Contour Gradient Tree for Automatic Extraction of Salient Object Surfaces from 3D Imaging Data*

Bong-So SOHN†a), Member

SUMMARY Isosurface extraction is one of the most popular techniques for visualizing scalar volume data. However, volume data contains infinitely many isosurfaces. Furthermore, a single isosurface might contain many connected components, or contours, with each representing a different object surface. Hence, it is often a tedious and time-consuming manual process to find and extract contours that are interesting to users. This paper describes a novel method for automatically extracting salient contours from volume data. For this purpose, we propose a contour gradient tree (CGT) that contains the information of salient contours and their saliency magnitude. We organize the CGT in a hierarchical way to generate a sequence of contours in saliency order. Our method was applied to various medical datasets. Experimental results show that our method can automatically extract salient contours that represent regions of interest in the data.

key words: volume visualization, computer graphics, isosurface

1. Introduction

Scientific and medical imaging techniques such as computed tomography and magnetic resonance imaging often generate high-resolution volume data in the form of a scalar function sampled on 3D rectilinear grids. Isosurface extraction is one of the most popular techniques for visualizing the volume data. An isosurface (also called an isocontour) is a level set surface defined as \( I(w) = \{ (x,y,z) | F(x,y,z) = w \} \), providing a geometric surface of objects or structures specified by an isovalue \( w \) in the volume data function \( F \).

However, there are an infinite number of isosurfaces defined in an image. Furthermore, even a single isosurface can have many connected components, or contours, with each representing a different object surface in complex imaging data such as medical images. Therefore, the problem of effectively finding and extracting a contour that represents a region of interest becomes important. For example, a mammogram image in Fig. 1 (a) shows an x-ray picture of breast tissue. Figure 1 (b) displays a set of isocontours where isovalue contours are regularly sampled. Automatically locating a contour that represents a mass, as shown in Fig. 1 (c), is proven to be useful for a radiologist because the size, shape, and margins of the mass help the radiologist in evaluating the likelihood of breast cancer [5].

There are well known methods such as the contour spectrum [1] that help users to find significant isosurfaces. However, it cannot distinguish individual contours in an isosurface. Topological saliency [4] and volume saliency [6] can be used to find salient areas, but they cannot generate salient surfaces from volume data. A contour tree (CT) is useful for visualization of contours [3]. The CT provides a topological structure of contours, and it is used as an interactive user interface to select and extract individual contours [8]. However, it cannot directly detect salient contours.

In this paper, we describe a novel method for automatically extracting salient contours from volumetric imaging data. The main contributions of this paper are as follows:

- To define a salient contour and its saliency magnitude.
- To propose a contour gradient tree (CGT) that contains information of the salient contours and their saliency.
- To use CGT for automatic extraction of salient contours (i.e. we can extract any \( k \)-th most salient contour).

We define the salient contour as a contour that has a locally maximal gradient value, meaning its gradient integral over the contour surface is bigger than the gradient values of adjacent contours. For the purpose of extracting such salient contours, we propose a CGT where contours with a locally maximal gradient are represented as leaf nodes and adjacent contours are connected via an edge. This definition of a salient contour may generate an excessive number of similar salient contours in real imaging data (which often contains a significant amount of noise). For this reason, we define a saliency metric to assign high saliency values to distinctive salient contours. We hierarchically organize a CGT to mea-
sure the saliency values and generate a sequence of salient contours with saliency order information as output.

We applied our method to various medical imaging datasets. Experimental results demonstrate that our automated method successfully extracted regions of interest in the data with minimal user interaction.

2. Salient Contour and Its Saliency Magnitude

Image function intensities tend to rapidly change at an object boundary in an image. Hence, a contour with a locally maximal gradient in an image is likely to represent an object boundary. Since the gradient values are not consistent along the points of a contour surface, the gradient $\text{Grad}(C)$ of a contour $C$ is computed as the gradient integral along the contour surface.

$$\text{Grad}(C) = \int_C |\nabla F| \text{d}C$$ (1)

We define a salient contour as a contour that has a locally maximal gradient integral value. A gradient integral value of a contour is assumed to be a local maximum if the value is bigger than those of adjacent contours.

In order to determine salient contours, adjacency and gradient information of each contour is necessary. We regularly sample the entire range of a function for isovalues and obtain a set of isocontours based on the isovalues. A contour, a connected component of an isosurface, is treated as an independent object. Each contour has two types of adjacent contours, inside-adjacent contours and outside-adjacent contours. A contour is considered as a salient contour if its gradient is greater than the gradients of inside- and outside-adjacent contours. Figure 2 shows an example of determining salient contours from an input height function. The contours colored in red are considered salient because their computed gradient values are greater than those of adjacent contours.

In addition to determining salient contours, we need to assign saliency magnitude to each salient contour. Gradient is a reasonable measure of saliency. However, we cannot simply set the gradient value as the saliency magnitude, as there would be many similar salient contours in practice (e.g. contours with gradient 17 and 16 in Fig. 2). Therefore, we need to consider the distance (i.e. dissimilarity) between salient contours as well as their gradient values to assign high saliency value to distinctive salient contours. For this reason, we define saliency magnitude of a contour as the depth of a path through contour adjacency from the given salient contour to the nearest salient contour with a bigger gradient value. This constraint prevents the designation of high saliency to nearly identical salient contours. The depth of the path is defined as the difference between the gradient values of the given salient contour and the contour with the smallest gradient on the path.

3. Proposed Approach for Automatic Extraction of Salient Contours

Our method is summarized in Fig. 3. A sampled contour tree (SCT) is constructed from input volume data. Each node and edge of the tree represent a contour and its adjacency information, respectively. We compute adjacency and gradient information for each contour that is represented as a node of the SCT. Using the information, we then construct a CGT and compute the saliency of each contour to output a sequence of salient contours with saliency order information. Noise in scanned imaging data often has a negative influence on quality and accuracy of the result (i.e. computed contour surface and CGT). Smoothing techniques such as bilateral filtering can be applied to input data as preprocessing to reduce the effect of the noise. The detail of each step is described as follows.

3.1 Construction of Sampled Contour Tree

An SCT is constructed to compute inside- and outside-adjacency information for each contour, which is necessary to determine salient contours. We first compute a contour tree (CT) using a well known method [2]. As an isovalue changes from the maximum to minimum value, contours are created, merged, split, and destroyed. The topological events are represented in the form of a tree. Each vertex
of the CT represents a critical point of an input function $F$ where the topological events occur. Each edge of the CT represents a maximal set of continuous contours that do not contain critical points. Vertical coordinates of CT vertices are determined to be the function values defined on the corresponding critical points. A point on a CT edge corresponds to a contour. Therefore, if we draw the CT and horizontal lines of regularly sampled isovalues, the intersection points become nodes of the SCT as shown in Fig. 2. We connect the nodes of neighboring horizontal lines to form an edge if they are connected through CT edges between the lines. In this way, we can construct the SCT from the CT. Each node and edge of the SCT represents a contour and adjacency between contours that are represented by two ending nodes of the edge, respectively.

3.2 Computation of Contour Gradient

The gradient value of each contour is also necessary to determine salient contours. To compute an approximated value of the gradient defined in Eq. (1), we extract a triangular mesh for each contour surface, compute the gradient for each triangle of the mesh, and obtain the gradient value of the contour by adding the gradient for each triangle. The gradient of a triangle is approximated by averaging the gradient values computed at three vertices of the triangle and multiplying the average with an area of the triangle.

For efficient mesh extraction and gradient computation, we use hybrid parallelization of multi-core CPUs and many-core GPUs [7]. A multi-threaded propagation algorithm that is performed on multi-core CPUs extracts active cells (i.e. cells containing a desired contour surface) from a seed cell generated by CT. Surface geometry and its gradient are quickly computed on many-core GPUs from the active cells.

3.3 Construction of Contour Gradient Tree

CGT is similar to CT in that they both summarize all possible contours defined in an input function. CT illustrates how contours change topology as an isovalue decreases from the maximum value. CGT, on the other hand, contains information regarding contours that have a locally maximal gradient (i.e. salient contours), and how those contours evolve and merge as a gradient value decreases.

The input of a CGT construction algorithm are nodes and edges of SCT. As preprocessing, the nodes are sorted in decreasing order of gradient values. We take a node one by one from the sorted array of nodes and insert it into CGT as node $x$. If a node that is adjacent to the taken node was already inserted to CGT as node $y$, insert an edge connecting the node $x$ to the last node (i.e. the node with the smallest gradient) of the connected node group that contains the node $y$ in CGT. Repeat this process until all nodes are processed. Finally, we delete all regular nodes of CGT that have one up-degree and one down-degree. A formal algorithm description is given in Algorithm 1.

### Algorithm 1 Construction of Contour Gradient Tree

**Input** - nodes, edges: nodes of SCT, edges: edges of SCT

**Output** - CGT

As preprocessing, sort nodes in decreasing order of gradient.

1. $CGT \leftarrow$ NewTree();
2. $UF \leftarrow$ NewUF(); // Create Union-Find Structure $UF$ [2]
3. for $i = 0$ to $n - 1$
4. \hspace{1em} $AddNode(CGT,i)$;
5. \hspace{1em} if IsLocalMax(nodes,$i$) then NewSet($UF,i$);
6. \hspace{1em} for each edge nodes,$nodes_j$ with $j < i$
7. \hspace{2em} $t' \leftarrow$ Find($UF,j$);
8. \hspace{2em} $f' \leftarrow$ Find($UF,f$);
9. \hspace{2em} if $t' \neq f'$ then AddEdge($CGT,t',f'$);
10. \hspace{1em} Union($UF,t',f'$);
11. DeleteRegularNodes($CGT$);
12. return $CGT$;

Upper leaf nodes of CGT represent salient contours. We need to assign a saliency value to each salient contour. For this purpose, we decompose CGT by iteratively removing the most insignificant leaf edges to generate a hierarchical form. This hierarchical form represents CGT as a union of tree edges where each edge corresponds to a salient contour that is represented by an upper leaf node of the edge. Saliency magnitude of a salient contour is computed as the persistence of the corresponding CGT edge. **Persistence** of an edge is computed as the difference in gradient values defined on two end nodes of the edge. The steps of this algorithm are described as follows:

1. All leaf edges of CGT are inserted into a priority queue.
2. The leaf edge with the smallest persistence is removed from the priority queue.
3. CGT and the priority queue are updated by merging the two edges sharing a vertex that was also shared by the removed edge.

![Fig. 4](image-url) Step-by-step construction of CGT and its hierarchical representation from adjacency/gradient information of data in Fig. 2. Saliency magnitude of salient contours 16, 13, 17, and 19 are computed as the persistence (i.e. difference of gradient values on two end nodes of an edge) of $e_1$, $e_2$, $e_3$, and $e_4$, which are 2, 3, 6, and 13, respectively.
4. For each $k$-th iteration of step 2 and 3, a leaf edge $e_k$ that was removed from the tree is generated. The iteration is repeated until no edge is left in the queue.

The union of all $e_k$ ($k = 1, \ldots, n$) forms the hierarchical representation of CGT. Figure 4 depicts the step-by-step process of CGT construction and hierarchical representation from the example height function of Fig. 2.

3.4 Automatic Extraction of Salient Contours

As a result of Sec. 3.3, the hierarchical form of a CGT is represented as a union of edges $e_1, e_2, \ldots, e_n$, which are generated in increasing order of saliency, where $\text{persistence}(e_i) < \text{persistence}(e_j)$ for $i$ and $j$ that are $i < j$. This means $e_{n-k+1}$ has the $k$-th biggest persistence. Users can extract any $k$-th most salient contour by extracting the contour surface that corresponds to the upper leaf node of $e_{n-k+1}$.

4. Experimental Results

We implemented and tested our algorithm on a desktop PC equipped with an Intel i7 (3.0 GHz) CPU, 8 GB main memory, and an NVIDIA GeForce 480GTX graphics card with 1.5 GB video memory.

We tested three datasets: a head MRI, a computed tomography of a human tooth, and a 2D x-ray of breast tissue. The number of sample isovalues for constructing SCTs was set to be 128. Figure 5, Fig. 6, and Fig. 1 show the results of automatically extracting salient contours from the test data. The number of nodes in CGT was 126, 264, and 168, respectively. As shown in Fig. 5(a), the head MRI dataset contains many contours that represent brain tissues, eyeballs, and ventricles. Using our method, we automatically extracted only the ventricle contour that is the main feature of the data. This allows us to visualize and analyze the ventricle surface without being distracted by other features. As shown in Fig. 6, our method automatically extracted the surfaces representing main structures of the tooth dataset, such as enamel, dentin, and pulp. Figure 1 implies our method can also be applied to a 2D image successfully. Table 1 summarizes the processing time for each step of our method.

Above results indicate that our method has a significant advantage compared to previous methods for contour extraction such as [3] and [7] in the sense that we provided a way to automate the manual work of locating salient contours, which is required by the previous methods.

5. Conclusion

We devised a novel method to automatically extract salient contours with saliency order information from 3D volume data. For this purpose, we proposed a CGT that contains the information of salient contours and their saliency magnitude. Our work is useful for automatically finding, displaying, and analyzing regions of interest in volume data.

References

[1] C.L. Bajaj, V. Pascucci, and D.R. Schikore, “The contour spectrum,” Proceedings of IEEE Visualization, pp.167–173, 1997.
[2] H. Carr, J. Snoeyink, and U. Axen, “Computing contour trees in all dimensions,” Computational Geometry: Theory and Applications, vol.24, no.2, pp.75–94, 2003.
[3] H. Carr, J. Snoeyink, and M. van de Panne, “Flexible isosurfaces: Simplifying and displaying scalar topology using the contour tree,” Computational Geometry: Theory and Applications, vol.43, no.1, pp.42–58, 2010.
[4] H. Doraiswamy, N. Shivashankar, V. Natarajan, and Y. Wang, “Topological saliency,” Computers & Graphics, vol.37, no.7, pp.787–799, 2013.
[5] B.-W. Hong and B.-S. Sohn, “Segmentation of regions of interest in mammograms in a topographic approach,” IEEE Transactions on Information Technology in Biomedicine, vol.14, no.1, pp.129–139, 2010.
[6] E. Shen, Y. Wang, and S. Li, “Spatiotemporal volume saliency,” Journal of Visualization, published online in April 2015.
[7] B.-S. Sohn, “Hybrid parallel extraction of isosurface components from 3d rectilinear volume data,” IEICE Transactions on Information and Systems, vol.E94-D, no.12, pp.2553–2556, 2011.

[8] B.-S. Sohn and C. Bajaj, “Time-varying contour topology,” IEEE Transactions on Visualization and Computer Graphics, vol.12, no.1, pp.14–25, 2006.