The Visual Computer manuscript No.
(will be inserted by the editor)

Xiaoru Yuan · Nan Zhang · Minh X. Nguyen · Baoquan Chen

Volume Cutout

Abstract We present a novel method for cutting out 3D volumetric structures based on simple strokes that are drawn directly on volume rendered images. The freehand strokes roughly mark out objects of interest and background. Our system then automatically segments the regions of interest and refines their boundaries in the rendered image. These 2D segmentation results provide constraints for 3D segmentation in the volume dataset. The objects of interest are then efficiently cut out from the volume via a combination of our novel two-pass graph cuts algorithm and the pre-computed over-segmentation. Our method improves traditional, fully automatic segmentation by involving human users in the process, yet minimizing user input and providing timely feedback. Our experiments show that our method extracts volumetric structures precisely and efficiently while requiring little skill or effort from the user.

Keywords Segmentation · Volume Editing · Visualization · Volume Rendering · Interaction

1 Introduction

Volume segmentation that identifies individual objects in volumetric datasets has been an area of extensive research. In medical imaging, an accurate volumetric segmentation of organ structures is needed for diagnostic and treatment planning applications. Volume segmentation is also essential for effective volume visualization. Many state-of-the-art volume visualization tasks involve exploring the substructures of volume datasets and require volume classification information. For example, two-level volume rendering [7,9] selectively applies different rendering techniques to different objects. More importantly, an effective volume segmentation method can facilitate interactive data exploration. With the ability to interactively identify and segment interesting structures or objects, a visualization system can adaptively apply importance-driven rendering [28] or deformation [16] to better visualize objects of interest.

Various sophisticated volume segmentation algorithms have been proposed. However, segmentation remains a challenge because of the semantic gap between the high-level human perception and the low-level features that the segmentation is based on. Automatic segmentation of objects from volume data is fundamentally difficult due to the ambiguity between object boundaries. To overcome the difficulty of automatic segmentation, human supervision/intervention becomes essential. Traditional manual segmentation processes involve drawing object contours around cross-sectional views and linking these cross-sections together. However, this manual procedure can be extremely tedious, very time consuming and requires user experience and skill. The objective, therefore, is to develop an efficient method that supports high level user interaction which is less laborious than operating on each slice individually. The method should provide instant feedback so that additional user interaction can be specified to refine the results. Our aim is to facilitate interactive data exploration and visualization through such interactive object selection. Offline segmentation does not accommodate this process.

In this paper, we present a novel method for cutting out 3D volumetric structures based on simple strokes drawn directly on rendered images. The freehand strokes placed by the user roughly mark out the objects of inter-

Support for this work includes University of Minnesota Computer Science Department Start-up funds, NSF ACI-0238486 (CAREER), and the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAD19-01-2- 0014. Its content does not necessarily reflect the position or the policy of this agency, and no official endorsement should be inferred.

Xiaoru Yuan, Nan Zhang, Minh X. Nguyen, Baoquan Chen
Department of Computer Science and Engineering,
University of Minnesota at Twin Cities,
4-192 EE/CS Bldg, 200 Union Street S.E.,
Minneapolis, MN, 55414 USA.
Tel.: +1-612-625-0365
Fax: +1-612-625-2002
E-mail: {xyuan,nanzhang,mnguyen,baoquan}@cs.umn.edu

est and the background. Our system then automatically segments the regions of interest and refines their boundaries on the rendered images based on a 2D graph cuts algorithm. The results from image segmentation provide constraints for volume segmentation. The object of interest is then efficiently cut out from the volume via a combination of our two-pass graph cuts algorithm and pre-computed over-segmentation for acceleration. Our method improves traditional fully automatic segmentation by involving human users in the process putting humans in the loop, yet minimizing user input. Our experiments show that our method extracts objects precisely and efficiently while requiring little skill or effort from the user.

The remainder of this paper is organized as follows. In Section 2, we briefly review related work. We give an overview of our method in Section 3, and describe several important steps of our approach in detail in Sections 4, 5, 6, and 7. We then discuss results and limitations in in Section 8. Finally we present our conclusions in Section 9.

2 Related Work

The volume cutout method that we propose in this paper is extended from 2D image segmentation and vision techniques with added interaction. Our method performs both 2D and 3D segmentations in two consecutive stages. In the following, we briefly review several state-of-the-art segmentation methods, emphasizing on those that support efficient user interaction.

Magic Wand [12] is a selection tool widely used by many image editing software. It collects color statistics from a user-specified image region and segments a subimage region, where colors fall within some tolerances of the statistics. Finding suitable tolerance values depends on individual image context. This method is prone to error when the foreground and background pixels overlap in their color space distributions. Intelligent paint [22] is a region-based interactive segmentation technique based on Toboganning [4]. It extracts regions of interest from the image background using paint strokes. Intelligent scissors [18,19] is a boundary-based method. It computes a minimum-cost path via a shortest-path graph algorithm between user-specified boundary points. Intelligent scissors and intelligent paint allow the user to quickly and accurately select objects of interest [17,20]. Tan and Ahuja [25] develop a tool for selecting objects using freehand sketches based on the decomposition of the image segmentation into a triangulation. This triangulation captures the adjacency information of the segments as well as the shape of the segment boundaries.

An image segmentation or classification problem can be posed as a 2D graph cuts problem. Recently, several interactive methods for 2D image segmentation with user interaction have been developed [1,14,23] based on the graph cuts algorithm. Graph cuts [6,2,3] is a combinatorial optimization technique using global optimal pixel labeling of an object and its background, which cab be computed by max-flow/min-cut algorithm. Grabcut [23] extends graph cuts by introducing an iterative scheme using graph cuts for intermediate steps. The user draws a rectangle around the object of interest. Each iteration estimates color statistics of the object and the background and applies graph cuts to compute a refined segmentation. More recently, graph cuts algorithm has been applied to interactive video cutout [29].

For volume segmentation, most segmentation methods are built on directly applying 2D segmentation techniques to volume slices. Hastreiter and Ertl [8] extend the 2D intelligent scissors to volume segmentation by considering inter-slice relations and using a 3D filter to compute the local cost. Huang and Ma [11] apply region-growing techniques to assist the user in defining and locating features of interest in a volume. Region growing starts from a seed point which is picked by the user on volume slices. Tzeng et al. [26] further support user input directly on a cross sectional plane and use neural networks to compute the high dimensional classification. Tzeng and Ma [27] allow the user to work in the cluster space to specify transfer functions of arbitrary dimensions and operate directly on the classification and visualization results. Boykov and Jolly [2] also extend the graph cuts algorithm to volumes by marking foreground and background seeds on volume slices.

All the above methods work on volume cross-sections. It is difficult for the user to have a global view of the objects and the contextual volume based on one single cross-section. Owada et al. [21] recently develop a method, named volume catcher, to directly segment a volume by 2D free-form sketching on the rendered image. Their system computes the 3D locations of the user-specified 2D stokes based on the assumption that the user always traces the silhouette of the object of interest. Our method is inspired by this method. However, the user does not necessarily need to follow the object contour in our case. Rather, he/she can draw any strokes as long as these strokes indicate the object and background regions. This alleviates the user's commitment when sketching. Hence, it is more practical and efficient.

3 Overview

Our method is a combination of 2D and 3D graph cuts algorithms based on 2D user input. The processing pipeline is shown in Figure 1. Given an input volume, we first apply a watershed based over-segmentation. After rendering an image of the volume, we apply similar 2D over-segmentation to the image. Then, the user directly marks out foreground and background objects using simple strokes on the image. Based on the over-segmentation of the image, our system separates background and fore-

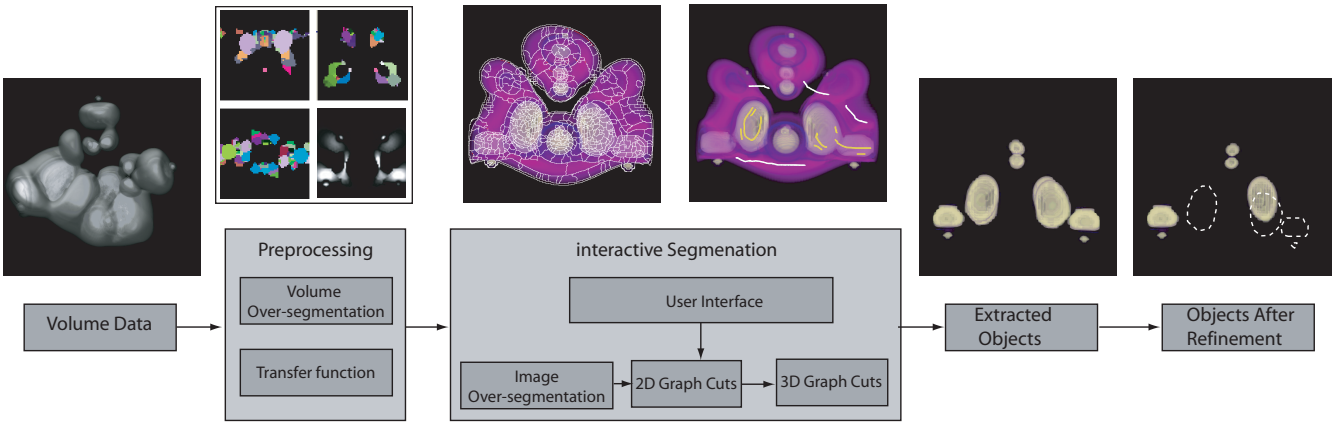
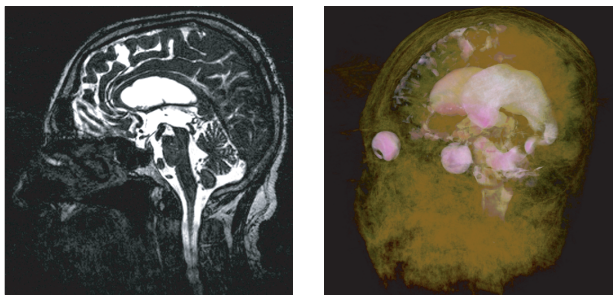


Fig. 1 Volume cutout pipeline: input volume data is first over-segmented and assigned a transfer function during the pre-processing stage. Based on the volume rendered image, 3D segmentation is obtained by a combination of 2D and 3D graph cuts algorithms initiated by 2D user input. The user can further select a subset of the extracted objects (In the last image, dotted lines indicate removed objects.).

ground regions based on a graph cuts algorithm. From image segmentation, 3D background seeds are generated for further volume segmentation. The objects of interest are then efficiently cut out from the volume by our two-pass graph cuts algorithm for 3D. Since only the 3D background seeds are specified, the first pass of the graph cuts algorithm returns a subset of the object (foreground) seeds. Those seeds are initial conditions passed to the next pass of the graph cuts algorithm, which achieves the final volume segmentation.

As discussed in the previous sections, one major difference of our segmentation approach from most existing methods is that we directly operate on rendered images. As shown in Figure 2, a single slice provides very limited information on the structure of a dataset. When the full volume is rendered, the overall structure of the data becomes visible. This provides an "overview" of the data, on top of which the user can select objects of interest or remove unwanted objects.



(a) (b)

Fig. 2 Comparison of information conveyed in (a) one cross-section of an MRI CISS dataset of a human head volume, and (b) a volume rendered image.

4 Preprocessing

In the preprocessing stage, we perform over-segmentation on the input volumetric datasets to accelerate the subsequent operations. We apply a 3D version of watershed over-segmentation. As in the Tobogganing method [4, 30], we over-segment the volume into small regions by sliding in the derivative terrain. Given the gradient magnitude of a volume, each voxel determines a slide direction by finding the voxel in its neighborhood with the lowest gradient magnitude. Each voxel is linked to the smallest of its neighbors and eventually to points of local minima, which form the basis for a watershed region. The gradient magnitude used in our method is computed using multi-scale derivative of a Gaussian kernel $N_\sigma(x, y, z)$, which is a three-dimensional normal distribution with a standard deviation of σ. The multi-scale gradient magnitude is given as the maximum convolution value of the derivative of Gaussian kernels. Note that in on-the-fly segmentation, a similar over-segmentation is applied for acceleration for each rendered image to be segmented.

After the over-segmentation, we convert the volume into clusters of approximately 20 to 100 voxels. The later volume segmentation will directly operate on these clusters.

Before the interactive segmentation, a transfer function is chosen for the volume dataset. The objective of this transfer function design is to reveal important structures residing in the volume.

5 User Interaction and Image Segmentation

After preprocessing, our system renders the volume according to the specified transfer function. The user directly draws strokes on the rendered image to guide the

segmentation. As illustrated in Figure 3, the user draws strokes to indicate the background (white strokes) and foreground (yellow strokes) regions on the volume rendered image. The system returns an initial image segmentation using the 2D graph cuts algorithm and the precomputed over-segmentation. The information regarding 2D segmented regions is then sent to the next step of volume segmentations.

5.1 2D Over-segmentation

To improve the efficiency of the 2D graph cuts based segmentation, we adopt the acceleration method used in Lazy Snapping [14] which is based on pre-segmented pixel clusters, instead of pixels. The pixel clusters are segmented regions from the watershed segmentation. In Figure 3(a), the red lines are the boundaries of over-segmented regions. During the over-segmentation, segmented regions with zero alpha value are set as invalid. No clusters are generated for those regions.

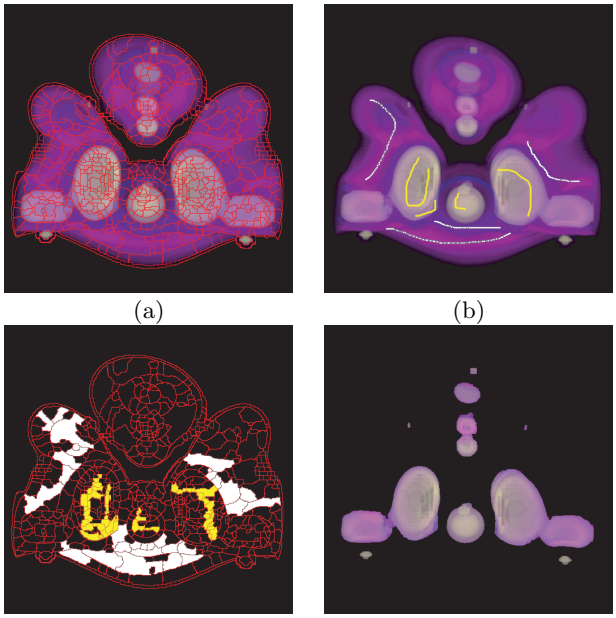


Fig. 3 (a) Over-segmentation of a 2D volume rendered image. (b) strokes placed by the user. (c) corresponding clusters covered by input strokes. (d) segmented 2D foreground regions.

5.2 2D Segmentation on Volume Rendered Image

Our algorithm uses graph cuts for both image segmentation and volume segmentation. In this section, we briefly describe the 2D graph cuts algorithm for image segmentation for completeness of explanation. Our extension on the graph cuts for 3D volume segmentation is discussed in the next section.

A 2D image can be treated as a graph $\mathcal{G} = \langle \mathcal{P}, \mathcal{N} \rangle$, where $\mathcal{P}$ is the set of all nodes and $\mathcal{N}$ is the set of all unordered pairs $\langle p, q \rangle$ connecting adjacent nodes. In the case of a 2D image, the nodes are pixels on over-segmented regions in the image and the arcs are adjacency relationships with connections between neighboring pixels. During user interaction, the user marks the foreground ($\mathcal{F}$) and background ($\mathcal{B}$) pixels. All other regions are assigned as unknown ($\mathcal{U}$).

The graph cuts algorithm divides the nodes in $\mathcal{P}$ into two groups (G_f and G_b) and assigns each node with a membership label value 0 or 1. For every $p_i \in \mathcal{F}$, $p_i \in G_f$, let its membership label be $x_i = 1$. For every $p_i \in \mathcal{B}$, $p_i \in G_B$, let its membership label be $x_i = 0$. The energy cost function can be defined as:

$$E(\mathcal{P}) = \lambda \sum_{p_i \in \mathcal{P}} R(p_i) + \sum_{p_i, p_j \in \mathcal{N}} B(p_i, p_j) \qquad x_i \neq x_j \quad (1)$$

The coefficient $\lambda \geq 0$ in Equation (1) specifies a relative importance of the region property term $R(p_i)$ versus the boundary property term $B(p_i, p_j)$.

Region property term $R(p_i)$. The regional term $R(p_i)$ assumes individual penalties for assigning pixel p to foreground ($x_i = 1$) or background ($x_i = 0$). For example, $R(p_i)$ may reflect how the intensity of pixel p fits into a known intensity model (e.g. histogram) of the object and background. The value of $R(p_i)$ is defined as follows:

if $p_i \in G_f$,

$$\begin{aligned} R(p_i) &= 0, & p_i \in \mathcal{F} \\ R(p_i) &= +\infty, & p_i \in \mathcal{B} \\ R(p_i) &= -ln(\tfrac{d^F_{p_i}}{d^B_{p_i} + d^F_{p_i}}), & p_i \in \mathcal{U} \end{aligned} \quad (2)$$

if $p_i \in G_b$,

$$\begin{aligned} R(p_i) &= +\infty, & p_i \in \mathcal{F} \\ R(p_i) &= 0, & p_i \in \mathcal{B} \\ R(p_i) &= -ln(\tfrac{d^B_{p_i}}{d^B_{p_i} + d^F_{p_i}}), & p_i \in \mathcal{U} \end{aligned} \quad (3)$$

$d^B_{p_i} = ||C_{p_i} - C_F||$ and $d^F_{p_i} = ||C_{p_i} - C_B||$ are the color distances of node p_i to the marked foreground ($\mathcal{F}$) and background ($\mathcal{B}$) clusters, respectively [14].

Boundary property term $B(p_i, p_j)$. This boundary term comprises the boundary properties of segmentation. Coefficient $B_{p_i,p_j} \geq 0$ should be interpreted as a penalty for the discontinuity between p_i and p_j. Normally, B_{p_i,p_j} is large when pixels p and q are similar (e.g. in their intensity) and close to zero when the two are very different.

$$B_{p_i,p_j} = \alpha exp(-\frac{||C_{p_i} - C_{p_j}||^2}{2\sigma^2}) \quad (4)$$

where α and σ are user defined parameters. If the color values are in $[0, 255]$, the values of α and σ range from 10 to 50.

The process of interactive image segmentation is illustrated in Figure 3. For a frame to be segmented, over-segmentation (Figure 3(a)) is first performed to accelerate segmentation. In Figure 3(b), the user places two types of strokes on the rendered images to indicate foreground (yellow strokes) and background (white strokes). The corresponding over-segmented cluster nodes covered by the user input (Figure 3(c)) are then set as initial foreground and background nodes for the graph cuts algorithm, which achieves the final segmentation (Figure 3(d)). Note that some structures not connected to the user marked foreground volume regions can also be segmented as foreground due to their similarity to the marked foreground regions.

6 3D Volume Segmentation

A volume segmentation problem can also be considered as a 3D labeling problem. If we are only interested in the foreground regions, it becomes a binary labeling problem and can be solved by the graph cuts algorithm. As stated in the previous section, graph cuts algorithm requires an initialization of foreground and background nodes. For a volume, they should represent seed voxels for the objects of interest and their contextual volume.

The 2D image segmentation only gives partial information needed for the 3D graph cuts algorithm. For any pre-segmented voxel cluster in the volume, if its projection is outside the 2D foreground region, then it must be in the background. Initial background nodes can thus be found. However, as it is also possible that background voxel clusters could be projected onto the 2D foreground region, foreground nodes cannot be directly identified. To address this issue, when the foreground $\mathcal{F}$ is unavailable for computing the region property term $R(p_i)$, we define a modified boundary property term $B(p_i, p_j)$ without specifying the foreground nodes in 3D:

if $p_i \in G_f$,

$$R(p_i) = +\infty, \quad p_i \in \mathcal{B} \qquad (5)$$
$$R(p_i) = -ln(\frac{K}{K+d_p^B}), \; p_i \in \mathcal{U}$$

if $p_i \in G_b$,

$$R(p_i) = 0, \quad p_i \in \mathcal{B} \qquad (6)$$
$$R(p_i) = -ln(\frac{d_{p_i}^B}{K+d_p^B}), \; p_i \in \mathcal{U}$$

where K is a constant which is the user estimation of average d_p^F. We develop a two-pass graph cuts algorithm. In the first pass, we use the above edge cost definition. It returns a subset of the object. We use the returned partial objects as foreground nodes and apply the regular graph cuts algorithm to achieve the final volume segmentation. Our two-pass 3D graph cuts algorithm is illustrated in Figure 4. For some situations, the graph cuts obtained in the first pass already give reasonable segmentation results.

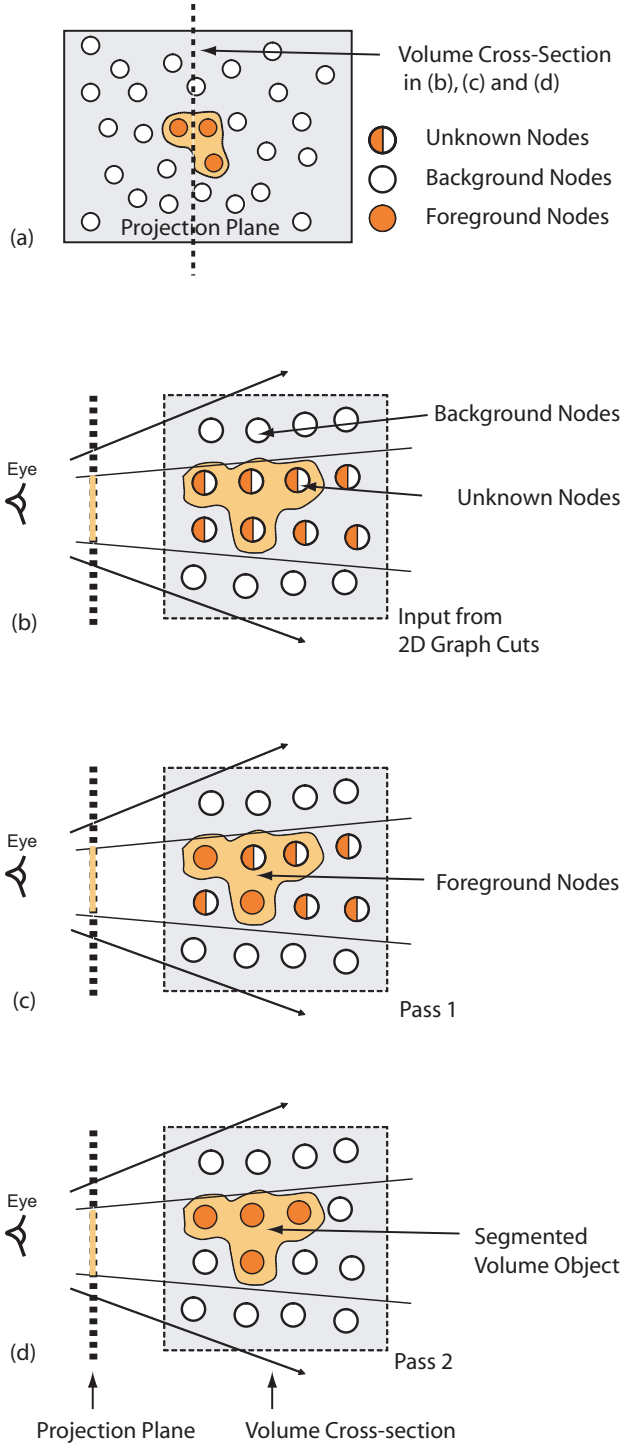


Fig. 4 Two-pass graph cuts algorithm for volume segmentation. (a) result of 2D segmentation on a volume rendered image. (b) 3D nodes in the volume are classified as background and unknown according to their projection to the 2D background and foreground regions. (c) The first pass graph cuts generates a subset of foreground seed nodes. (d) The second pass achieves the final segmentation.

7 Segmentation Refinement

In cases when the interactive segmentation procedure introduces unwanted objects, it is desirable to have an interactive tool for the user to edit them. For example, the user can keep a subset of extracted objects by removing unwanted ones. As illustrated in Figure 5, the user draws freehand curves to selectively remove unwanted objects from the segmentation result. Our system achieves this refinement by projecting segmented volume objects onto the current viewing plane and checking whether their projections fall into the region specified by the user. The user can also adjust viewing angles to get a best editing position.

8 Results and Discussion

We perform all our experiments on a Dell Precision 530 workstation with a single Intel Xeon 2.20G Hz CPU, 1GB RAM, 4X AGP motherboard and a Nvidia GeForce 6800 Ultra graphics card with 256MB memory. Our system is implemented using MFC, OpenGL, and Cg. The volume rendering window size is 512^2. The user interaction and image segmentation in 2D achieve interactive rates. The user normally does not notice any lag when placing strokes on the rendered volume image. The running time of volume segmentation depends on the complexity of the input volume and ranges from less than one tenth of a second for 64^3 volumes to around one minute for 256^3 volumes. Over-segmentation plays a very important role on the performance of volume segmentation. With a robust over-segmentation, it is possible to generate less a number of segments, resulting the run time performance improved.

Images in Figure 6 show the volume cutout results of several datasets: an Neghip ($64 \times 64 \times 64$), an MRI CISS human head ($256 \times 256 \times 124$), and a CT scanned foot ($160 \times 430 \times 183$). Images in the first column are one frame of volume rendering. The middle column shows the freehand sketches indicating foreground and background. Images in the right column show how the objects of interest are cut out from the original volume. It is evident that the user's input efforts for achieving these volume cutouts are simply a few strokes, hence are almost effortless. Also, the user can further refine object extraction results by removing unwanted objects.

It is worth pointing out that user does not need to mark out every foreground object; only representative ones are sufficient. Our method will extract all objects with appearance similar to the marked-out foreground objects, except those that are explicitly marked out as background objects. The last row of Figure 6 shows an example where objects that are likely to be marked as foreground, when marked out as background, are not extracted out. However, in certain situations, segmented regions that are not directly connected to the user specified regions might be undesirable. We are working on an algorithm that is able to enforce that only regions connected to the marked foreground be segmented out.

Even though our method has generally been proven effective, there are several issues that we need to address to improve its robustness. It is very important for the graph cuts algorithm to set appropriate parameters. For different datasets, such parameters vary. For noisy datasets, obtaining the best parameters is not trivial. It would be beneficial to have the system incorporate an adaptive graph cuts parameter setting scheme based on user interaction. It is also possible to incorporate some prior knowledge of the dataset into the graph cuts energy cost function to achieve better segmentation results.

There are some difficult situations which our method cannot handle well. For noisy data sets, our methods may include some extra objects into the results. The user can apply the refinement tool available in our system to remove unwanted objects. Such refinement is applied on the segmented result of the head dataset (Figure 6 2c). For very fuzzy object boundaries, it is difficult to obtain an accurate image segmentation. Changing transfer functions or applying some image processing operators may improve the image contrast and sharpen boundaries. Applying higher dimensional transfer functions [13] which consider gradient information will also improve the image quality. For a very complex object inside a volume, the seed points acquired from one rendered image may not be sufficient. In such cases, we could exploit images rendered from multiple view positions to better define the object boundary.

Because we segment the volume based on its rendered image, an ideal transfer function for our purpose should be able to display most internal structures simultaneously. Many methods have been proposed to automatically design transfer functions according to the data properties. Fujishiro et al. [5,24] automate transfer function design based on topology analysis for 3D field . He et al. [10] use genetic algorithms to create a population of transfer functions starting with a user given or random set of transfer functions. Marks et al. [15] propose design galleries to automatically generate a distribution of possible transfer functions and their corresponding rendering results. We are working on an automatic method for generating transfer functions that suits our requirements.

9 Concluding Remarks

We have presented a novel method for cutting out 3D volumetric structures based on simple strokes drawn directly on rendered images. Unlike most existing volume segmentation methods, our interactive method operates directly on 2D rendered images. We also propose a novel extension of the 2D graph cuts algorithm to overcome the difficulty of identifying foreground objects in the volume.

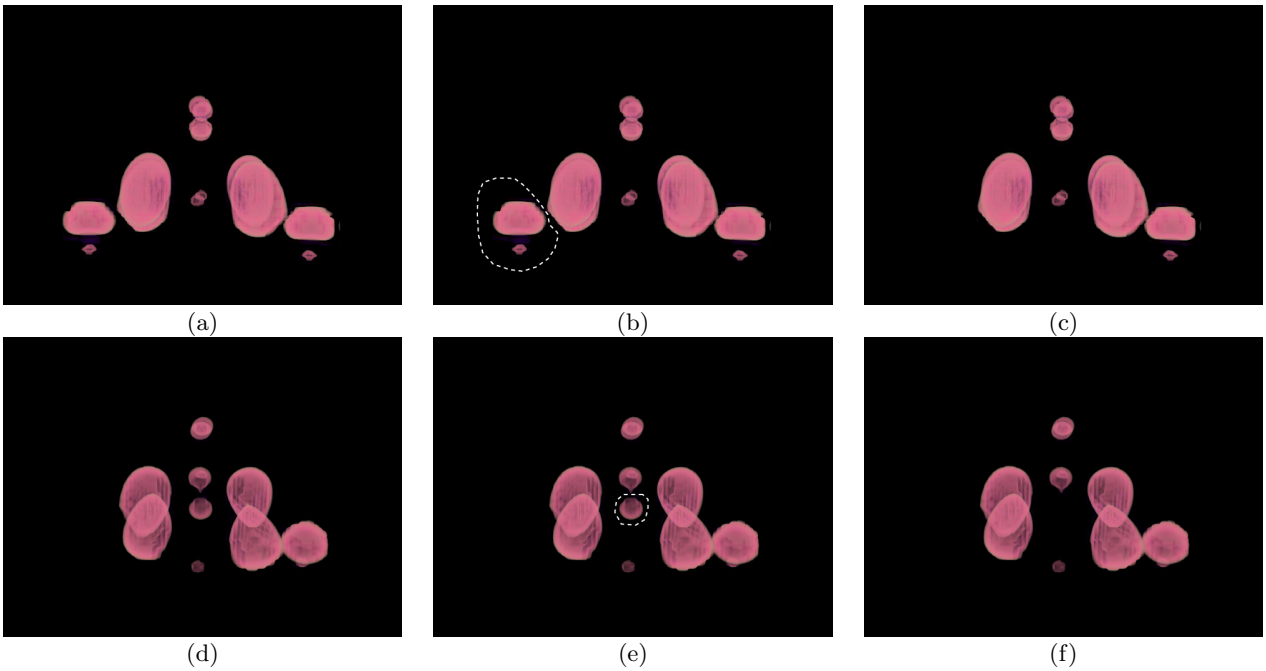


Fig. 5 Refining segmentation results. (a) a volume rendered image of the segmentation result. (b) the user draws an enclosed curve on the volume rendered image to remove unwanted structures. (c) volume objects after applying user refinement. (d) for further refinement, the user changes the viewing angle. (e) the user draws another enclosed curve on the volume rendered image to remove more unwanted structures. (f) volume objects after applying second user refinement.

Our method improves upon traditional automatic segmentation by involving human users in the process, yet minimizing user input and providing timely feedback. Our experiments show that the proposed method extracts objects precisely and efficiently while requiring little skill or effort from the user. We believe that intuitive volume cutout can be used to facilitate interactive data exploration and enhance visualization.

Acknowledgements We especially thank Yuri Boykov and Vladimir Kolmogoro for their help on graph cuts. We also thank Nathan Gossett and Amit Shesh for proofreading the paper; Hui Xu, Lijun Qu, and anonymous reviewers for helpful suggestions.

Neghip is VolVis distribution of SUNY Stony Brook. MRI CISS human head is from Dirk Bartz, VCM, University of Tübingen, Germany. Thanks to Michael Meissner for maintaining the volume data repository and providing downloads. The CT-scanned foot data set is downloaded from the web site of the Department of Radiology, University of Iowa.

References

1. Agarwala, A., Hertzmann, A., Salesin, D.H., Seitz, S.M.: Keyframe-based tracking for rotoscoping and animation. ACM Trans. Graph. **23**(3), 584–591 (2004)
2. Boykov, Y., Jolly, M.P.: Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In: International Conference on Computer Vision (ICCV '01), vol. I, pp. 105–112 (2001)
3. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Tran. on Pattern Analysis and Machine Interlligence **26**(9), 1124–1137 (2004)
4. Fairfield, J.: Toboggan contrast enhancement for contrast segmentation. In: 10th IEEE International Conference on Pattern Recognition, vol. I, pp. 712–716 (1990)
5. Fujishiro, I., Azuma, T., Takeshima, Y.: Automating transfer function design for comprehensible volume rendering based on 3D field topology analysis. In: Proceedings of IEEE Visualization, pp. 367–470 (1999)
6. Greig, D., Porteous, B., Seheult, A.: Exact map estimation for binary images. J. Roy. Stat. Soc. B. **51**, 271–279 (1989)
7. Hadwiger, M., Berger, C., Hauser, H.: High quality two-level volume rendering of segmented data sets on consumer graphics hardware. In: Proceedings of IEEE Visualization, pp. 301–308 (2003)
8. Hastreiter, P., Ertl, T.: Fast and interactive 3D-segmentation of medical volume data. In: Computer Graphics International, Visualization Minisymposium, pp. 78–85 (1998)
9. Hauser, H., Mroz, L., Bischi, G.I., Gröller, M.E.: Two-level volume rendering. IEEE Transactions on Visualization and Computer Graphics **7**(3), 242–252 (2001)
10. He, T., Hong, L., Kaufman, A., Pfister, H.: Generation of transfer functions using stochastic search techniques. In: Proceedings of IEEE Visualization, pp. 227–234 (1997)
11. Huang, R., Ma, K.L.: RGVis: Region growing based visualization techniques for volume visualization. In: Proceedings of Pacific Graphics Conference, pp. 335–333 (2003)
12. Incorp., A.S.: Adobe photoshop user guide (2002)
13. Kniss, J., Kindlmann, G., Hansen, C.: Multidimensional transfer functions for interactive volume rendering. IEEE Transactions on Visualization and Computer Graphics **8**(3), 270–285 (2002)
14. Li, Y., Sun, J., Tang, C.K., Shum, H.Y.: Lazy snapping. ACM Trans. Graph. **23**(3), 303–308 (2004)
15. Marks, J., Andalman, B., Beardsley, P.: Design galleries: A general approach to setting parameters for computer

graphics and animation. SIGGRAPH '97 pp. 389–400 (1997)
16. McGuffin, M.J., Tancau, L., Balakrishnan, R.: Using deformations for browsing volumetric data. In: Proceedings of IEEE Visualization, pp. 401–408 (2003)
17. Mortensen, E.N.: Vision-assisted image editing. Computer Graphics **33**(4), 55–57 (1999)
18. Mortensen, E.N., Barrett, W.A.: Intelligent scissors for image composition. In: Computer Graphics (SIGGRAPH '95), pp. 191–198 (1995)
19. Mortensen, E.N., Barrett, W.A.: Toboggan-based intelligent scissors with a four parameter edge model. In: Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR '99), Vol. II, pp. 452–458 (1999)
20. Mortensen, E.N., Reese, L.J., Barrett, W.A.: Intelligent selection tools. In: Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR '00), vol. 2, pp. 776–777 (2000)
21. Owada, S., Nielsen, F., Igarashi, T.: Volume catcher. In: Proceedings of the 2005 symposium on Interactive 3D graphics and games (I3D '05), pp. 111–116 (2005)
22. Reese, L.: Intelligent paint: Region-based interactive image segmentation. Master thesis, Department of Computer Science, Brigham Young University, Provo, UT (1999)
23. Rother, C., Kolmogorov, V., Blake, A.: "GrabCut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. **23**(3), 309–314 (2004)
24. Takahashi, S., Takeshima, Y., Fujishiro, I.: Topological volume skeletonization and its application to transfer function design. Graph. Models **66**(1), 24–49 (2004)
25. Tan, K.H., Ahuja, N.: Selecting objects with freehand sketches. In: Proceedings IEEE International Conference on Computer Vision (ICCV '01), vol. 1, pp. 337–344 (2001)
26. Tzeng, F.Y., Lum, E., Ma, K.L.: A novel interface for higher-dimensional classification of volume data. In: Proceedings of IEEE Visualization, pp. 505–512 (2003)
27. Tzeng, F.Y., Ma, K.L.: A cluster-space visual interface for arbitrary dimensional classification of volume data. In: Proceedings of Joint IEEE/EG Symposium on Visualization (VisSym '04), pp. 17–24 (2004)
28. Viola, I., Kanitsar, A., Gröller, M.E.: Importance-driven volume rendering. In: Proceedings of IEEE Visualization, pp. 139–145 (2004)
29. Wang, J., Bhat, P., Colburn, A., Agrawala, M., Cohen, M.F.: Interactive video cutout. to appear in ACM SIGGRAPH (2005)
30. Yao, X., Hung, Y.P.: Fast image segmentation by sliding in the derivative terrain. In: SPIE Proc. Intelligent Robots and Computer Vision X: Algorithms and Techniques, vol. 1607, pp. 369–379 (1991)

Xiaoru Yuan is a PhD candidate in Computer Science at the University of Minnesota at Twin Cities. He received a BS degree in Chemistry and a BA degree in Law from Peking University, China, in 1997 and 1998, respectively. His primary research interests include nonphotorealistic rendering and its application in visualization, high dynamic range imaging and rendering, volume visualization, illustrative visualization, and computational geometry.

Nan Zhang is a research associate of Army High Performance Computing Research Center (AHPCRC), University of Minnesota at Twin Cities. He received the BS degree in computer science from Xi'an Jiaotong University, China, in 1992, the MS degree in computer science from Tsinghua University, China, in 1995, and the PhD degree in computer science from Stony Brook University, USA, in 2004. Prior to his PhD studies, he had some years of experience in networking engineering. His research interests include volume visualization, medical imaging, and terrain rendering.

Minh X. Nguyen is a PhD candidate in Computer Science from the University of Minnesota, Twin Cities. He obtained his BS degree in Computer Science from Ho Chi Minh City University (now is Ho Chi Minh City University of Natural Sciences), Vietnam in 1994. Before going back to graduate study, he was a lead programmer at a cartography company (DolSoft Co. Ltd., Ho Chi Minh City, Vietnam) specializing in GIS and a junior researcher at the Institute of Applied Mechanics, Ho Chi Minh City, Vietnam. His research interests are interactive 3D visualization with emphasis on hardware support, geometry modeling, scientific visualization and computational geometry.

Baoquan Chen is an assistant professor of Computer Science and Engineering at the University of Minnesota at Twin Cities; there he is also a member of the Digital Technology Center and Digital Design Consortium. His research interests generally lie in computer graphics and visualization, focusing specifically on 3D data acquisition, illustrative rendering and visualization and interactive techniques. His research is supported by National Science Foundation, Army Research, Microsoft Research, and private donation. Chen is the recipient of the Microsoft Innovation Excellence Program 2002, the NSF CAREER award 2003, and McKnight Land-Grant Professorship of the University of Minnesota 2004-2006. Chen has served, or is serving on program and paper committees of several conferences in the field, most notably IEEE Visualization (program co-chair 2004, general co-chair 2005/2006), and Symposium on Point Based Graphics (2004/2005, papers co-chair 2006). Chen received an MS in Electronic Engineering from Tsinghua University, Beijing (1994), and a second MS (1997) and then PhD (1999) in Computer Science from the State University of New York at Stony Brook. For more information see http://www.cs.umn.edu/∼baoquan.

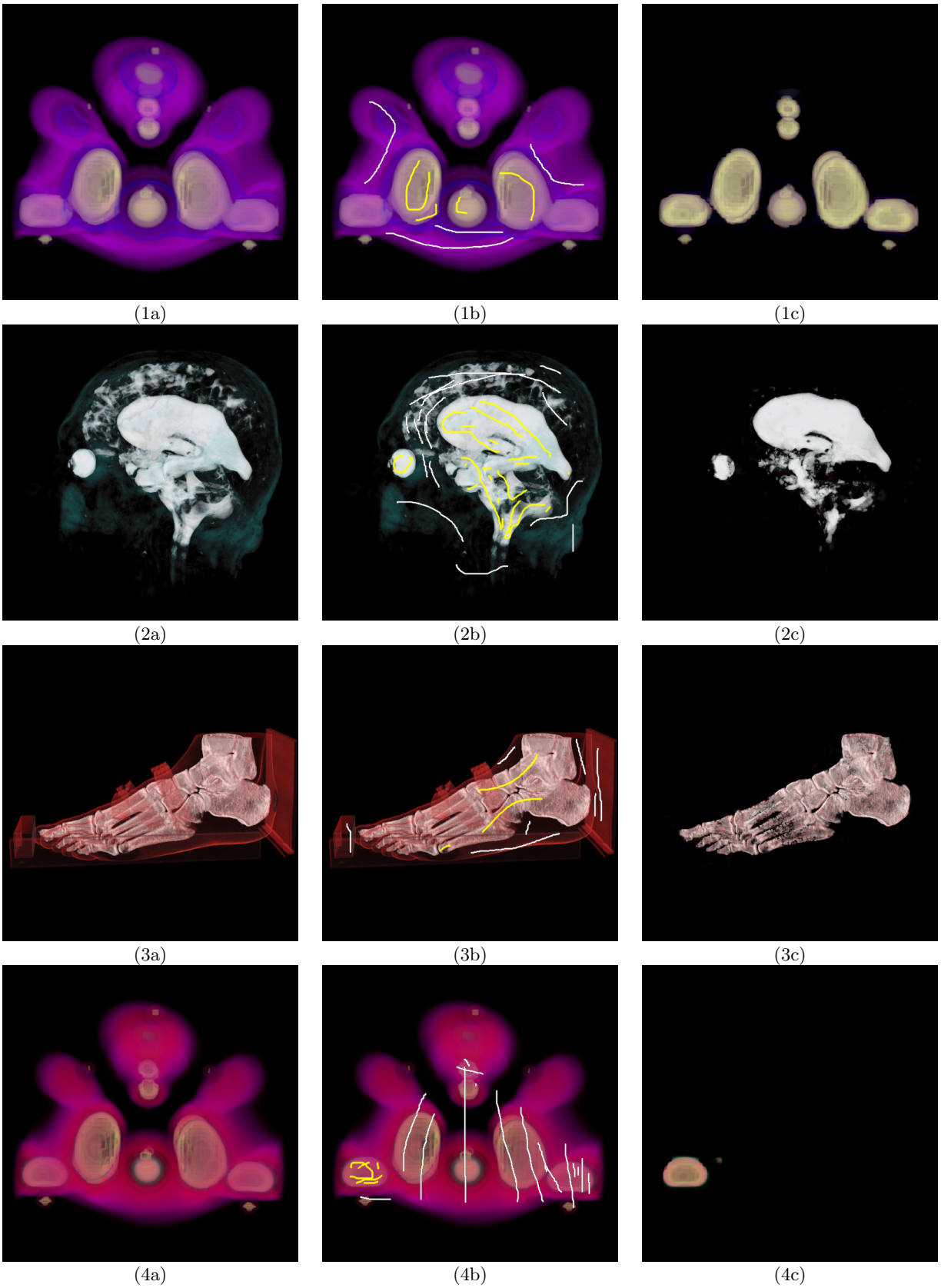


Fig. 6 Results of volume cutout on several volume datasets: an Neghip (the first and fourth rows), an MRI CISS human head (the second row) and a CT scanned foot (the third row). (left column) 2D volume rendered images. (middle column) user supplied strokes indicating foreground and background. (right column) extracted objects.