Interactive Volume Segmentation with Threshold Field Painting

Takeo Igarashi
Dept. of Computer Science
The Univesity of Tokyo

Naoyuki Shono **Taichi Kin**
Dept. of Neurosurgery
The Univesity of Tokyo

Toki Saito
Dept. of Clinical Info. Eng.
The Univesity of Tokyo

ABSTRACT

An interactive method for segmentation and isosurface extraction of medical volume data is proposed. In conventional methods, users decompose a volume into multiple regions iteratively, segment each region using a threshold, and then manually clean the segmentation result by removing clutter in each region. However, this is tedious and requires many mouse operations from different camera views. We propose an alternative approach whereby the user simply applies painting operations to the volume using tools commonly seen in painting systems, such as flood fill and brushes. This significantly reduces the number of mouse and camera control operations. Our technical contribution is in the introduction of the threshold field, which assigns spatially-varying threshold values to individual voxels. This generalizes discrete decomposition of a volume into regions and segmentation using a constant threshold in each region, thereby offering a much more flexible and efficient workflow. This paper describes the details of the user interaction and its implementation. Furthermore, the results of a user study are discussed. The results indicate that the proposed method can be a few times faster than a conventional method.

ACM Classification Keywords

I.3.6 Computer Graphics: Methodology and Techniques—*Interaction Techniques*; I.3.5 Computer Graphics: Computational Geometry and Object Modeling—*Geometric Algorithms*

Author Keywords

Volume segmentation; medical imaging.

INTRODUCTION

Volume segmentation is the process of extracting meaningful regions, such as organs, from 3D volumetric data obtained using scanning devices (e.g., CT and MRI). This is a necessary and important step for medical professionals as it allows informed decision-making by enabling them to see the 3D geometry of the targets. Many sophisticated algorithms for automatic and semi-automatic segmentation have been proposed; however, it is still difficult for these algorithms to work perfectly and professionals heavily rely on manual segmentation such as outlining and thresholding ([26, 33, 31]). Users are often required to draw outlines or paint target regions manually on each 2D slice (e.g. [5]).

This paper addresses the problem of volume segmentation mainly for blood vessels inside brain, and basic manual thresholding is still the most popular method for segmentation in this domain [1]. In thresholding techniques, the user specifies a threshold, and the system then extracts voxels with higher (or lower) values than the threshold. This process is inherently interactive because an appropriate threshold is not known beforehand. The user tests various threshold values iteratively, observes the result, and then adjusts the threshold until a satisfactory result is obtained. When the user needs to apply different thresholds to different regions, the volume is partitioned into multiple regions using bounding boxes or lassos. This process is a notoriously tedious and time-consuming task that requires expert knowledge, concentration, and cannot be easily automated. A doctor interviewed during the study reported that he often spends several hours on such segmentation tasks.

Figure 1 shows a typical workflow in a conventional method [19]. With a high threshold, only a fraction of the target organ, typically a large organ, is visible (Figure 1a). Thus, the user lowers the threshold to view smaller organs; however, irrelevant elements (noise) also become visible and clutter the view (Figure 1b). To observe both the large and small organs appropriately, the user must apply different threshold values to respective regions. The user selects a region containing the small organs using a bounding box tool (Figure 1c) and further removes clutter using lassoing tools (Figure 1d). Finally, the user obtains the final result by combing the surface model obtained by the high threshold and that obtained with the low threshold (Figure 1e). This process might appear straightforward in 2D; however, it actually involves many operations because the data is 3D. Thus, the user must perform these operations from multiple view directions by switching between camera control and tool operations.

To address this problem, we propose an alternative interaction workflow wherein the user *paints* isosurfaces iteratively using tools commonly seen in 2D painting systems, such as flood fill and brushes. Figure 2 shows an example of the proposed workflow. The user begins with a high global threshold (Figure

[1] We confirmed this through personal communications with professional neurosurgeons in more than 20 medical institutes.

UIST 2016, October 16–19, 2016, Tokyo, Japan.

http://dx.doi.org/10.1145/2984511.2984537

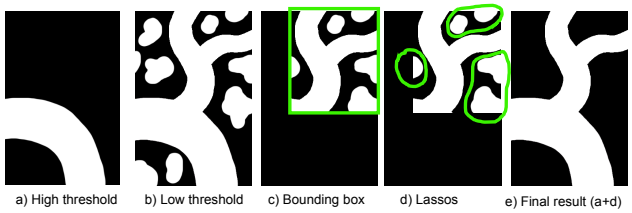


Figure 1. Conventional volume segmentation workflow (bounding box and lassos require operations from multiple camera views)

2a) and applies a flood fill to paint the visible isosurface with a single click (Figure 2b). The painted region will be locked as a part of the segmentation result and will not be affected by the subsequent changes in the global threshold. The user then switches to a brush tool and traces the gradually emerging vessel to paint it with a lower local threshold using a single drag operation[2] (Figure 2c). The user repeats these operations until the final result is obtained (Figure 2d). This workflow is much more efficient than the current workflow because it requires significantly fewer mouse and camera control operations.

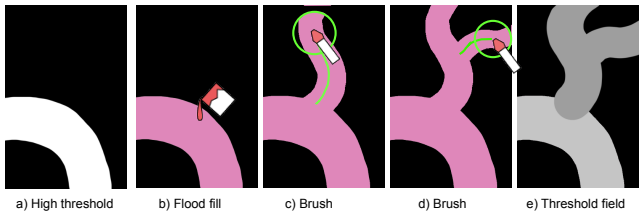


Figure 2. Volume segmentation by threshold field painting (this can be performed in a single camera view)

Internally, the final result is given as a segmented volume obtained with spatially-varying threshold values within the domain (Figure 2e). We call the threshold values assigned to individual voxels a threshold field. This is a generalization of discrete volume decomposition into regions and segmentation using a constant threshold in each region. The user's task is to paint this volumetric threshold field using various painting tools. Figure 3 shows a cross sectional view explaining how varying threshold values produces appropriate isosurfaces. Spatially-varying threshold values (also called as local thresholding) have been used in automatic segmentation ([17, 3]), but they did not directly expose them to the user nor allowed them to manually edit it. Manual editing of 3D threshold field is a non-trivial task and the introduction of painting interface for efficient editing of such a field is a novel contribution of our work.

The target application of the current work is diagnosis and surgical planning for aneurysms. Surgical treatment of aneurysms is particularly difficult because neurosurgeons must carefully avoid many highly sensitive regions surrounding the target aneurysm inside the brain. Thus, to facilitate informed decision-making during surgical planning, the patient's brain is scanned, and the scanned volumetric data is segmented to

[2]We assume that the user has identified the approximate location of the vessel beforehand by inspecting the volume data by applying various global threshold values.

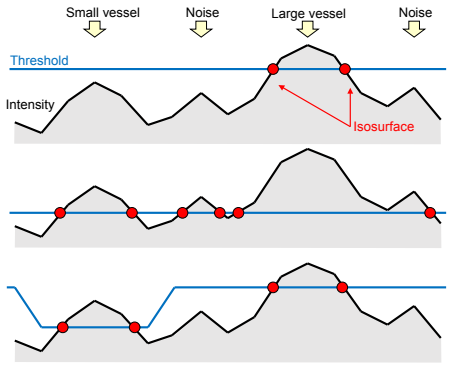


Figure 3. Cross-sectional view. Top: high threshold only captures the large vessels. Middle: low threshold captures noise. Bottom: various thresholds capture both large and small vessels while suppressing noise.

visualize complicated blood vessel structures. Although the current painting interface is primarily designed to extract blood vessels, volumetric segmentation is a fundamental problem with many applications such as diagnosis, surgical planning, and explanations to patients in various medical domains. We believe the proposed workflow can be useful in other domains.

The contributions of this work are summarized as follows.

- We present an interactive workflow for volume segmentation wherein the user paints isosurfaces interactively using a painting interface.
- We introduce the concept of a threshold field, i.e., a collection of different threshold values assigned to individual voxels, as an underlying data representation that enables such interaction.
- We implement an interactive volume segmentation system based on the threshold field and report the results of the evaluation. We demonstrate that the proposed method can be more efficient than a conventional method.

PREVIOUS WORK

Volume segmentation is a fundamental problem, and many automatic methods have been developed previously [28]. There are also segmentation techniques specifically designed for tubular structures, such as blood vessels [7, 20]. However, it remains difficult to obtain desirable results using automatic methods alone; thus, various semi-automatic methods have been proposed. A popular approach is to have the user provide simple hints as input, and the system applies optimization using the user input as constraints. For example, Boykov and Jolly presented a method wherein the user specifies a few point constraints and the system applies graph cut optimization to identify segmentation boundaries [4]. Ramírez et al. presented a grab-cut method for 3D volume segmentation, where the user specifies a box surrounding the target region and the system applies a graph cut algorithm to extract target region inside [30]. Various other algorithms have been proposed to solve similar constrained optimization problems [25, 15]. Another example is to have the user draw on the camera view freely with strokes and infer depth by identifying intrinsic features in the volume data [27]. These methods are primarily designed for rotund objects; however, there are semiautomatic methods

specialized for fibers [2, 1] and sheets [16]. Such methods also take user input in the form of points or freeform strokes and apply optimization to identify tubular or sheet structures.

One class of semi-automatic methods is region growing (active contours or level set method) [32, 37, 18], where the system begins with a seed point or initial contour provided by the user and gradually grows the contour by considering the content of the volume data, e.g., growing rapidly when there is no gradient and stopping growth when there is a large gradient (edges). However, it is still difficult to use this technique in practice because the user must place the initial seed at the right location. In addition, it is necessary to adjust many parameters to control growth appropriately [38].

The proposed method differs from these automatic and semi-automatic methods in that we use simple thresholding as a basis. In the proposed method, the user specifies a threshold explicitly, and the system segments the volume using the threshold without filtering or optimization. Simple thresholding is popular among practitioners because of its simplicity (the user only specifies a scalar value), transparency (it is easy to understand what the system is doing), and predictability (small changes in user input do not cause sudden changes in the result), which are often lacking in automatic methods. However, a problem with simple thresholding is that the user must partition the volume to apply different thresholds to different regions, which causes significant operational overhead. Our goal is to address this problem by introducing sophisticated interaction techniques and novel data representation.

Volume visualization also has a long history of research and some of them are closely related to our work. Guo et al. presented a painting interface for editing transfer functions in volume rendering [10]. The user can selectively change color and opacity of certain voxels by drawing strokes on the screen to obtain desired volume rendering results (raster image). Despite superficial similarity in the interaction, our workflow and toolset are very different from theirs since our goal is the extraction of 3D meshes representing isosurfaces. For example, operations such as changing local threshold while applying the brush tool and changing global thresholds while fixing the thresholds of painted voxels are not seen in their work. Huang and Ma presented interactive volume visualization based on region growing. They propose to generate 2D transfer function for visualization based on partial region growing [12]. Zhou et al. presented a method to interactively control volume visualization by setting color and opacity to segmentation result obtained by mean-shift [39]. Fujishiro et al. presented the concept of interval volume as a generalization of an isosurface [9]. They proposed to obtain 3D solid representing subvolume bounded by two threshold values for efficient data exploration.

Our user interface is inspired by interactive segmentation tools for 2D images. Intelligent scissors snap freeform user strokes to visually salient image boundaries [24]. Soft scissors apply segmentation inside of a brush tip continuously as the user traces a boundary [35]. Paint selection [21] updates segmentation results progressively as the user specifies the foreground region with a paint operation. The notable feature of these methods is that the system presents feedback to the user continuously as they drag the mouse cursor. Our goal is to achieve such fluid interaction in 3D volume segmentation.

This work also builds on various interactive techniques developed for 3D modeling and texture painting. Our brush tool resembles techniques that generate a freeform mesh around a user-drawn freeform stroke in empty space [36, 23]. 3D sculpting tools allow the user to sculpt a 3D model surface by applying paint operations [29, 34]. Sketch-based modeling systems allow the user to create 3D models by drawing silhouettes [14]. Texture painting in a 3D view was first presented by Hanrahan and Haeberli, [11], and various extensions have been proposed to date [13, 8]. We apply interactive techniques presented in such systems to paint a threshold field and sculpt isosurfaces to assist volume segmentation.

USER INTERACTION

Here, we describe the system from the user's perspective. The input to the system is volumetric data, i.e., a 3D array of intensity values obtained using a scanning device, such as CT and MRI. The output is a segmentation result that represents a region of interest within the domain (blood vessels in the current target application). The segmentation result is visualized as an isosurface, which is computed using the marching cubes algorithm [22]. The user's task is to obtain a meaningful segmentation result by applying painting operations. This process requires significant expert knowledge and is difficult to automate. The user must examine the volume data carefully and construct a segmentation result interactively.

Here, we illustrate the process using a typical workflow. Note that this particular workflow is presented for illustration purpose only, and the user does not need to follow this workflow exactly. This process is explorative in nature, and the user must make various decisions by examining the data interactively. As discussed in the introduction, our target application is the extraction of vessels from a volumetric scan of a human brain. Thus, the task is to trace fine vessels surrounded by clutter.

Figure 4 shows a screenshot of the proposed system. It intentionally mimics a typical paint tool user interface, such as Windows Paint and Adobe Photoshop, to leverage the painting metaphor. The left panel shows the tools, e.g., flood fill and brush. The bottom panel shows the slider for controlling the global threshold. The top panel shows the camera control. The center panel shows the segmentation result as a collection of isosurfaces. We assign various colors to different regions of the isosurfaces to visualize threshold values used for the region. If the region is surfaced using a low threshold, we assign darker colors (meaning the signal was weak). If the region is surfaced using a high threshold, we assign brighter colors. In addition, we assign red hues to painted regions and monochrome colors to the rest.

The user first loads the volume data into the system and examines the overall structure by observing the data with various global thresholds. The user changes the global threshold freely using the slider. If the global threshold is low, the user can see details; however, such details are often occluded by clutter (Figure 5a). If the global threshold is high, the user can

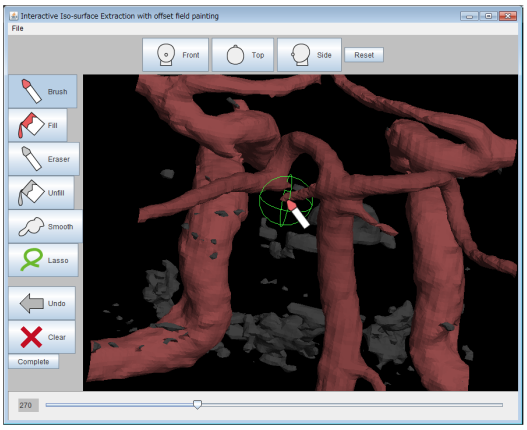


Figure 4. Screenshot of the proposed system

see dominant structures but details are lost (Figure 5b). After grasping the overall structure of the data by viewing the data with various threshold values, the user sets a high global threshold such that sufficient structure is visualized while surrounding clutter is hidden. The user then selects the flood fill tool in the left panel and clicks on the desired segmented regions on the screen (large vessels in our application). At this point, the voxels inside of the selected regions are painted (Figure 5c). The painted region becomes reddish. Painting locks the region so that the segmented region will not be affected by the global threshold value controlled by the slider. The user repeats the above process to segment the entire volume roughly with various thresholds (Figure 5d, e).

One can complete most of the task using only the flood fill tool in simple cases; however, minor tweaks are often required for complicated cases. For example, if the target fine vessel is surrounded by significant amounts of clutter, the clutter can prevent the user from clicking a target when the global threshold is low. The brush tool is useful in such cases. With the brush tool, the user first hides the clutter by raising the global threshold and applies the brush tool where the fine vessel begins. Note that the user cannot click on empty space; therefore, they click on a visible isosurface where the (currently invisible) fine vessel is connected. The user then lowers the threshold value inside the brush tip to make the fine vessel visible (Figure 5f). The user uses the left and right arrow keys on the keyboard to lower and raise the brush threshold while pressing the mouse button. After setting the brush threshold sufficiently low, the user performs a drag operation to trace the fine vessel (Figure 5g). As the user drags the mouse cursor, the threshold inside the brush tip is updated, and the vessel gradually appears. The depth of the brush tip is set to that of an gradually emerging isosurface under the mouse cursor. Note that, as the user proceeds, appropriate threshold values may change. In such cases, the user can adjust the brush threshold using the left and right arrow keys. The user can also adjust the size of the brush tip by pressing the up and down arrow keys. The region painted by the brush is also locked; thus, it will not be affected by the global threshold later (Figure 5h).

If the user incorrectly paints a region or determines that the painted area is inappropriate, they can undo paint operations easily or explicitly erase the painted result using the eraser tool. The erased region will be unpainted and will be once again affected by the global threshold (Figure 6b). We also provide an unfill tool, which reverses flood fill operations. The unfill tool collects painted data points connected to the clicked surface and converts their state to unpainted (Figure 6c). Another tool is for smoothing the user-painted threshold field. This tool applies a simple Laplacian smoothing (diffusion) operation to the threshold values around the clicked point. This tool is useful for alleviating discontinuous changes in the threshold values that appear at brush boundaries (Figure 6e). Finally, the system provides a mask tool that explicitly excludes certain regions from segmentation. The user draws a lasso on the screen, and the data points inside the lasso are masked (i.e., excluded from segmentation). This mask tool is not really necessary for our workflow; however, it is the main tool used in conventional systems and has been provided because users are familiar with such tools.

The user repeats the above operations until all necessary regions are painted appropriately (i.e., they have become reddish). The user can see the overall appearance of the painted regions by setting the global threshold sufficiently high such that all unpainted regions are hidden. The system then exports the final segmentation result as an isosurface associated with the painted regions. The output is a standard mesh model that can be imported into standard systems.

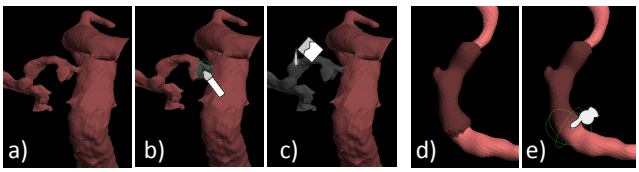


Figure 6. Additional tools: a, b) Eraser, c) Unfill, d, e) Smooth.

IMPLEMENTATION

The system stores three volumetric arrays of equal size internally. The input volume data stores the intensity value of each data point. This data remains fixed throughout the process. The threshold field, which can be edited (painted) by the user, stores a threshold value for each data point. The state field stores state data (painted or unpainted) for each data point. Initially, all states are set to unpainted. They are set to painted when the user applies paint operations, such as flood fill and brush.

We use the standard marching cubes algorithm for isosurface extraction. We compute modified volume data by subtracting the threshold field values from the original intensity values. Then, we apply the marching cubes algorithm to the modified volume data (threshold value = 0). This guarantees that the extracted isosurface is watertight, with the exception of the volume boundary.

The following subsections describe implementation details. Here we define several terms used in the descriptions (Figure 7). We refer to volumetric data points as nodes. Each node

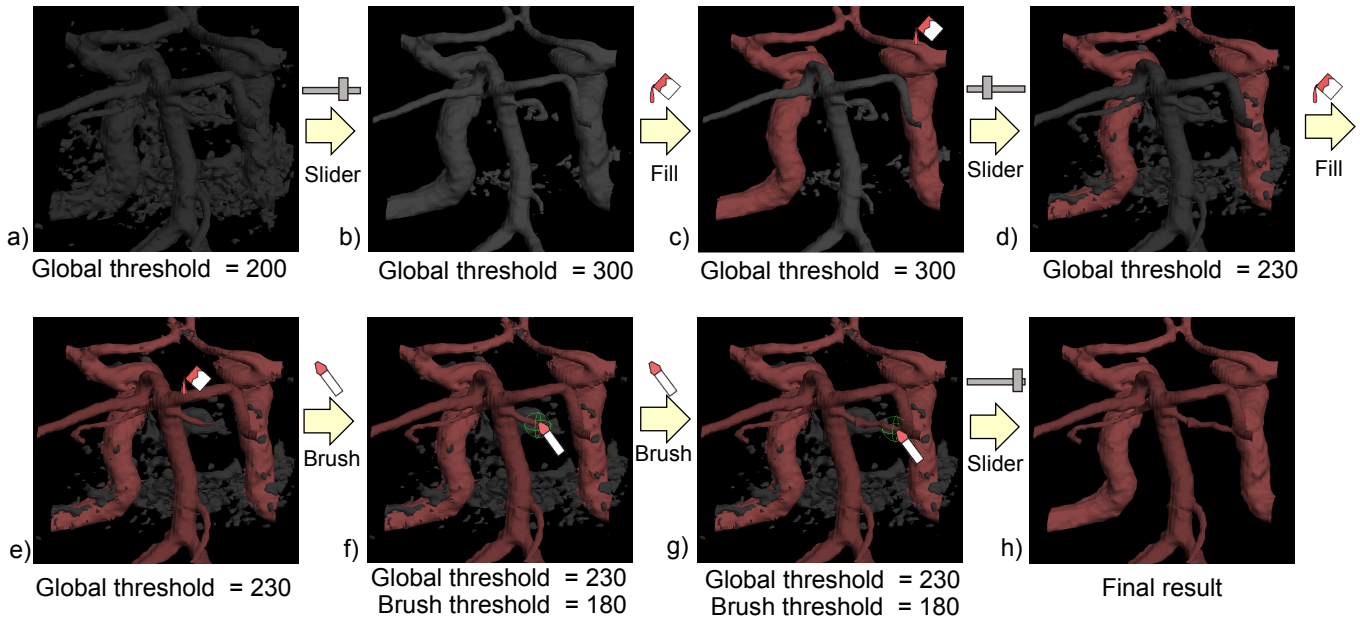


Figure 5. Volume segmentation by threshold field painting

is associated with intensity, threshold, and state values. The marching cubes algorithm generates mesh faces inside a cell surrounded by eight adjacent nodes. If we have XYZ nodes, we have (X-1)(Y-1)(Z-1) cells in total. Each node has six neighbors.

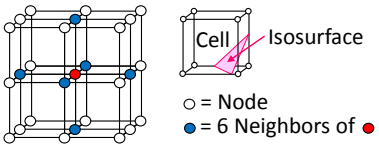
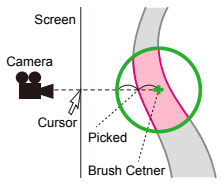


Figure 7. Data representation.

Tool Operations

The flood fill operation works as follows. When the user clicks on the screen, the system applies a pick (projection) operation to find a face of the current isosurface under the mouse cursor. The system then identifies the cell that contains the face and visits the eight nodes surrounding the cell. If the node's intensity value is higher than its threshold value (i.e. the node is foreground), the system sets its state to painted. The system recursively visits the six neighbors of the foreground nodes and sets their states to painted. Propagation stops when the next node's intensity is lower than its threshold (i.e., in the background) or the state is already painted.

The brush operation works as follows. As the user drags the mouse cursor, the system applies a pick operation to find a foreground cell. If no cell is picked, the system does nothing. The system then places the brush tip at the center of the picked cell. As an additional tweak to improve the usability, the system moves the brush tip slightly in the viewing direction ($0.5 \times brushradius$) from the picked position. This makes the brush tip cover the vessel under the cursor more efficiently (see Inset). The system collects the nodes whose distance to the brush tip is less than the brush radius. The system then sets the threshold of the collected nodes to the brush threshold. If the threshold of the node becomes lower than its intensity, the system sets its state to painted. Otherwise, its state is set to unpainted.



At the beginning of the brush operation (dragging), the system sets the brush threshold to the threshold of a foreground node surrounding the picked cell. If there are multiple foreground nodes around the cell, the threshold is set to the minimum. The user can change the brush threshold during dragging using arrow keys. We also tested an alternative design wherein the brush threshold was carried over to the next paint operation; however, this alternative was frustrating because the underlying isosurface suddenly disappears when a brush operation is initiated if the carried brush threshold is higher than the intensity value of the picked node. This does not occur if the brush threshold is selected from the picked node for each operation.

The user can adjust the slider at the bottom of the screen to change the global threshold. When the global threshold is changed, the system visits all unpainted nodes and sets their threshold to the global threshold. However, this naive approach is problematic because it changes the appearance of the painted isosurface (the isosurface inflates and deflates slightly), as shown in Figure 8 (left). To prevent this, we also lock the nodes neighboring the painted nodes, as shown in Figure 8 (right). Thus, we can prevent changes in the appearance of the isosurface when the global threshold is

changed. In other words, the painted isosurface is protected (coated) by a thin layer of locked nodes that are unaffected by the slider operation.

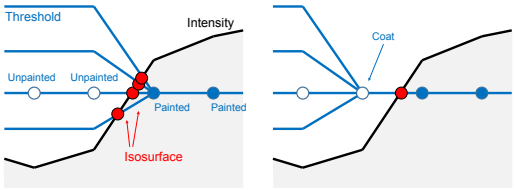


Figure 8. Coated region. Without coating, the isosurface fluctuates as the user changes the global threshold (left). Therefore, the system also fixes the threshold of a coated node neighboring the painted nodes to stabilize the isosurface (right).

The smoothing tool works in a way similar to the brush tool, except that the smoothing tool sets the threshold of a node inside the brush tip to the average of its neighboring nodes. More specifically, the tool sets the threshold of a *painted* node to the average of its neighboring *painted* nodes. Unpainted nodes are excluded in the smoothing operation even if they are inside the brush tip. After updating the thresholds, the system sets the state of a node to painted if its intensity is higher than its smoothed threshold, and sets its state to unpainted otherwise.

Ghost Isosurface

A problem with our painting interface is that the system can create a false-positive isosurface (ghost isosurface) at the boundary of a painted area even if there is no corresponding gradient in the intensity values. For example, consider the situation shown in Figure 9, where there is a background field with intensity values of 100. If there are high intensity regions inside the background with intensity values of 200, then it is appropriate to paint the area surrounding the region with a threshold value of 150 to visualize the high intensity region. However, if the user paints the area with a threshold value of 50, then a visible ghost isosurface will appear at the boundary of the painted region. This occurs when a sudden change in threshold values occurs where there is no gradient in the intensity values.

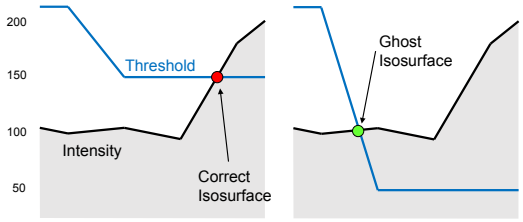


Figure 9. Ghost isosurface. A properly designed threshold field appropriately captures the high intensity region (left). However, a poorly designed threshold field can produce a false-positive isosurface where there is no high intensity region (right).

We address this problem by automatically detecting such ghost isosurfaces and assigning a special color (green) to the surface to warn the user. For each cell, we compare the gradient descent direction of the intensity values and the normal direction of the isosurface. If these two directions are not aligned (facing opposite directions), then the isosurface does not correctly represent the intrinsic structure of the original data. Therefore, the system identifies the faces inside the cell as a ghost isosurface and colors them green. We currently use the gradient descent direction of the modified volume (intensity minus threshold) as the isosurface normal to simplify the implementation.

Figure 10 shows example cases. In Figure 10 (left), the user sets a very low threshold value inside the brush tip. This generates an undesired ghost isosurface at the boundary of the brush tip (isosurface normal is outwards while gradient descent is inwards) and the system warns the user by making such regions green. The user is encouraged to remove these green regions by adjusting the threshold values around them. In Figure 10 (right), there are also non-zero gradient threshold values around the brush tip. However, in this case, the normal of the isosurface is almost aligned to the gradient descent; therefore, the system does not consider it a ghost isosurface. This occurs whenever the user applies varying thresholds to a continuous isosurface. The algorithm employed in the proposed method can distinguish between these two cases.

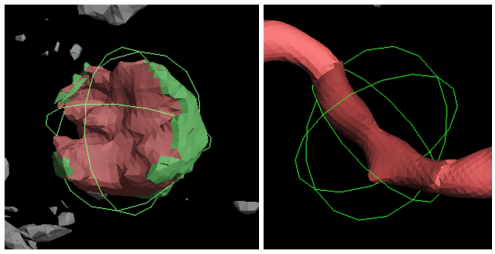


Figure 10. Left: Isosurface at the brush boundary is identified as a ghost because the isosurface normal is very different from the gradient descent direction of the intensity values. Right: Isosurface at the brush boundary is not identified as ghost because the isosurface normal is sufficiently close to the gradient descent.

RESULTS

We have implemented a prototype system in 64-bit Java using Jogl, and run it on a laptop PC running 64-bit Windows 8.1 Pro. with Intel Core i7 (3.30GHz) CPU and 32 GB memory. We have tested the prototype system with various datasets as shown in Figure 11. It shows the segmentation results obtained using the proposed method by an author, following the workflow illustrated in Figure 5. Most results took only a few minutes, but some complicated ones (a and h) took several minutes. The most expensive computation is isosurface extraction (marching cubes) as the user changes the global threshold using the slider, but it still runs at an interactive rate even though it is not fully optimized for speed. Painting operations such as flood fill and brushes are computationally inexpensive and the computation completes instantly.

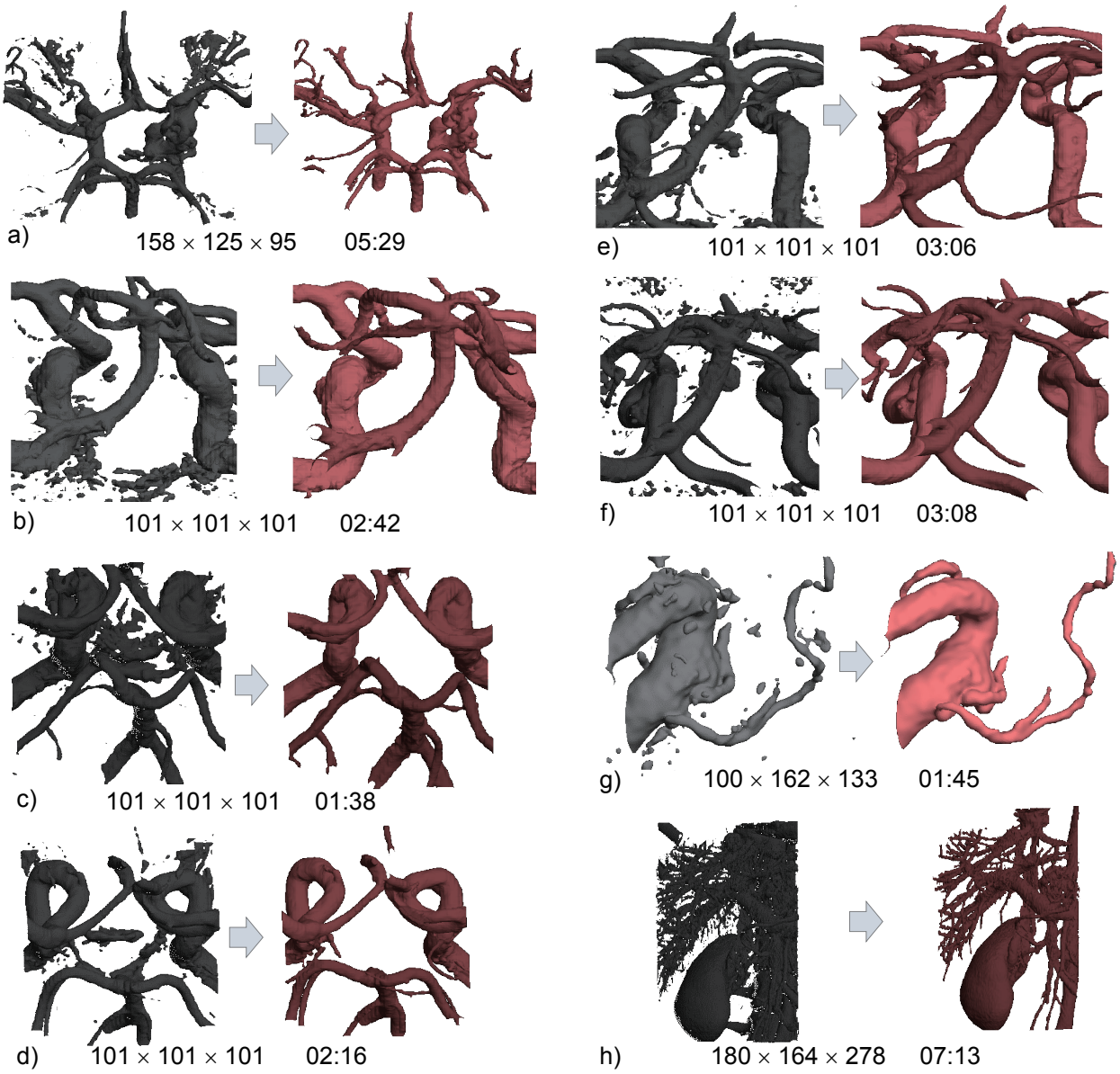


Figure 11. Volume segmentation results using our system. In each pair, left shows the isosurface before painting (extracted by a global threshold) and the right shows the isosurface after painting (extracted by the user-defined threshold field). The number below indicates the volume size and the time spent for each manual segmentation. Data source; a-f: obtained using time-of-flight magnetic resonance angiography (TOF-MRA). g: three-dimensional rotational angiography (3D-RA). h: computed tomography (CT). a-g show blood vessels inside brain and h shows kidney and liver.

USER STUDY

We conducted a user study to compare the proposed method to a conventional method to assess effectiveness. We used commercial state-of-the-art software (Avizo [6]) as the conventional method. We used our prototype implementation described in the previous section. The goal in this user study was to measure expert performance. Therefore, we asked the participants to practice with predefined volume data. We then asked the participants to work on the main task using the same data repeatedly. It was expected that this would minimize the time required for learning and exploration during the test and allow head-to-head comparison of manual task execution performance. Measuring the time required for exploration would also be useful; however, we did not undertake that measurement in this study because the variance would be significant, making it difficult to compare the results quantitatively.

Procedure

The task is to extract blood vessels inside a given volume data (a scan of a human brain). We used the same volume data throughout the study and explicitly specified target blood vessels to be extracted inside the volume data (see Appendix for the details). The task was intentionally designed to be simple and straightforward, i.e., it required basic tools, such as flood fill and lassos. Four volunteers participated in the study. Table 1 summarizes their profiles. All were experienced neurosurgeons who used the commercial software to perform segmentation tasks on a daily basis. The study took approximately 1.5 hours for each participant. First, we described the task and explained how to use the two tools using a practice dataset. We guided the participants as they practiced using the tools. Then, we provided a different dataset (which will be used for the main task) and asked the participants to practice the task five times on their own using the two tools in turn. We did not provide any advice during this second practice run; consequently, each participant was free to pursue different strategies. After a short break (10 minutes), the participants worked on the main task using the two tools in turn. For each tool, the participants worked on the same task five times consecutively. The order of using the two tools was balanced among the participants.

Table 1. Profiles of the participants

ID	A	B	C	D
Sex / Age	Male / 39	Male / 36	Male / 36	Male / 35
Position	Resident	Research Associate	Ph.D Student	Ph.D Student
Years of experience in medicine	15	13	9	10
Years of experience using Avizo	4.4	5.1	1.2	2.8

Figure 12 shows a screenshot of the commercial software with the dataset employed in the user study for the main task. The size of the dataset was $101 \times 101 \times 101$. The participants primarily used the lasso tool to remove clutter, as shown in this figure. The right pane shows a data flow diagram that allows the user to configure the visualization process. The users used the data flow diagram to specify the process of duplicating the original volume, applying different thresholds to each, removing irrelevant segments from each, and finally merging the volumes.

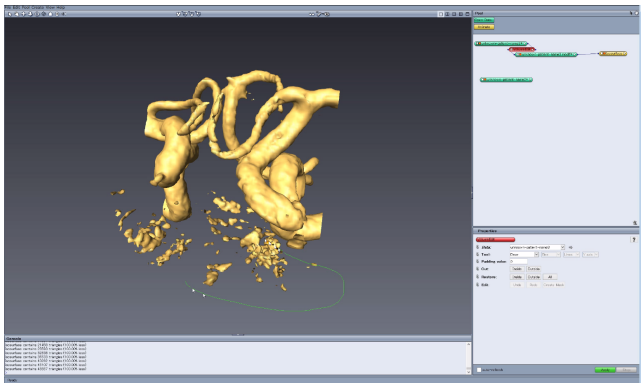


Figure 12. Screenshot of the commercial software (Avizo)

Results

Figure 13 (left) shows the task completion time. The participants completed the task faster using the proposed method (average 64 seconds) than using the conventional method (150 seconds). Figure 13 (right) shows the number of mouse operations. Again, the proposed method required fewer mouse operations (average 27) than the conventional method (76 operations). Evaluation of segmentation quality is difficult because the segmentation results (3D surface models) are all different (the supplemental material shows the variation). This is because the instruction only specified the most important points and details were left to each participant. This mimics what the neurosurgeons do in practice; because of tight time constraints, they focus on the most important parts relevant to diagnosis and planning without paying too much attention to the remaining parts. We therefore recruited another neurosurgeon to examine the segmentation results and asked for his opinion. He considered that the observed differences are not critical and confirmed that all the results have sufficient quality for clinical purposes.

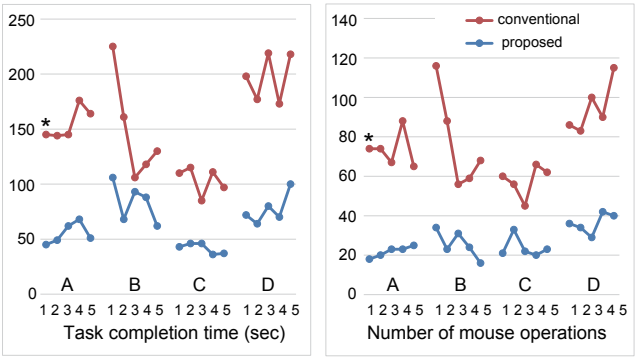


Figure 13. User study results (* indicates that a fraction of a required vessel was missing in the segmentation result.)

Table 2 presents a more detailed statistics. In addition to the total task completion time, we also measured the time spent referring to the data flow diagram pane (Figure 12) in the conventional method. The time spent in the data flow pane

was approximately 30 seconds on average (column Time′); therefore, the time spent in the main view was approximately 121 seconds, which is still much longer than the time required using the proposed method. The number of tool operations (lasso, fill, and brush) is much less with the proposed method (5.1 vs. 9.6). With the proposed method, most participants completed the task using only the flood fill tool. However, two participants also used the lasso and brush occasionally. The participants used the slider to adjust the threshold more frequently when using the proposed method (4.4 vs. 2.7). This is probably because choosing the right threshold is critical for flood fill to capture the target vessel successfully.

Table 2. User study results (Average of five trials. Time is task completion time; Mouse is the number of mouse operations, including clicks and drags; slider is the number of threshold adjustments; lasso, fill, and brush show the number of respective tool operations.)

	ID	Time	Time'	Mouse	Slider	Lasso	Fill	Brush
Proposed	A	55		21.8	4.6	0	3	0
	B	83.4		25.6	4.8	0.2	4.6	1
	C	41.6		23.8	4.6	0	4.2	0
	D	77.2		36.2	3.4	0.2	3	4
	Ave	64.3		26.9	4.4	0.1	3.7	1.3
Conventional	A	154.8	21.3	73.6	2.4	11.4		
	B	148	29.8	77.4	2.4	8.2		
	C	103.6	33.8	57.8	2.4	6.8		
	D	197	35.1	94.8	3.6	11.8		
	Ave	150.9	30.0	75.9	2.7	9.6		

We solicited feedback from the participants after completion of the tasks. They all understood how to use the proposed method quickly and appreciated its novel workflow, saying that the method would significantly reduce the time required for volume segmentation of blood vessels. The participants used a limited set of tools in the test; however, we did introduce and explain the other tools, such as the eraser tool and smooth tool, and the participants agreed that these tools would probably be useful to handle complicated cases. They also identified various issues with the current implementation. The primary complaint was the sluggish behavior of the current unoptimized prototype implemented in Java. However, we do not see this as a fundamental problem because the proposed method does not require additional costly computation compared to the conventional method and a more efficient implementation should be possible with a little more engineering effort.

DISCUSSIONS AND FUTURE WORK

A possible criticism of the proposed approach is that the goal of medical imaging should be to observe measured data as is. It is inappropriate to allow the user to *sculpt* data freely using design tools. However, we do not think that this criticism applies to the proposed method because we do not modify the measured intensity data itself and only modify the threshold field. Various image filtering methods are commonly applied to medical images (such as brightness and contrast) to improve the readability of the data, and the proposed threshold field can be considered a new form of filtering of volumetric intensity data. We have discussed this issue intensively with medical professionals, and they have confirmed that it is acceptable to use manual threshold painting for clinical purposes provided the user correctly understands how to interpret the segmentation result. The visualization of ghost isosurfaces (Section 4.2) is an important step toward this goal.

The current toolset is very rudimentary with only a few basic tools provided. However, the painting metaphor is very powerful, and we can further leverage the metaphor by implementing various other tools seen in image editing software (e.g., Photoshop). For example, advanced selection tools such as magic wand would significantly improve productivity when working on a complicated structure. In addition, layering would be necessary to segment volume data that consists of large and heterogeneous organs.

The current brush tool with a spherical tip is primarily designed to trace tubular structures, and it may not be ideal for other structures, such as lumps and sheets. However, the threshold painting itself is a very general concept and should be applicable to other structures if we develop appropriate tools. We are considering developing a brush tool with a circular tip whose normal is always set parallel to the intensity gradient, which may be useful for tracing sheets and the surface of lumps.

In this work, we have intentionally avoided automatic or semi-automatic methods to clarify the strengths and limitations of plain threshold painting. However, our contribution in the user interface and the data representation is orthogonal to existing filtering and optimization techniques and can be combined with them to facilitate advanced segmentation processes. In addition, automatic or semi-automatic computation of the proposed threshold field may present interesting research opportunities, which we plan to explore in future.

CONCLUSION

We have presented an interactive method for volume segmentation. The main contribution is the introduction of the painting metaphor for volume segmentation and the threshold field as an underlying data representation. The target users, i.e., neurosurgeons, found the proposed method highly novel and expressed a strong desire to use it in daily practice. Our small-scale user study showed that the proposed method can outperform a conventional method in terms of task completion time by a factor of two or more. Neurosurgeons regularly spend many hours in segmentation tasks, and thus our method, which greatly increases the speed at which this task can be completed, can make a significant impact to the field.

Acknowledgements

We thank the neurosurgeons who participated in the study for their time and valuable feedback. We also thank Thomas Auzinger for his help in writing the paper. This work was partially supported by JSPS KAKENHI Grant Numbers JP26240027.

REFERENCES

1. Abeysinghe, S., and Ju, T. Interactive skeletonization of intensity volumes. *The Visual Computer 25*, 5-7 (2009), 627–635.
2. Akers, D. CINCH: a cooperatively designed marking interface for 3d pathway selection. In *Proceedings of the*

19th annual ACM symposium on User interface software and technology (UIST '06) (2006), 33–42.

3. Batenburg, K., and Sijbers, J. Automatic local thresholding of tomographic reconstructions based on the projection data. In *Medical Imaging*, International Society for Optics and Photonics (2008), 69132P–69132P.
4. Boykov, Y. Y., and Jolly, M. P. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. *Computer Vision (Proceedings on ICCV 2001) 1* (2001), 105–112.
5. Fedorov, A., Beichel, R., Kalpathy-Cramer, J., Finet, J., Fillion-Robin, J.-C., Pujol, S., Bauer, C., Jennings, D., Fennessy, F., Sonka, M., et al. 3d slicer as an image computing platform for the quantitative imaging network. *Magnetic resonance imaging 30*, 9 (2012), 1323–1341.
6. FEI. Avizo. `www.fei.com/software/avizo3d/`.
7. Frangi, A. F., Niessen, W. J., Vincken, K. L., and Viergever, M. A. Multiscale vessel enhancement filtering. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI'98* (1998), 130–137.
8. Fu, C.-W., Xia, J., and He, Y. Layerpaint: a multi-layer interactive 3D painting interface. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)* (2010), 811–820.
9. Fujishiro, I., Maeda, Y., Sato, H., and Takeshima, Y. Volumetric data exploration using interval volume. *Visualization and Computer Graphics, IEEE Transactions on 2*, 2 (1996), 144–155.
10. Guo, H., Mao, N., and Yuan, X. Wysiwyg (what you see is what you get) volume visualization. *Visualization and Computer Graphics, IEEE Transactions on 17*, 12 (2011), 2106–2114.
11. Hanrahan, P., and Haeberli, P. Direct WYSIWYG painting and texturing on 3d shapes. *Computer Graphics* (1990), 215–224.
12. Huang, R., and Ma, K.-L. Rgvis: Region growing based techniques for volume visualization. In *Computer Graphics and Applications, 2003. Proceedings. 11th Pacific Conference on*, IEEE (2003), 355–363.
13. Igarashi, T., and Cosgrove, D. Adaptive unwrapping for interactive texture painting. In *Proceedings of the 2001 symposium on Interactive 3D graphics (I3D '01)* (2001), 209–216.
14. Igarashi, T., Matsuoka, S., and Tanaka., H. Teddy: A sketching interface for 3d freeform design. In *Proceedings of SIGGRAPH 99* (1999), 109–116.
15. Ijiri, T., Yoshizawa, S., Sato, Y., Ito, M., and Yokota, H. Bilateral hermite radial basis functions for contour-based volume segmentation. *Computer Graphics Forum 32*, 2 (2013), 123–132.
16. Ijiri, T., Yoshizawa, S., Yokota, H., and Igarashi, T. Flower modeling via x-ray computed tomography. *ACM Transaction on Graphics 33*, 4 (July 2014), 48:1–48:10.
17. Jentzen, W., Freudenberg, L., Eising, E. G., Heinze, M., Brandau, W., and Bockisch, A. Segmentation of pet volumes by iterative image thresholding. *Journal of Nuclear Medicine 48*, 1 (2007), 108–114.
18. Jeong, W.-K., Beyer, J., Hadwiger, M., Vazquez, A., Pfister, H., and Whitaker, R. T. Scalable and interactive segmentation and visualization of neural processes in em datasets. *IEEE transactions on visualization and computer graphics 15*, 6 (2009), 1505–1514.
19. Kin, T., Shin, M., Oyama, H., Kamada, K., Kunimatsu, A., Momose, T., and Saito, N. Impact of multiorgan fusion imaging and interactive 3-dimensional visualization for intraventricular neuroendoscopic surgery. *Neurosurgery 69*, 1 (2011), ons40–ons48.
20. Lesage, D., Angelini, E. D., Bloch, I., and Funka-Lea, G. A review of 3d vessel lumen segmentation techniques: Models, features and extraction schemes. *Medical Image Analysis 13*, 6 (2009), 819–845.
21. Liu, J., Sun, J., and Shum, H.-Y. Paint selection. *ACM Transaction on Graphics 28*, 3 (2014), 69:1–7.
22. Lorensen, W. E., and Cline, H. E. Marching cubes: A high resolution 3d surface construction algorithm. *Computer Graphics 21*, 4 (1987), 163–169.
23. Markosian, L., Cohen, J. M., Crulli, T., and Hughes, J. Skin: a constructive approach to modeling free-form shapes. In *Proceedings of SIGGRAPH 99* (1999), 393–400.
24. Mortensen, E. N., and Barrett, W. A. Intelligent scissors for image composition. In *Proceedings SIGGRAPH 95*, ACM (New York, NY, USA, 1995), 191–198.
25. Nock, R., and Nielsen, F. Statistical region merging. *IEEE Transactions on pattern analysis and machine intelligence 26*, 11 (2004), 1452–1458.
26. Oishi, M., Fukuda, M., Ishida, G., Saito, A., Hiraishi, T., and Fujii, Y. Presurgical simulation with advanced 3-dimensional multifusion volumetric imaging in patients with skull base tumors. *Neurosurgery 68* (2011), ons188–ons199.
27. Owada, S., Nielsen, F., and Igarashi, T. Volume catcher. In *Proceedings of the 2005 symposium on Interactive 3D graphics and games (I3D '05)*, vol. 13 (2005), 111–116.
28. Pham, D. L., Xu, C., and Prince, J. L. A survey of current methods in medical image segmentation. *Annual review of biomedical engineering 2* (2000), 315–338.
29. Pixologic. Zbrush. `www.pixologic.com`.
30. Ramírez, J., Temoche, P., Carmona, R., et al. A volume segmentation approach based on grabcut. *CLEI Electronic Journal 16*, 2 (2013), 4–4.
31. Rudyanto, R. D., Kerkstra, S., Van Rikxoort, E. M., Fetita, C., Brillet, P.-Y., Lefevre, C., Xue, W., Zhu, X., Liang, J., Öksüz, İ., et al. Comparing algorithms for automated vessel segmentation in computed tomography scans of the lung: the vessel12 study. *Medical image analysis 18*, 7 (2014), 1217–1232.

32. Sethian, J. A. Level set methods and fast marching methods, 1999.

33. Shimizu, M., Imai, H., Kagoshima, K., Umezawa, E., Shimizu, T., and Yoshimoto, Y. Detection of compression vessels in trigeminal neuralgia by surface-rendering three-dimensional reconstruction of 1.5-and 3.0-t magnetic resonance imaging. *World neurosurgery 80*, 3 (2013), 378–385.

34. Stanculescu, L., Chaine, R., and Cani, M.-P. Freestyle: Sculpting meshes with self-adaptive topology. *Computers & Graphics 35*, 3 (2011), 614–622.

35. Wang, J., Agrawala, M., and Cohen, M. F. Soft scissors: an interactive tool for realtime high quality matting. *ACM Transaction on Graphics 26*, 3 (2007), 9:1–9:6.

36. Welch, W., and Witkin, A. Free-form shape design using triangulated surfaces. In *Proceedings SIGGRAPH 94)* (1994), 247–256.

37. Whitaker, R., Breen, D., Museth, K., and Soni, N. A framework for level set segmentation of volume datasets. In *Proceedings of International Workshop on Volume Graphics* (2001).

38. Yushkevich, P. A., Piven, J., Hazlett, H. C., Smith, R. G., Ho, S., Gee, J. C., and Gerig, G. User-guided 3d active contour segmentation of anatomical structures: significantly improved efficiency and reliability. *NeuroImage 31*, 3 (2006), 1116–1128.

39. Zhou, F., Zhao, Y., and Ma, K.-L. Parallel mean shift for interactive volume segmentation. In *International Workshop on Machine Learning in Medical Imaging*, Springer (2010), 67–75.

Appendix

The following instruction was given to the participants.

1. Right P1 (part of Posterior Cerebral Artery) and right Posterior Communicating Artery should be thoroughly depicted.
2. Bilateral Internal Carotid Arteries should be thoroughly depicted within the range of the dataset.
3. Basilar Artery should be thoroughly depicted.
4. All the signals that are not vascular origin should be removed.