Multi-material Volume Segmentation for Isosurfacing Using Overlapped Label Propagation

Takuma Niizaka Yutaka Ohtake Takashi Michikawa Hiromasa Suzuki

The University of Tokyo
4-6-1 Komaba, Meguro-ku, Tokyo, 153-8904, JAPAN
{ niizaka | yu-ohtake | michi | suzuki }@den.rcast.u-tokyo.ac.jp

ABSTRACT

This paper proposes a simple and efficient method for segmenting multi-material volumes. Unlike standard image segmentation techniques, this approach aims to extract accurate isosurfaces representing the boundary surfaces between different materials. The segmented volumes obtained in this way meet the requirements of the topological correctness and geometrical accuracy for isosurfacing. The approach involves three steps: first, voxels far from the boundary surfaces are labeled; then, the labels are propagated with overlapping to ascertain which materials meet near the boundary surfaces; finally, an appropriate material is selected from among the multiple labels assigned to a voxel with adaptive thresholding. Since the method consists of iterative local operations, it is easy to parallelize for fast computation. The efficiency of the technique is demonstrated here using CT scanned objects for complex shapes.

Keywords:

Volume segmentation, multi-material, adaptive thresholding, isosurface extraction.

1 INTRODUCTION

3D images (referred to here as *volumes*) consisting of scalar fields sampled on a regular grid are obtained as the output of CT/MRI scanning. Due to technological advances in scanning, volume data is widely used for many applications including medical imaging, computer graphics and industrial applications. In this research, we considered the problem of accurately extracting the boundary surfaces of CT scanned objects. This issue was mainly focused on industrial applications in which the required level of accuracy is too high for the voxel size.

Image segmentation methods are generally used to extract the boundary surfaces. So far, many segmentation approaches have been developed [1, 19, 4, 5, 6] (see also references therein). However, these techniques produce only the boundary surfaces with a level of voxel-size accuracy that is too rough for use in industrial applications. As a result, standard segmentation techniques are not suitable for solving our problem.

Another conventional method for extracting boundary surfaces is isosurface extraction [12, 3, 10]. In

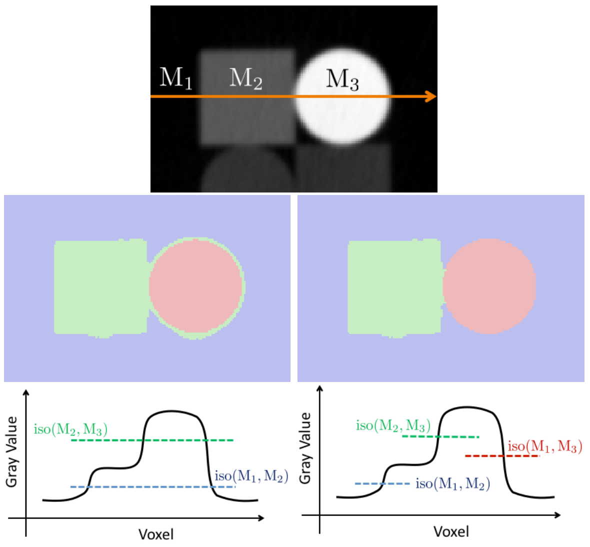

Figure 1: Segmentation results of a multi-material 2D image. The input CT image (top) is segmented (middle) with setting threshold values (bottom). Left: two iterations of thresholding with globally constants. Right: adaptive thresholding proposed in this paper.

isosurfacing, a threshold value is specified as an isovalue, and a polygon mesh approximating the isosurface is then generated. Since the positions of the mesh vertices are computed with continuous interpolation of scalar values at grid points, sub-voxel accuracy can be achieved. This simple method is successfully used when the scanned object consists of a single material

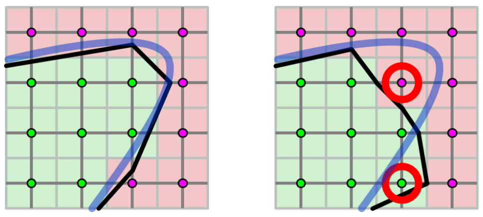
Figure 2: Geometrical accuracy of segmentation results for isosurfacing. Left: in the case the segmentation does not contradict the iso-contour (blue curve), a good polygonal approximation (black polyline) is obtained by a grid-based polygonization. Right: if the segmentation contradicts the iso-contour at the voxels indicated by the red circles, a poor approximation is obtained.

to segment the volume into its background (air) and the object itself.

Let us consider a case in which the scanned object consists of multi-materials (see Fig. 1 for an example). In such a case, one might consider simply performing two iterations of isosurfacing with setting globally constant isovalues, but this solution leads to a misclassification around material M_3 (shown in white). This problem is caused by blurring effects common in CT images, which means there are intermediate values corresponding to material M_2 between materials M_1 (black) and M_3 (white).

To extract accurate boundary surfaces from a multi-material volume, we purpose to develop volume segmentation that satisfies the following two criteria:

- Topological correctness
- Geometrical accuracy for isosurfacing

In order to satisfy the second criterion, the boundary surfaces of the segmented volume should not contradict the isosurfaces, as shown in Fig. 2. Further, the isosurfaces are defined by isovalues depending on the materials meeting at the boundaries. For example, the boundary of material M_3 in Fig.1 is defined by two isovalues $\mathrm{iso}(M_1,M_3)$ and $\mathrm{iso}(M_2,M_3)$ which are adaptively selected according to the materials meeting at the boundary.

In this paper, we describe a simple method for segmenting a volume using label propagation. The important point here is that the propagation generates overlapped labeling, which means multiple labels are assigned to each voxel. Since the multiple labels tell us which materials may be present at the voxel, the isovalues to be used for adaptive thresholding can be calculated from these labels.

Fig. 3 demonstrates the effectiveness of our method. The isosurfaces with incorrect topology are obtained by two iterations of isosurfacing with globally constant isovalues. In contrast, we can extract the topologically correct isosurface via our volume segmentation.

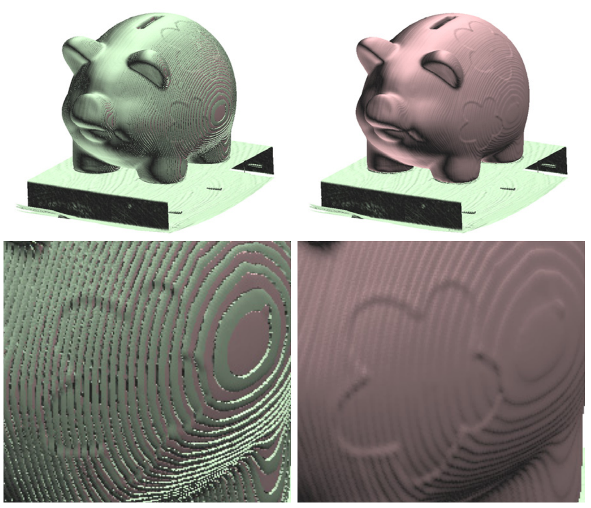
Figure 3: Polygon meshes approximating the isosurfaces of a multi-material volume using two iterations of isosurfacing with globally constant isovalues (left), and isosurfacing with adaptively selected isovalues (right).

This paper is organized as follows. We mention the related work to our method in Section 2. Section 3 describes the multi-material segmentation algorithm, and Section 4 outlines the results, details related limitations and future work.

2 RELATED WORK

Active contours

To extract edges on a noisy image, the active contour model [9] is often used. Using a closed curve as an active contour, the input image is segmented into the foreground and the background. Since polyline or spline is used for the evolution of a boundary curve, we can obtain the boundary curve with sub-pixel accuracy. It is possible to extract the boundary surface in a volume by extending polylines to polygon meshes [20]. Level-set formulations for active contours [15, 13] are also useful to extract topologically complicated boundaries. For extracting the boundaries of multi-material objects, we can use multiphase level-set functions [21].

The active contour models are effective to extract smooth boundaries in noisy volumes, for example, medical imaging. In industrial applications focused in this paper, input CT volumes are more clear and less noisy because they are measured with more radiation exposure of X-rays. Basically, it is possible to adapt the active contours to our problem, but it is not an appropriate choice for solving the problem in terms of its computational cost and accuracy. The surface evolution with minimizing an objective function is a time-consuming task, and a smoothness term in the objective function prevents the surfaces from fitting the accurate isosurfaces. Our method is specialized for fast extraction of the accurate boundaries of multi-material objects in a clear CT volume.

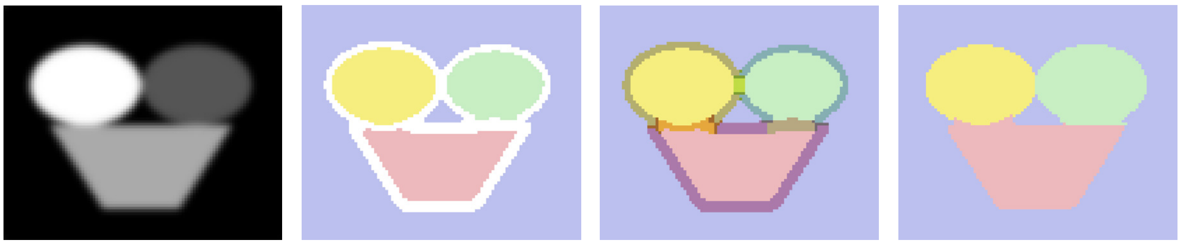
(a) (b) (c) (d)

Figure 4: Algorithm overview. (a) A cross-section of an artificial volume. (b) Labeling of the voxels far from the boundary. (c) Overlapped labeling with the label propagation. (d) Labels decided by the adaptive thresholding.

Polygonization of multi-material objects

Volume segmentation is regarded as a pre-processing of polygonizing multi-material objects, that means polygonizers require the segmented volume (identified materials) as input data in order to find the boundaries. So far, several types of polygonizers have been proposed: extensions of Marching Cubes [22, 7], dual of Marching Cubes [8, 14] and an accurate method for voxels equipped with volume of fractions [2].

Volume segmentation for isosurfacing

Most closely related works are the volume segmentation algorithms proposed in [17, 18]. These segmentation algorithms are also purposed for accurate isosurfacing. In [17], binary partitioning using Graph-cut [4] are repeated after setting the seed regions obtained with region-growing [1]. In [18], a voxel-based active contour model is introduced. Both of the methods give us highly accurate and topologically correct segmentation results.

Comparing with the previous methods, our method can be parallelized easily because label propagation is performed as a local iterative operation. It is more efficient than the graph-cut or active contour based algorithms while offering a similar quality of results.

3 MULTI-MATERIAL SEGMENTATION ALGORITHM

Here we introduce an efficient algorithm for multi-material segmentation of volumes. We denote the input volume as V, which is a set of voxels $\{v_j\}$. A *gray value* denoted by g_j is assigned to voxel v_j. Given number of the materials n, the output is a segmented volume S, which is a set of *material ID* s $\{s_j\}$ each representing the material ID $s_j \in \{1, \cdots, n\}$ of voxel v_j.

As shown in Fig. 4, the algorithm consists of several steps, which are briefly described below.

1. Voxels far from the boundary surfaces are labeled with thresholding of the gray values and their gradients . (Fig. 4 (b))
2. Labels near the boundary surfaces are propagated with overlapping to ascertain which materials meet at the boundaries. (Fig. 4 (c))
3. The appropriate label is selected as the material ID among the multiple labels assigned to a voxel with adaptive thresholding. (Fig. 4 (d))

The rest of this section explains the details of the above steps.

3.1 Labeling apparent regions

With thresholding, we first label the regions in which a material is apparently present. This section first explains how the threshold values are computed semi-automatically, and then describes the method of labeling voxels according to the gray values of the volume.

Semi-automatic computation of threshold values

We compute threshold values using k-means clustering on the gray values [11]. The computation is started by setting the initial threshold values $\{T_i\}$ $(i = 1, \cdots, n-1)$ given by the user. The threshold T_i is given as the user's guess for the isovalue $\mathrm{iso}(M_i, M_{i+1})$. We also assume that the values are enumerated in the ascending order as $T_1 < T_2 < \cdots < T_{n-1}$.

The following steps are then iterated to optimize the threshold values.

1. Define sets of voxels $\{G_i\}$ $(i = 1, 2, \cdots, n)$ as
$$G_i = \begin{cases} \{v_j \mid g_j \le T_1\} & \text{if } i = 1 \\ \{v_j \mid T_{i-1} < g_j \le T_i\} & \text{if } 1 < i < n \\ \{v_j \mid T_{n-1} < g_j\} & \text{if } i = n. \end{cases}$$
2. For each G_i, calculate the average value m_i and the standard deviation σ_i of the gray values $\{g_j\}$.
3. Update the threshold as $T_i = (m_i + m_{i+1})/2$.
4. Go back to Step 1 if the new thresholds are far from the old ones.

According to our experiments, the above iterative process quickly converges within ten iterations. We use a

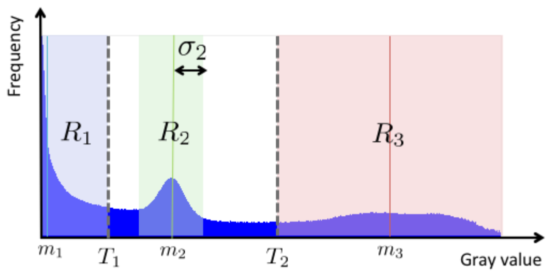

Figure 5: Setting threshold ranges $\{R_i\}$ for three materials on the histogram.

histogram of the gray values to accelerate the computation of the average value and the standard deviation via integrating the gray values in G_i.

We then define a set of ranges $\{R_i\}$ referred to as the *threshold range*, as shown in Fig. 5.

$$R_i = \begin{cases} [g_{\min}, T_1) & \text{if } i = 1 \\ [\max(T_{i-1}, m_i - \sigma_i), & \\ \quad \min(T_i, m_i + \sigma_i)) & \text{if } 1 < i < n \\ [T_{n-1}, g_{\max}] & \text{if } i = n, \end{cases}$$

where $g_{\min}$ and $g_{\max}$ are the minimum and maximum of the whole gray values, respectively.

The average value m_i corresponds to the representative values of the material M_i. Using the values $\{m_i\}$, we compute the isovalues which consists of a full symmetric matrix. The (α, β)-th element of the matrix corresponds to the isovalue $\text{iso}(M_\alpha, M_\beta)$. More details are described in Section 3.3.

For volumes obtained with well-analyzed CT scanning systems, the representative values of known materials are sometimes given. In such a case, the ranges $\{R_i\}$ are computed with fixing the average values $\{m_i\}$ as the given representative values.

Labeling with the threshold range

We decide on the label for voxel v_j, whose gray value g_j is in the range of R_i and whose gradient magnitude is low. Let L_i be the set of voxels classified as the material M_i.

$$L_i = \{v_j | \; g_j \in R_i \text{ and } \|\nabla g_j\| < \varepsilon\},$$

where the gradient ∇g_j is computed using 3D Sobel filter, and ε is a user-specified parameter.

Once the voxels in L_i are decided, we eliminate isolated voxels from L_i. The elimination rule is formulated as

$$L_i \leftarrow L_i \setminus \{v_j | \; N_j \cap L_i = \emptyset\},$$

where N_j represents 6-connected neighbors of voxel v_j.

We also define *doubtful voxels* D as

$$D = V \setminus \bigcup L_i.$$

Fig. 6 shows an example of labeling with the threshold range.

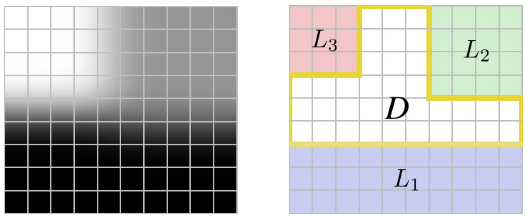

Figure 6: Labeling apparent regions. Left: gray values $\{g_j\}$. Right: labeling with the threshold ranges $\{R_i\}$.

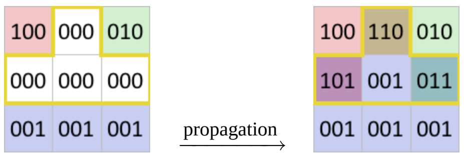

Figure 7: Label propagation with bitwise OR operation.

3.2 Label propagation

To fill out doubtful voxels D, the steps of label propagation are performed by setting apparently labeled voxels as seeds. The proposed method involves two propagations. The first aims to fill voxels without overlapping, and the second generates multiply labeled voxels while allowing overlapping of labeled regions.

Before describing the details of theses propagations, we introduce the data structure for representation of multiply labeled voxels.

Data structure of labels

To represent sets of labeled voxels $\{L_i\}$ efficiently, we assign an n-bit sequence f_j to voxel v_j. The i-th bit denoted by $f_j^{(i)} \in \{0, 1\}$ is defined by

$$f_j^{(i)} = \begin{cases} 1 & \text{if } v_j \in L_i \\ 0 & \text{otherwise.} \end{cases}$$

For convenience, we denote the number of "1" in f_j by $|f_j| = \Sigma_{i=1}^{n} f_j^{(i)}$.

Label propagation without overlapping

We propagate label over doubtful voxels D by performing bitwise OR operation denoted by $\vee$ on the neighboring labels, as shown in Fig. 7.

$$f_j \leftarrow \begin{cases} \bigvee_{v_k \in N_j} f_k & \text{if } |f_j| = 0 \\ f_j & \text{otherwise} \end{cases} \qquad (1)$$

The above updating rule is simultaneously applied to all voxels in D, and is iterated until there are no voxels with $|f_j| = 0$. The two images of Fig. 8 show an intermediate step and the converged result.

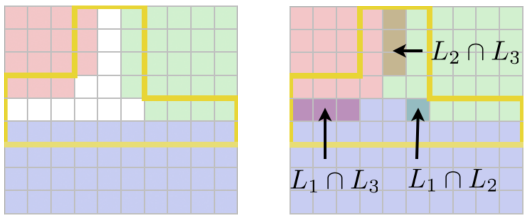

Figure 8: Label propagation without overlapping by equation (1). Left: the labels after performing a propagation step for the right image in Fig.6. Right: the converged labels.

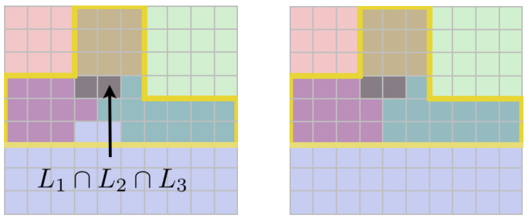

Figure 9: Label propagation with overlapping by equation (2). Left: the labels after performing a propagation step for the right image in Fig.8. Right: the converged labels.

Label propagation with overlapping

The second propagation is similar to the first one, and is as shown below.

$$f_j \leftarrow \begin{cases} \bigvee_{v_k \in N_j} f_k & \text{if } |f_j| = 1 \\ f_j & \text{otherwise} \end{cases} \qquad (2)$$

The above propagation does not change the labels at the regions in D each of which is surrounded by a single label. Except in such regions, the iteration will converge when voxels in D are given at least two labels. The two images of Fig. 9 show an intermediate step and the converged result.

3.3 Adaptive thresholding

The final result of volume segmentation is obtained by selecting one label at multiply labeled voxels. The selected label at voxel v_j is set as a material ID s_j in the segmented volume S. The left image of Fig. 10 shows the segmentation result of the input shown in Fig.6.

Our segmentation is aimed at extracting the boundary surface as an isosurface. Accordingly, the isovalue $\text{iso}(M_\alpha, M_\beta)$ should be between the two gray values at the neighboring voxels segmented as materials M_α and M_β. As proposed in [17], the isovalue is calculated by

$$\text{iso}(M_\alpha, M_\beta) = (m_\alpha + m_\beta)/2.$$

Thresholding doubly labeled voxels

Before considering the thresholding of multiply labeled voxels in general, we explain a simple case involving the thresholding of a doubly labeled voxel v_j with $|f_j| =$

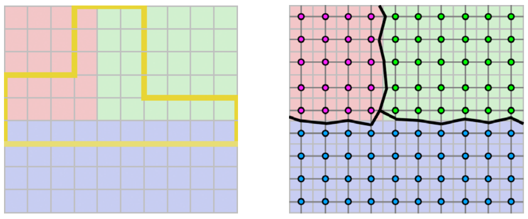

Figure 10: Adaptive thresholding. Left: the selected labels on the right image of Fig.9 with adaptive thresholding. Right: the extracted iso-contours using the segmentation result.

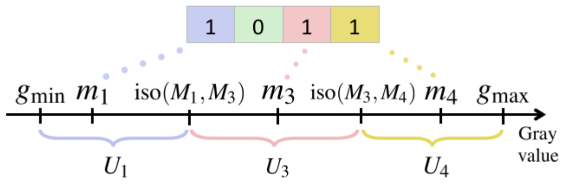

Figure 11: Ranges $\{U_{i_k}\}$.

2. Let i_1, i_2 $(i_1 < i_2)$ be two material indices such as $f_j^{(i_1)} = 1, f_j^{(i_2)} = 1$, and all other bits cleaned.

In this case, voxel v_j is classified by comparing its gray value g_j and the isovalue $\text{iso}(M_{i_1}, M_{i_2})$. The material ID s_j is decided by the following rule.

$$s_j = \begin{cases} i_1 & \text{if } g_j < \text{iso}(M_{i_1}, M_{i_2}) \\ i_2 & \text{if } g_j \geq \text{iso}(M_{i_1}, M_{i_2}) \end{cases}$$

Thresholding multiply labeled voxels

In the general case of $|f_j| = K$, we divide the whole range of the gray values $[g_{\min}, g_{\max}]$ into K sub-ranges. Then, the range containing the gray value g_j is found to decide the material ID s_j.

We first prepare the sequence $\{i_1, \cdots, i_K\}$ satisfying $f_j^{(i_k)} = 1$ $(k = 1, \cdots, K)$. The following ranges $\{U_{i_k}\}$ are then created, as shown in Fig. 11.

$$U_{i_k} = \begin{cases} [g_{\min}, \text{iso}(M_{i_1}, M_{i_2})) & \text{if } k = 1 \\ [\text{iso}(M_{i_{k-1}}, M_{i_k}), \text{iso}(M_{i_k}, M_{i_{k+1}})) & \text{if } 1 < k < K \\ [\text{iso}(M_{i_{K-1}}, M_{i_K}), g_{\max}] & \text{if } i = K \end{cases}$$

Finally, the material ID s_j is set to the index satisfying the following relationship.

$$g_j \in U_{s_j}$$

See the appendix for more details on the implementation of the above procedure.

Isosurface extraction

Given the input volume V and its segmentation S, an isosurface is polygonized with a grid-based algorithm, for example Marching Cubes [12]. As shown in the right image of Fig. 10, we generate mesh vertices which

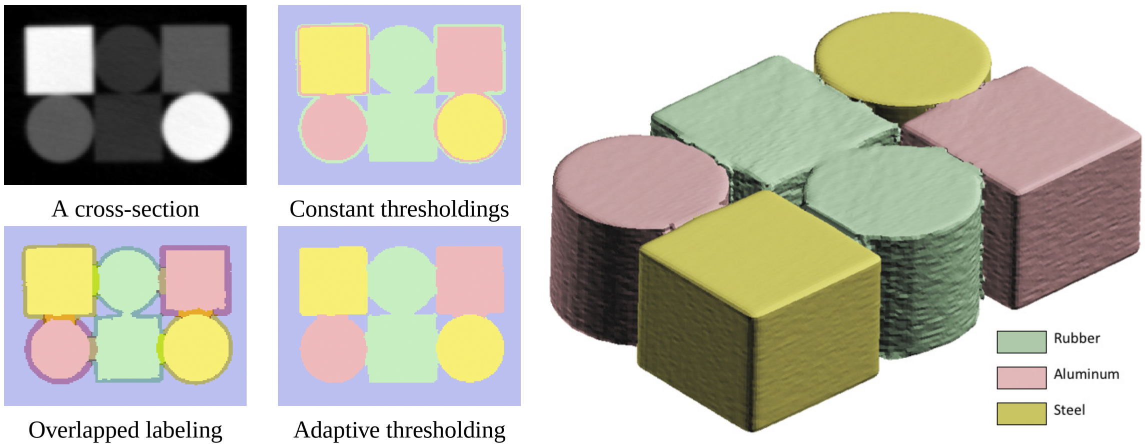

Figure 12: A segmentation result of the CT volume obtained by scanning blocks consisting of steel, aluminum and rubber. Left: cross-sections of the volume. Right: polygonized isosurface using our segmentation.

are the intersection points of the grid-edges and the isosurface, and the vertices are then connected with polygons in the grid-cells.

If two neighboring voxels denoted by v_j and v_k are associated to different material IDs ($s_j \neq s_k$), a mesh vertex is generated on the grid-edge whose end points are v_j and v_k. Let $\mathbf{p}_j$ and $\mathbf{p}_k$ be the coordinates of the voxels v_j and v_k, respectively. The coordinates of the mesh vertex $\mathbf{p}$ are computed using the following linear interpolation.

$$\mathbf{p} = (w_k \mathbf{p}_j + w_j \mathbf{p}_k)/(w_j + w_k), \tag{3}$$

where the weights $w_j = g_j - \mathrm{iso}(M_{s_j}, M_{s_k})$ and $w_k = \mathrm{iso}(M_{s_j}, M_{s_k}) - g_k$.

If $w_j \cdot w_k < 0$, the isosurface does not intersect to the grid-edge, that means the material IDs contradict the isosurface as shown in the right image of Fig. 2. In this case, the mesh vertex is located at the middle point of $\mathbf{p}_j$ and $\mathbf{p}_k$. As reported in the next section, this case happens only less than 10%.

4 RESULTS AND DISCUSSION

Results

Fig. 12, 13 and 14 show the results. The sizes of the input volumes are summarized in Table 1.

In Fig. 12, the input CT volume consists of 4 materials: steel, aluminum, rubber and air. The left four images show cross-sections of the volume. The right image shows a polygon mesh approximating the isosurface in the volume. The isovalues of the isosurface are adaptively selected in terms of the segmented volume.

Fig. 13 and Fig. 14 show the resulting isosurfaces representing engine parts. The input CT volumes consist of steel, aluminum and air.

Timing and quality

In Table 1, we demonstrate the speed of our segmentation algorithm. The timing is measured on a desktop computer equipped with Intel Core i7 4-cores of 2.93 GHz and 16 GB main memory. Since the algorithm can be parallelized in a straightforward way, the computational time proportionally improves with increasing the number of threads.

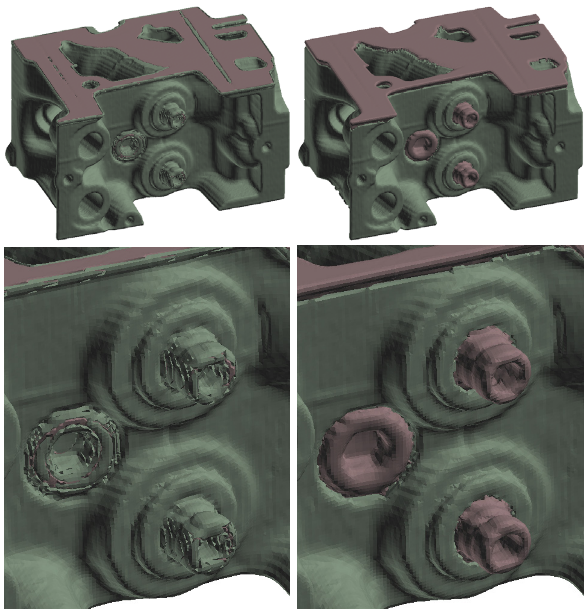

Figure 13: Isosurfaces of the CT scanned engine parts consisting of steel (pink) and aluminum (green). Left: the isosurfaces obtained with the globally constant isovalues. Right: the isosurface obtained with the adaptive isovalues.

Table 2 shows the segmentation quality for isosurfacing. For each material ID, a polygon mesh is firstly generated by Marching Cubes [12] with the same grid-resolution as that of the input volume V. Then, the quality is computed as the rate of the mesh vertices which are successfully located on the isosurface ($w_j \cdot w_k \geq 0$ in equation (3)). In all results, more than 90% vertices are generated on the isosurfaces.

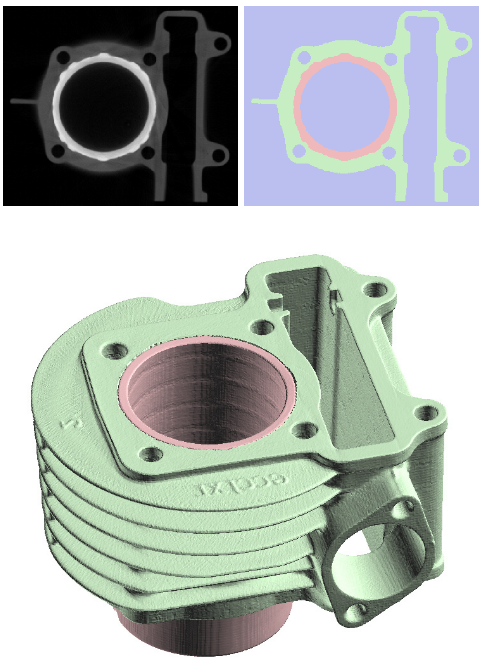

Figure 14: A result of the CT volume segmentation (the cylinder head of a motor-bike). Top: cross-sections of the input and segmented volume. Bottom: the isosurface obtained with adaptive isovalues.

Table 1: Computational time of volume segmentation.

Name	Size	#threads	Time (sec.)
Blocks (Fig. 12)	$200 \times 160 \times 50$	1	0.2082
		2	0.1264
		4	0.0766
Engine (Fig. 13)	$256 \times 256 \times 256$	1	1.7132
		2	0.9099
		4	0.5408
Cylinder (Fig. 14)	$512 \times 512 \times 178$	1	4.6464
		2	2.4587
		4	1.3674

Table 2: Isosurface quality.

Name	Material	#vertices	Quality (%)
Blocks (Fig. 12)	Air	67 k	96.02
	Rubber	28 k	91.20
	Aluminum	27 k	95.36
	Steel	25 k	99.07
Engine (Fig. 13)	Air	314 k	98.98
	Aluminum	297 k	97.32
	Steel	80 k	94.42
Cylinder (Fig. 14)	Air	746 k	99.84
	Aluminum	714 k	99.81
	Steel	209 k	99.92

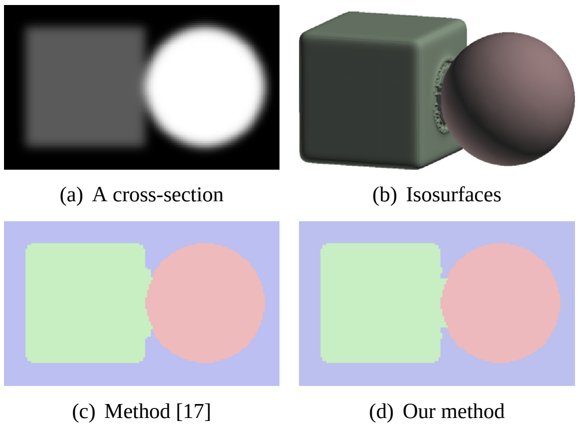

Figure 15: A comparison with the method [17] using an artificial volume.

Comparison

Fig. 15 shows a comparison with the graph-cut based method [17] using an artificial volume which simulates a scanning result of a sphere (high gray value) and a cube (middle gray value). We applied Gaussian convolution to produce blurring effects in CT scanning after rasterizing the cube and sphere. The variance of the Gaussian kernel is six voxels. The size of the volume is $288 \times 192 \times 192$.

To apply the method [17], we used the public available software *VolCut* [16]. Fig. 15(c) and (d) show cross-sections of the segmentation results. As summarized in Table3, almost the same quality results for isosurfacing are obtained by our method and the method [17].

In order to evaluate the accuracy of the segmentation, we also counted the misclassified voxels by comparing the results and the rasterization of the cube and sphere. In the result of [17], about twice voxels which should be classified as the background are misclassified as the cube. The misclassification is mainly observed near the contact part of the cube and sphere where the boundary surface of the background has sharp features. This problem is caused by the smoothing effects of the graph-cut.

From the above comparison, we achieve almost the same level of the quality while the speed is much faster (more than ten times faster).

Parameter setting

In our segmentation method, the resulting segmentation depends mainly on the user-specified parameter ε. Fig. 16 shows a part of the cross-section of the volume in Fig. 14, and the effects of changing ε. Thin parts of materials in the CT image do not appear in the segmented image with too small ε (Fig. 16(b)), while unnecessary parts appear with too large ε (Fig. 16(d)). Since an appropriate ε value is not computed with the

Table 3: Quantitative comparison with the method [17] by using the volume shown in Fig. 15.

Method	Time (sec.) Single thread	Isosurface quality (%)			#Misclassified voxels		
		Sphere	Cube	Background	Sphere	Cube	Background
Method [17]	12.0	99.37	97.70	98.92	4,592	5,632	3,584
Our method	0.95	99.46	95.77	97.74	4,516	5,632	1,876

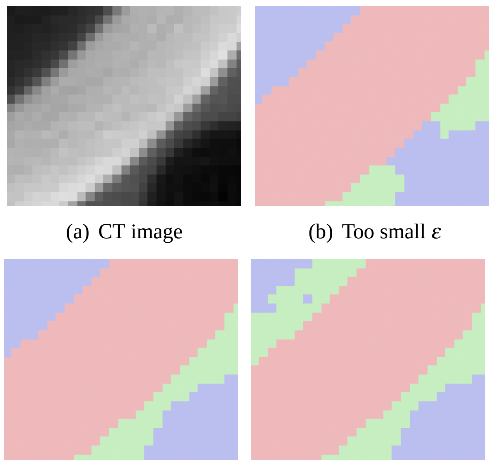
(a) CT image (b) Too small ε (c) Appropriate ε (d) Too large ε

Figure 16: Effects of the gradient threshold ε.

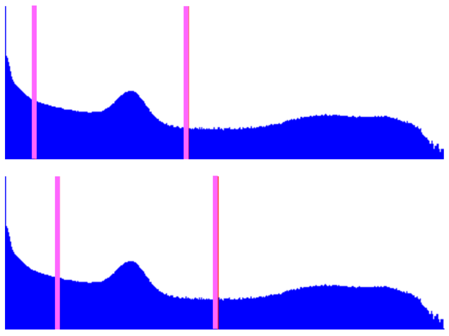
Figure 17: Two different selections of the initial $\{T_i\}$ both of which will converge to Fig. 5.

current method, a number of ε values must be examined to obtain a segmented image suitable for the user.

In the case representative CT values $\{m_i\}$ are unknown, the user must guess the initial thresholds $\{T_i\}$. Then, values $\{m_i\}$ are automatically computed through the iterative process described in Section 3.1. According to our experiments, a rough selection of $\{T_i\}$ is fine for computing the appropriate values for $\{m_i\}$. For example, Fig. 17 shows two different selections of $\{T_i\}$ on the histogram of gray values, and both of the selections give the same $\{m_i\}$ (shown in Fig. 5) within ten iterations.

Limitation and future work

Since our fast segmentation algorithm is specialized for CT volumes used for industrial applications, highly noisy volumes used for medial imaging might not be segmented correctly. This problem is a drawback of the simplicity of the algorithm. We plan to make the algorithm more noise-robust in order to deal with various kinds of volumes while keeping its high-speed.

Another limitation of our algorithm is that it is difficult to correctly find thin-structures consisting of a few voxel thickness (tubes and sheets). The reason of this problem is that the CT value on such a thin-structure can not reach to the corresponding representative value because of blurring effects. According to our experiments, at least four or five voxel thickness is required to find appropriate seed voxels and doubtful voxels.

ACKNOWLEDGMENTS

Authors would like to thank for anonymous reviewers for valuable comments and suggestions. The CT volumes are courtesy of Toyota Motor Corporation (Fig. 2 and 12), Michael Bauer (Fig. 3), General Electric (Fig. 13), and VCAD System Research Program at RIKEN (Fig. 14). This work was partly supported by KAKENHI (22246018).

REFERENCES

[1] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):641–647, 1994.

[2] J.C. Anderson, C. Garth, M.A. Duchaineau, and K.I. Joy. Smooth, volume-accurate material interface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 16(5):802–814, 2010.

[3] J. Bloomenthal. Polygonization of implicit surfaces. *Computer-Aided Geometric Design*, 5(4):341–349, 1982.

[4] Y. Boykov and M.-P. Jolly. Minteractive graph cuts for optimal boundary & region segmentation of objects in n-d images. In *International Conference on Computer Vision (ICCV)*, pages 105–112, 2001.

[5] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[6] A. Delong, A. Osokin, H.N. Isack, and Y. Boykov. Fast approximate energy minimization with label costs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2173–2180, 2010.

[7] T. Fujimori and H. Suzuki. Surface extraction from multi-material ct data. In *Ninth International Conference on Computer Aided Design and Computer Graphics (CAD-CG'05)*, pages 319–324, 2005.

[8] T. Ju, F. Losasso, S. Schaefer, and J. Warren. Dual contouring of hermite data. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2002)*, 21(3):339–346, 2002.

[9] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1987.

[10] F. Labelle and J. R. Shewchuk. Isosurface stuffing: Fast tetrahedral meshes with good dihedral angles. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007)*, 26(3):#57, 2007.

[11] S. P. Lloyd. Least square quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[12] W. E. Lorensen and H. E. Cline. Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics (Proceedings of SIGGRAPH '87)*, 21(3):163–169, 1987.

[13] S. Osher and R. Fedkiw. *The Level Set Method and Dynamic Implicit Surfaces*. Springer-Verlag, 2002.

[14] P. Powei Feng, T. u, and J. Warren. Piecewise tri-linear contouring for multi-material volumes. *Advances in Geometric Modeling and Processing (Lecture Notes in Computer Science)*, pages 43–56, 2010.

[15] J. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[16] H. M. Shammaa. Volcut. http://sites.google.com/site/mhaithamshammaa/.

[17] H. M. Shammaa, Y. Ohtake, and H. Suzuki. Segmentation of multi-material ct data of mechanical parts for extracting boundary surfaces. *Computer-Aided Design*, 42(2):118–128, 2010.

[18] H. M. Shammaa, H. Suzuki, and Y. Ohtake. Creeping contours: A multi-label image segmentation method for extracting boundary surfaces of parts in volumetric images. In *Asian Conference on Design & Digital Engineering*, pages 603–610, 2010.

[19] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[20] I. Takahashi, S. Muraki, A. Doi, and A. Kaufman. Three-dimensional active net for volume extraction. In *Visual Data Exploration and Analysis (SPIE Proceedings of SPIE Volume: 3298)*, pages 184–193, 1998.

[21] L. A. Vese and T. F. Chan. A multiphase level set framework for image segmentation using the mumford and shah model. *International Journal of Computer Vision*, 50:271–293, 2003.

[22] Z. Wu and J. M. Sullivan, Jr. Multiple material marching cubes algorithm. *International Journal for Numerical Methods in Engineering*, 58(2):189–207, 2003.

APPENDIX

Algorithm 1 Adaptive-Thresholding

```
input:
$n$ - the number of materials
$f$ - $n$ bit sequence
$g$ - gray value of a voxel
output:
$i$ - material ID

$i \leftarrow 0$
for $k = 1 \rightarrow n$ do
  if $f^{(k)} = 1$ then
    if $i = 0$ then
      $i \leftarrow k$
    else
      $iso \leftarrow (m_i + m_k)/2$
      if $g < iso$ then
        return $i$
      end if
      $i \leftarrow k$
    end if
  end if
  $k \leftarrow k + 1$
end for
return $i$
```