Surface Extraction from Multi-material CT Data

Tomoyuki Fujimori* and Hiromasa Suzuki
Research Center for Advanced Science and Technology, The University of Tokyo, Tokyo, JAPAN

Abstract

This paper describes a method for extracting surfaces from multi-material CT (Computed Tomography) data. Most contouring methods such as Marching Cubes algorithm assume that CT data are composed of only two materials. Some extended methods such as [3, 6] can extract surfaces from the multi-material (non-manifold) implicit representation. However, these methods are not directly applicable to CT data that are composed of three or more materials. There are two major problems that arise from fundamentals of CT. The first problem is that we have to use $n(n-1)/2$ threshold values for CT data contains n materials and select appropriately one threshold value for each boundary area. The second is that we cannot reconstruct only from CT data in which area three or more materials are adjacent each other. In this paper, we propose a method to solve the problems by using image analysis and demonstrate the effectiveness of the method with application examples construct polygon models from CT data of machine parts.

Key Words: Computed Tomography, Medial Axis/Surface, Contouring

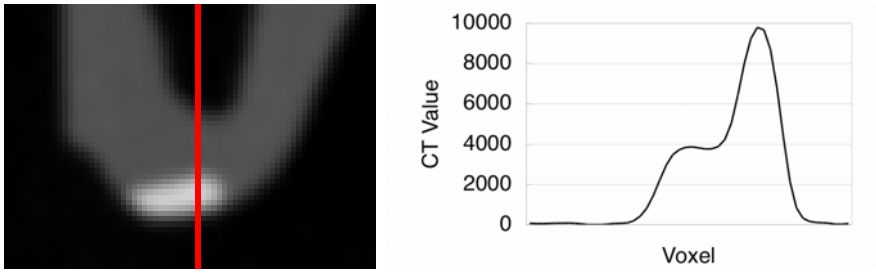
1. Introduction

CT (Computed Tomography) is a powerful non-destructive evaluation technique to generate cross-sectional x-ray images of objects, from which we can further produce three dimensional volumetric images. Characteristics of the internal structure of objects such as dimensions, shape, internal defects and density are readily available from CT images.

In this study, we focused on the industrial application of x-ray CT to analyze multi-material machine parts. Compared to medical applications, industrial ones are prescribed relatively higher precision for their purpose. Therefore, we concentrate on solving problems which causes degradation of the precision when we extract surfaces from multi-material CT data.

There are two major problems that arise from fundamentals of CT. Figure 1 shows the fundamentals in brief. CT values change gradually corresponding to the x-ray absorption of materials in a real space as shown Figure 1 (b), because each voxel value is influenced by surrounding area in a real space. Some filtered reconstruction methods are proposed, however the blur is still ineludible.

* Corresponding author:
Tel: +81-(0)3-5452-5183
Fax: +81-(0)3-5452-5186
E-mail: {fujimori, suzuki}@den.rcast.u-tokyo.ac.jp
Email: kdhong @university.ac.kr



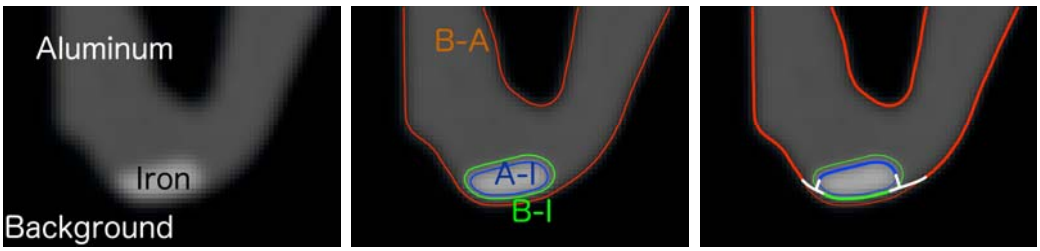
(a) (b)

Fig. 1. A typical example image of multi-material CT data (a) and transition of CT values on the red line (b). Currently, CT data do not clearly picture a real space. At the boundary of materials, CT values change gradually corresponding to the x-ray absorption of materials.

Figure 2 shows the problems caused by the fundamentals. The first problem is that we have to use $n(n-1)/2$ threshold values for CT data contains n materials and select appropriately one threshold value for each boundary area. For example, when CT data contains $n=3$ materials: *background*, *aluminum* and *iron* as shown in (a), we can take three threshold values: *background-aluminum*, *background-iron* and *aluminum-iron* into account. For extracting the iron area properly, we have to use the *aluminum-iron* threshold value at the upper side of the area and the *background-iron* threshold value at the downside as shown in (c). We solve this problem by using a classification method described in Section 3.1.

Before illustrating the second problem, we define *junction areas* in which three or more materials are adjacent each other. The problem is that we cannot adopt existing contouring methods directly to junction areas. Because a CT value is influenced by surrounding areas, the value are not credible and so threshold values are indeterminable for extracting desired surfaces which are described as white lines in Figure 2(c). To solve the problem, we introduce a thinning method in Section 3.2.

Figure 3 shows an overview of our method. Figure 3(a) describes an example of a multi-material object. In this example, the white area is filled with higher x-ray absorption material (e.g. *iron*), the gray area is filled with lower x-ray absorption material (e.g. *aluminum*) and the black area is *background* (e.g. air). Figure 3(b) describes multi-material CT data of (a) and we call each element of



(a) (b) (c)

Fig. 2. Given multi-material CT data (a), we have to select from $n(n-1)/2$ threshold values to construct surfaces (b) and create non-manifold surfaces appropriately at the junction areas (c).

data *voxel* based on common manner. This simple CT data shows our problems described earlier. To solve them, at the beginning, we apply *voxel based classification* to determine which threshold value we should use for extracting boundaries of two materials as shown in Figure 3(c). However, the classification is not perfect at this point. We cannot classify junction areas and then mark them as *junction* as shown in white voxels of Figure 3(c). Next, we construct *cell* models from classified voxels. A cell is cube comprising eight corner voxels. When all of voxels are classified (not marked as junction), we can determine a threshold value for these cells as shown in Figure 3(d). At the same time, junction areas remain in Figure 3(d). Therefore, we apply *cell thinning* to remove junction cells and guarantee contouring algorithms can extract surfaces as shown in Figure 3(e). At the last, we apply a Marching Cubes variant which correspond to non-manifold objects as shown in Figure 3(f).

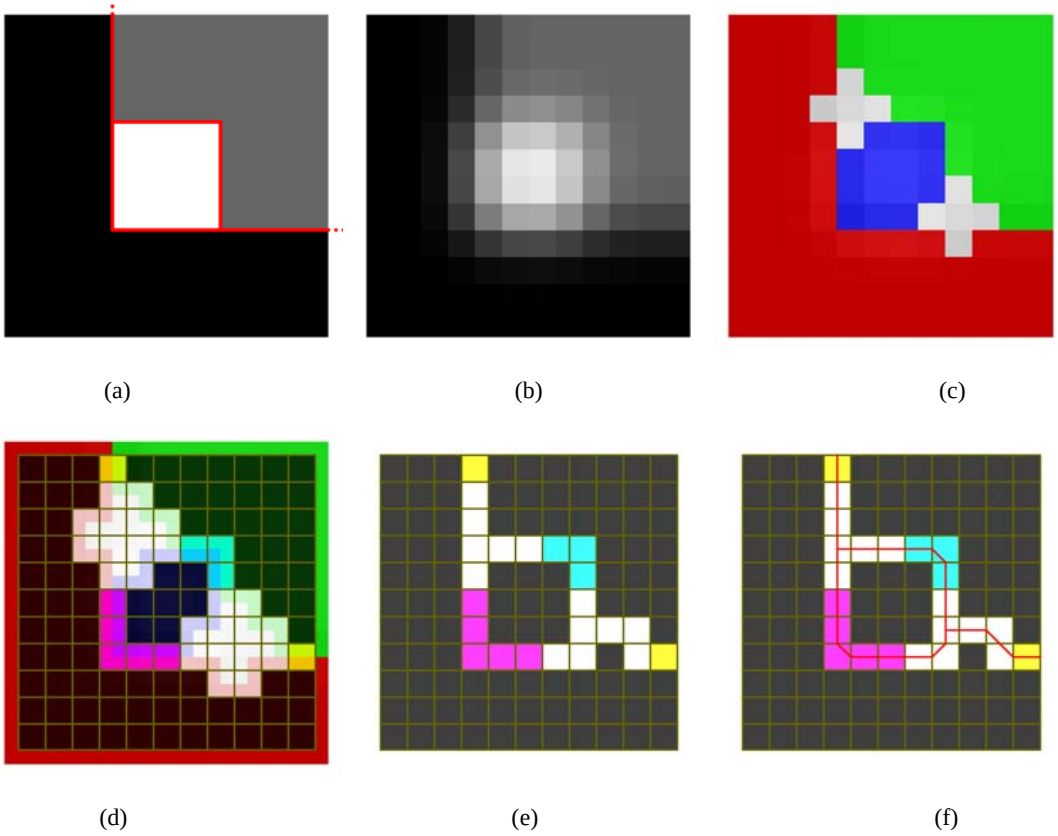


Fig. 3. An overview of our method: a multi-material object in a real space (a), multi-material CT data (b), voxel based classification (c), cells which are derived from classified voxels (d), cell thinning (e), and multi-material contouring (f).

This paper consists of six sections. Section 2 surveys important related research works and Section 3 discusses our major contributions of this study. We propose three dimensional image analysis methods. Section 4 discusses an algorithm when applying the existing contouring method. Section 5 shows some results obtained by prototype systems and Section 6 gives summary and future works.

2. Related Works

Cube-based methods such as the Marching Cubes algorithm [5] and its variants generate one or more triangles for each cube in the grid that intersects the contour of a signed field. Surface samples are computed by the intersection of the cube edges with the surface, and triangles connecting these samples are generated. Most cube-based techniques are conceptually derived from the Marching Cubes algorithm where a pre-processed triangulation is stored in a table for all possible configurations of edge intersections. Additionally, Lorensen's original Marching Cubes is unable to process non-manifold objects. Hege et al. [3] proposed the Generalized Marching Cubes.

Furthermore, Nielson et al. [6] proposed the extended Marching Tetrahedra for segmented data. Their approached problems alike our ones, however, they are not referring threshold values and characteristics of CT images. Also, Adolfsson et al. [1] removed surface irregularities such as ridges at junction areas. However, they targeted medical applications and their purpose is different from us. Therefore we proposed a cell-based thinning algorithm [2] to extract surfaces from CT data of thin plate structures.

We can apply a general contouring algorithm for the signed field to the cell. This paper discusses the generalized Marching Cubes algorithm but other algorithms are applicable in principle. However, they require Hermite data or normal vectors at samples that cannot be accurately defined in our typical CT image examples. Thus we cannot guarantee the accuracy with those contouring methods.

3. Multi-material Image Analysis

In this section, we discuss characteristics of multi-material CT data at the beginning. Then, we produce two methods which leverage the characteristics to solve the problems. The first method classifies voxels to select threshold values at boundary of two materials. Secondly, we construct cell models from voxels and apply a cell thinning algorithm to set up junction areas so that the contouring algorithm may extract desired surfaces.

In Section 1, we remarked CT data do not clearly picture a real space. Therefore, we expect this CT blur is approximated by the Gaussian blur and test it through experiments. Figure 1(b) is a collateral evidence of the expectation. We applied the normal distribution curve to the graph and estimated the Gaussian kernel. In this example case, the Gaussian kernel is a sphere with radius 2 or 3 voxels. In other words, we blur the clear image of a real space by such a kernel, the result well-approximates CT data. Additionally, we define a distance value R which is the radius of the estimated spherical Gaussian kernel. We propose image analysis methods in the following subsections consistent with the blur characteristics and definition.

3.1. Voxel based Classification

When CT data contains N materials, we denote average CT values of those materials by $(v_1, v_2, \cdots, v_n)$. A value v_i is obtained by CT scanning a space filled with a material m_i and averaging values to cancel noises. Next, we denote a threshold value to extract the boundary between material x and y by t_{xy}. Because CT value of an arbitrary voxel is blurred by the Gaussian-like kernel, threshold value $t_{xy} = (v_x + v_y)/2$ become good approximation.

In the first step, we roughly classify all voxels by Table 1. This classification has a problem that there exists a material pair x, y that satisfy $t_{(i-1)i} \leq t_{xy} < t_{i(i+1)}$ where classes are given as $a = a_i (1 < i < n)$. For example, we have no way of distinguishing the *aluminum* area and the transition area from *background* to *iron* based on only its CT value.

CT Value	Class of the 1st step
$v < t_{12}$	a_1
$t_{12} \leq v < t_{23}$	a_2
$t_{23} \leq v < t_{34}$	a_3
$\vdots$	
$t_{(n-2)(n-1)} \leq v < t_{(n-1)n}$	a_{n-1}
$t_{(n-1)n} \leq v$	a_n

Table. 1. We classify voxels into a_i depending on its value in the first step of voxel based classification.

We solve this problem by using a geometrical three dimensional image analysis in the second step and the third step. Basically, these steps classify voxels again and again depending on how its neighbors were classified.

In the second step, about a voxel V which is classified as class a_i, we consider voxels $\mathbf{V_s}$ comprised in V's neighbor sphere with radius r_s. Where appropriate r_s is given, we can assume that a material of V is i when all of $\mathbf{V_s}$ are classified as a_i and we define such a voxel has class $b = b_i$. The assumption is considered reasonable and proper if $r_s \geq R$. This step classifies regions which have some extents and filled with the same material. Figure 4 shows an example application of the second step.

Figure 5 shows the result of the step and it is an input of the next step.

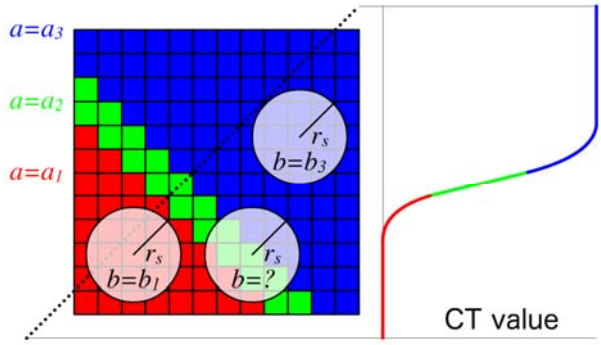


Fig. 4. In the transition area from material 1 to material 3, the second step classifies voxels which we can determine its material certainly.

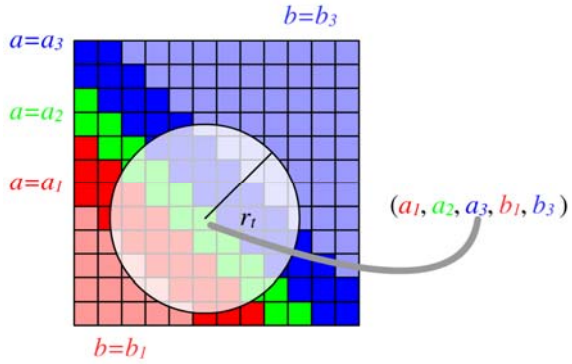


Fig. 5. The third step uses the neighbor sphere of the doubled radius, and classifies properly boundaries of two materials.

In the third step, we consider voxels $\mathbf{V_t}$ in the same manner as the previous step. $\mathbf{V_t}$ comprised in a voxel V's neighbor sphere with radius r_t. This step aims exact detection of boundaries between two materials. We adopt a condition $r_t \geq 2R$ so that the neighbor sphere can involve two or more different b-classified voxels. Classes are given as a list $(a_{j_1}, a_{j_2}, \cdots, a_{j_l}, b_{k_1}, b_{k_2}, \cdots, b_{k_m})$ as shown in Figure 5. The list is consist of l-tuple of classes a_j and m-tuple of classes b_k. If $m = 2$ and $^{\forall} j \in (j_1, j_2, \cdots, j_l), k_1 \leq j \leq k_2$, we can determine a material of the voxel is either k_1 or k_2. We give class $b = b_{k_1}$ to the voxel if $v < t_{k_1 k_2}$, or give class $b = b_{k_2}$. We assume each material region has R voxel extent at the lowest and classifies properly boundaries of two materials.

At this point, voxels which are not given class b remain. Such voxels compose a junction area, therefore we give class $b_{junction}$ to them.

3.2. Cell Thinning

We construct a cell model from the classified voxels, because most contouring methods require not only voxels but also cells to extract surfaces. And we put a concept ¥textit{foreground} to work. If a set of cells represent objects in regular grid space, we say those cells are foreground cells (We can also say background cells in contrary senses, however this concept is ambiguous with the same term of material).

In this subsection, we roughly construct a set of foreground cells $\mathbf{C_x}$ at the beginning. This is a discrete well-approximated of the desired surfaces. However we cannot apply contouring methods directly to $\mathbf{C_x}$, because junction areas still remain in $\mathbf{C_x}$. Therefore we remove unnecessary cells by using a cell thinning method and construct new foreground cells $\mathbf{C_Y}$ from which we will extract desired surfaces.

A set of foreground cells $\mathbf{C_x}$ consists of cells whose eight vertices are classified into two or more materials or one of vertices is classified into $b_{junction}$. If eight vertices have the same material except $b_{junction}$, the cell exists inside of an object and so no surface pass through it. $\mathbf{C_x}$ has the following properties:

- If no class is $b_{junction}$ for each vertex of a cell, the cell is already valid to apply contouring methods. From the geometrical characteristics of voxel based classification, the cell represents a boundary and its eight voxels are classified just two materials.
- If classes of eight vertices involve $b_{junction}$, we call the cell *junction cell* and it is invalid for the contouring method. Therefore we apply a cell thinning method to those cells.

Now we construct a set of cells $\mathbf{C_Y}$ from $\mathbf{C_x}$ as shown in Figure 6. We apply a cell thinning method to $\mathbf{C_x}$ to keep the topology of the surfaces. It is different between our method and the existing thinning methods that we take account of not only cells but also classified vertices and the result cells become suitable for contouring.

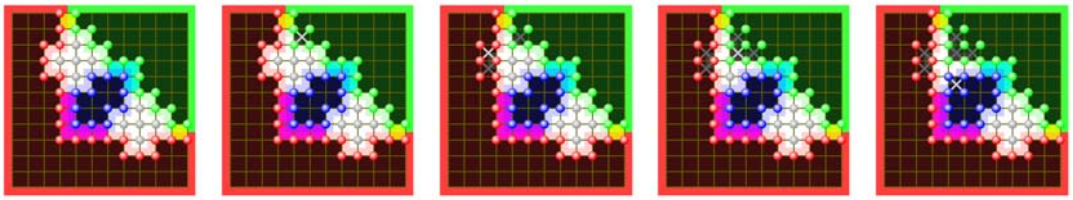


Fig. 6. An overview of our cell thinning algorithm. Colored spheres describes the classified voxels. Especially we note that white spheres are junction voxels. In the first (upper-left) figure, junction cells are described as white squares. According to the progress of thinning process, white squares (cells) are removed and white spheres (vertices) are given another classes.

Basically, our thinning process is described as the simple iteration of the following procedure:

1. If eight voxels of a cell are classified into just two materials and one of them is $b_{junction}$, the cell is removable. In this case, we denote the classes by a list $(b_x, b_{junction})$.
2. If the cell is removable, we give all eight vertices class b_x. As the result, vertices of adjacent cells are also re-classified.
3. For a finale, we remove the cell.

We repeat the iterations until no removable cell exists. When the terminal condition is satisfied, thinning is almost done excepting remarkable cases such as Figure 7.

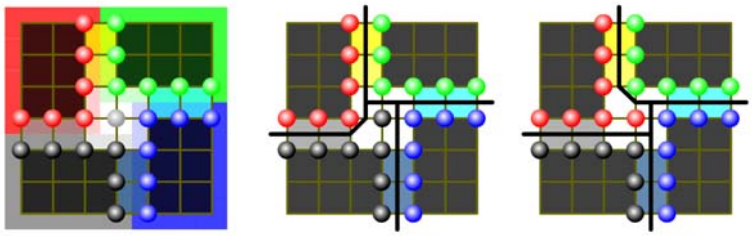


Fig. 7. A two dimensional example of remarkable cases: the terminal condition is already satisfied in the left. However a junction vertex still remains. Actually, we can give an arbitrary class to the vertex and then extract surfaces in those cases. The center and right show two from four possible results.

At this point, we fall all of $\mathbf{C_Y}$ into two categories:

- A cell which eight voxels are classified into two or more classes. The classes do not include $b_{junction}$ and so we can denote the list by $(b_x, b_y, \cdots)$.
- A cell which eight voxels are classified into three or more classes. The classes include $b_{junction}$ and we denote the list by $(b_{junction}, b_x, b_y, \cdots)$. We have to deal these cells as the remarkable cases.

We discuss about cells of the latter category. Those cells have one or more junction vertices. Therefore, we consider how to give an appropriate class to the vertices as shown in Figure 6. Consequently, we apply the following operations to the vertex:

1. We inspect classes of the vertex neighborhoods. The classes may include $b_{junction}$ and one or more classes denoted by a list $(b_x, \cdots)$ in addition to $b_{junction}$.
2. Replace the class of the target vertex by an arbitrary class b from $(b_x, \cdots)$. When the number of $(b_x, \cdots)$ is greater than one, all choices are topologically equivalent.

Applying the above processes, we construct a foreground cells $\mathbf{C_Y}$ for extracting surfaces. By the way, what are represented by $\mathbf{C_Y}$ that we remarked the approximation of desired surfaces? At the area wedged between different two materials, foreground cells well-approximate the boundary surface geometrically and topologically. In other words, we can determine an appropriate threshold value uniquely for those cells and contouring algorithms generate high-precision surfaces of real objects. Next, we examine junction areas. We does not use the information of CT data at the junction areas because CT values in the areas are not credible for their nonlinear behavior. Therefore, we adopt the cell thinning algorithm that assures only topological reasonabilities such as conductivities of surfaces. As the result, $\mathbf{C_Y}$ is not good geometrical approximations in the junction areas.

4. Multi-material Contouring

We constructed a cell model in the previous section, then we construct surfaces which pass through the cells. First, we determine a threshold value for each foreground cell. Next, we apply the existing contouring method to them.

So-called contouring methods require signs and field values to extract surfaces. Figure 8 shows examples of the generalized Marching Cubes table [3]. If vertices are given signs, Marching Cubes algorithm always generates triangles in the cell. Note that the original Marching Cubes is not applicable to non-manifold objects by using only two signs plus and minus. However the generalized Marching Cubes algorithm can generate non-manifold surfaces by using two or more signs.

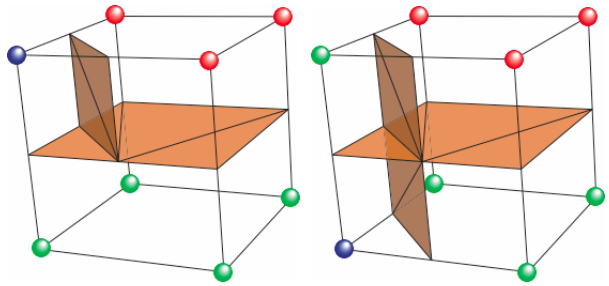


Fig. 8. An optimization of the generalized Marching Cubes table [3]. The original Marching Cubes method [5] is only applicable for two-manifold objects. However by extending to use not only two signs but also more signs, we can create non-manifold surfaces. Three signs are given in this figure: red, green and blue.

Therefore, we determine a sign for each vertex from its material. The determination is very simple. We only have to map a material to a sign one to one. Next, we consider field values for each vertex, because Marching Cubes algorithm and its variants calculate the position of a triangle vertex by linear interpolation of field values of two edge endpoints. If a cell is not derived from junction cells, vertices of the cell are classified by only two materials (b_x, b_y). In this case, we use a threshold value $t_{xy} = (v_x + v_y)/2$ and a field value $\mathrm{f} = |v - t_{xy}|$ for each vertex to contour. If a cell is derived from

junction cells, we consider CT values bound to the cell are incredible. We have no alternative but to use a constant field value f_{const} to contour. In other words, we use trivial boundary mapping [4] which create a triangle vertex on the center of the cell edge. This strategy causes the bad geometrical accuracy of extracted surfaces. However CT data has enough information in junction areas, we consider the problem is insoluble as and when we use only CT data.

After determination of signs and field values, we apply the existing Marching Cubes algorithm to the foreground cells and extract the boundary surfaces.

5. Result

We implemented the proposed methods on a standard workstation equipped with AMD Opteron 1.8GHz and 16GB main memory. We experimented on several parts of a car engine-head as shown in Figure 9. The size of the whole CT data is 500 × 700 × 231 and the resolution is 0.4mm × 0.4mm × 0.5mm.

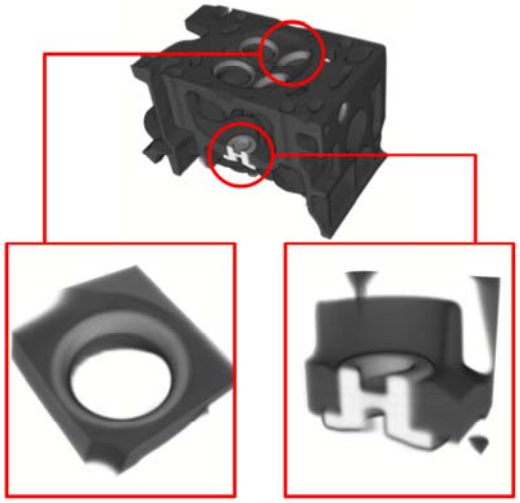


Fig. 9. CT data of a car engine-head which contains *aluminum* parts (gray areas) and *iron* parts (white areas). Specially, we extracted two multi-material areas from the data for experiments.

It took 18 seconds to extract 56,631 triangles from CT data of a manfold part (50 × 90 × 90) as shown in Figure 10. And it took 29 seconds to extract 102,659 triangles from CT data of an attachment part (130 × 120 × 62) as shown in Figure 11. We checked our algorithm can select threshold values properly for surfaces between two materials and checked topologies of the generated surfaces are correctly constructed. In practice, the maximum error of the extracted surface is ± 0.1mm in the boundary areas. In the junction areas, the maximum error is ± 0.5mm (as same as the voxel size). Parameters r_s and r_t are decided as $r_s = R$ and $r_t = 2R$ through experiments. When we adopted bigger parameters, the junction areas grown. Contrarily, when we adopted smaller parameters, the

maximum error of boundary areas enlarged.

We also applied our algorithm to the whole CT data for feasibility study. It took 30 minutes to process the data and about 10 mega triangles are generated (Figure 12).

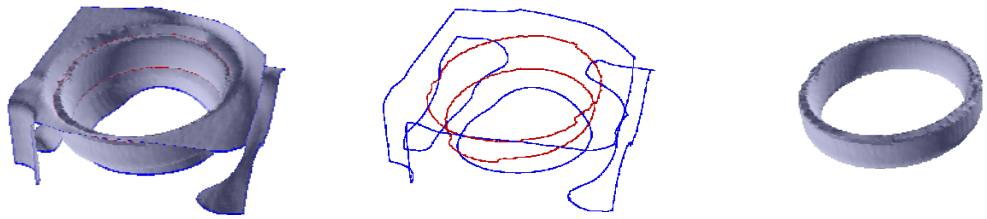


Fig. 10. The left is the extracted polygon model of the manfold part. An iron ring are knocked into an aluminum manfold. The center shows non-manifold edges in red and boundary edges in blue. The right shows only the iron ring.

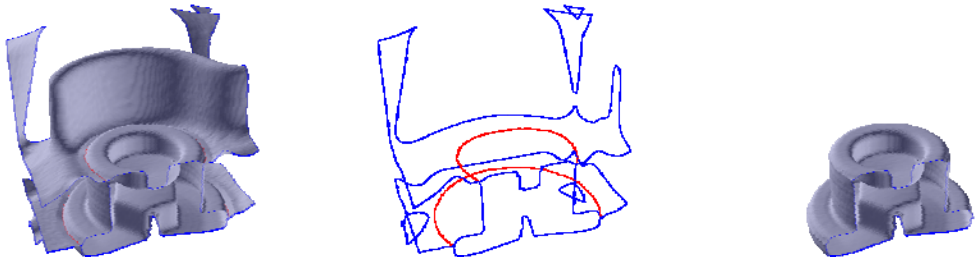


Fig. 11. The left is the extracted polygon model of the attachment part. As same as shown Figure 10, an iron attachment is fit in an aluminum engine-head. The center shows non-manifold edges and boundary edges. The right shows only the iron attachment.

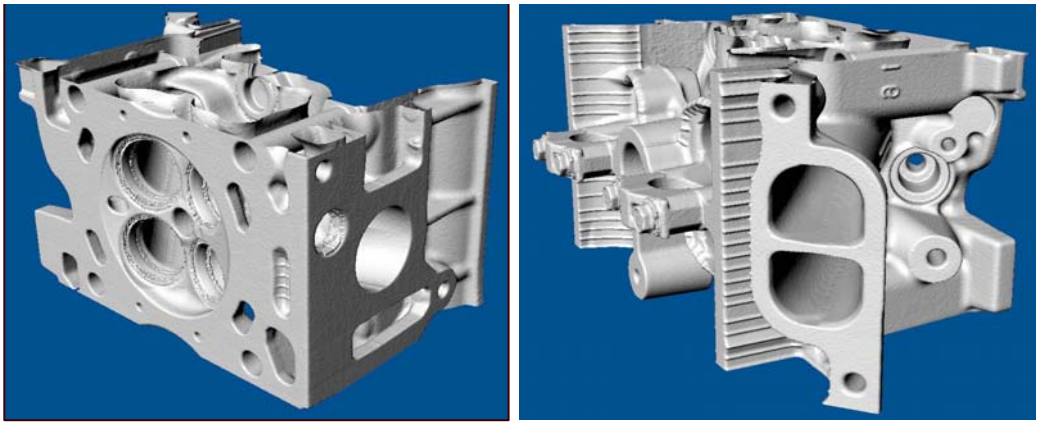


Fig. 11. These are the whole polygon model of the engine-head.

6. Conclusion and Future Works

We presented a new method for extracting surfaces from multi-material CT data. Our algorithms are based on three dimensional image analysis and aimed to solve the problems caused by fundamentals of CT itself. First, we applied a voxel based classification method to the data to detect boundary areas between two materials. Secondly, we proposed a cell thinning algorithm to solve problems at junction areas. Thus we could produce a cell model approximate desired surfaces. And last, we determined signs and field values for each vertex and applied the existing contouring algorithm to extract surfaces.

Some of the problems to be resolved in future work include the issue of the accuracy of the junction areas. We consider we cannot improve the precision without another information except for CT data. Therefore, we plan to add some knowledge when extracting surfaces of junction areas.

7. Acknowledgements

We would like to thank Toyota Motor Co., Japan for motivating us to conduct this research and providing suggestions and comments. This research is funded by Grant-in-Aid for Scientific Research by Japan Society for the Promotion of Science, No. 15360080.

References

1. Adolfsson, J. and Helgesson, J. (2004), Generation of smooth non-manifold surfaces from segmented image data. Master's thesis, Lund Institute of Technology.
2. Fujimori, T., Suzuki, H., Kobayashi, Y., and Kase, K. (2005), Contouring medial surface of thin plate structure using local marching cubes. *Journal of Computing and Information Science in Engineering* (Short Paper).
3. Hege, H.-C., Seebass, M., Stalling, D. and Zöckler, M. (1997), A generalized marching cubes algorithm based on non-binary classifications. Technical report, Konrad-Zuse-Zentrum für Informationstechnik Belin.
4. Lachaud, J.-O. and Montanvert, A. (2000), Continuous analogs of digital boundaries: A topological approach to isosurfaces. *Graphical Models and Image Processing* 62, 129–164.
5. Lorensen, W. E. and Cline, H. E. (1987), Marching cubes: A high resolution 3d surface construction algorithm. In Proceedings of the 14th annual conference on Computer graphics and interactive techniques, 163–169.
6. Nielson, G. M. and Franke, R. (1997), Computing the separating surface for segmented data. In VIS '97: Proceedings of the 8th Conference on Visualization '97, 229–233.