Application of the semiautomatic segmentation methods is more reliable for clinical usage in comparison with manual or fully automated approaches. The important factor that affects the efficiency rate in the semiautomatic methods is the rate and the way of operator interference during segmentation. Our designed system provides good 3D visualization advantage to the operator using 3D glasses and a haptic device for camera control, which enables the operator to trace all parts of the 3D space and select the best seed points. In addition, the haptic feedback plays a navigation role during the mentioned process, and it prevents passing the edges and boundaries by the operator and alerts the user when passing from a region to another one based on the gradient change. Therefore, the combination of visual feedback and force feedback provides better understanding of the model and more accurate operation and increases the performance of the operator. Furthermore, our proposed novel strategy, which uses a ball with variable size, enables the operator to reach and segment the branches and narrow corners of the 3D image model with high accuracy and at the same time increases the speed of segmentation in the homogeneous or empty volumes. Flexibility in number of seed points and the adjustment of lower and upper thresholds inside the ball during the segmentation are the other positive points of the proposed system. Application of parallel programming on the GPU has a great potential to increase the efficiency and rate of the computations, especially when dealing with the virtual reality problems of which our algorithm is an example. In conclusion, combination of the virtual reality and haptic feedback techniques has great potential for increasing the efficiency of semiautomatic segmentation of the 3D MR images. The evaluation of the proposed method shows high accuracy of segmentation while the process is 4 times quicker than manual segmentation. It also gives more accurate results compared to the similar region-growing algorithm used in commercial software. Such an accuracy increase is very clear at the boundaries, where intensity-based methods are error-prone.

Acknowledgment
The first author would like to thank the Scientific and Technological Research Council of Turkey (TÜBİTAK-BİDEB) for financial supports during his PhD studies.
References


[33] Nikodym T. Ray tracing algorithm for interactive applications. BSc, Czech Technical University, Prague, Czech Republic, 2010.


**Appendix.** Parallel ball size adjustment algorithm.

<table>
<thead>
<tr>
<th>Global value</th>
<th>BallPosition;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>AdjustBallSize(Texture cell)</td>
</tr>
<tr>
<td></td>
<td>{</td>
</tr>
<tr>
<td>I=1;</td>
<td></td>
</tr>
<tr>
<td>Test if RayUnitVector is a valid Ray vector. If NOT return an invalid value as MaxRayLength.</td>
<td></td>
</tr>
<tr>
<td>If (RayUnitVector.x = 0 &amp;RayUnitVector.y = 0 &amp;RayUnitVector.z = 0)</td>
<td></td>
</tr>
<tr>
<td>Return MaxRayLength;</td>
<td></td>
</tr>
<tr>
<td>Do</td>
<td></td>
</tr>
<tr>
<td>CurrentRayPosition = BallPosition + RayUnitVector*i;</td>
<td></td>
</tr>
<tr>
<td>Convert Ray Position World coordinates to Texture coordinates</td>
<td></td>
</tr>
<tr>
<td>TexCoord = convertTextureCoord(CurrentRayPosition);</td>
<td></td>
</tr>
<tr>
<td>Test If ray blocked by Edge if Blocked we cannot go further thus just return i</td>
<td></td>
</tr>
<tr>
<td>IF Not go Advance 1 unit more.</td>
<td></td>
</tr>
<tr>
<td>If(IsEdge(TexCoord.y, TexCoord.x) )</td>
<td></td>
</tr>
<tr>
<td>Return i;</td>
<td></td>
</tr>
<tr>
<td>Else</td>
<td></td>
</tr>
<tr>
<td>I = i + 1.0;</td>
<td></td>
</tr>
<tr>
<td>Test if we reached maximum predefined Step Size (Maximum Ray Length)</td>
<td></td>
</tr>
<tr>
<td>}while(i&lt;MaxRayLength)</td>
<td></td>
</tr>
<tr>
<td>Return i;</td>
<td></td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>