Non-manifold Implicit Surfaces Based on Discontinuous Implicitization and Polygonization

Shuntaro Yamazaki
University of Tokyo, RIKEN
shun@cvl.iis.u-tokyo.ac.jp

Kiwamu Kase
RIKEN
kiwamu@riken.go.jp

Katsushi Ikeuchi
University of Tokyo
ki@cvl.iis.u-tokyo.ac.jp

January 14, 2004

Abstract

Implicit surfaces in 3D geometric modeling are limited to two manifolds because the corresponding implicit fields are usually defined by real-valued functions which bisect space into interior and exterior. We present a novel method of modeling non-manifold surfaces by implicit representation. Our method allows discontinuity of the field function and assesses the special meaning of the locus where the function is not differentiable. The enhancement can yield a non-manifold surface with such features as holes and boundaries. The discontinuous field function also enables multiple classification of the field, which makes it possible to represent branches and intersections of the implicit surfaces. The implicit field is polygonized by the algorithm based on the marching cubes algorithm, which is extended to treat discontinuous fields correctly. We also describe an efficient implementation of converting a surface model into a set of discrete samples of field function, and finally present the result of the non-manifold surfaces reproduced by our method.

Keywords: implicit surface, non-manifold, implicitization, polygonization, segmented distance, vertex generation diagram

1 Introduction

Volumetric representation of surface models is beginning to play an important role in areas of geometric processing such as blending, deformation, and boolean operation. The major advantage of volumetric representation over explicit surface representation is uniformity in three-dimensional space and scalability for large and detailed models. In the volumetric representation, models are defined as arrays of volume elements (*voxels*) placed on a regular grid, which makes it easy to process complicated objects with constant time and space complexity.

One common approach to encoding surface models into volumetric structure is to employ an implicit surface[2]. An implicit surface is originally defined as an isosurface of a real-valued function in three-dimensional space. Given a surface model, first it can be converted into the functional representation by the techniques called *implicitization*, and then the implicit field can be converted again into the explicit representation, such as a triangular mesh, through processing called *polygonization*.

Since the isosurface of a continuous function is manifold in principle, most existing methods guarantee that the resulting triangulated surface is manifold, which means it has neither holes, boundaries nor dangling faces. On the other hand, if the original surface is non-manifold, it can be modified or changed topologically into manifold ones. Since

many of the existing surface models in computer graphics and computer-aided designs are not necessarily manifold, the scope of the implicit surface has been restricted.

In this paper we propose a method of representing the non-manifold implicit surface within the framework of a scalar field and polygonizer based on the traversal of hexahedral cells. A surface model with arbitrary topology can be converted into volumetric representation of the scalar field with little loss of topology, and then our polygonizer can reproduce the surface approximately in the form of triangular meshes.

The major contributions of this paper are as follows.

- We propose a novel method of converting surface models into a volumetric data structure which can represent non-manifold features correctly. The volumetric data is a set of discrete samples of the *segmented distance field* which is designed to take discontinuity and multiple classification into account. We also present an effective implementation of the distance transformation of given surface models by making the best use of hardware acceleration in common graphics hardware.

- An algorithm for converting the distance field described above into the triangular mesh is proposed. While it is based on the standard polygonization technique known as the marching cubes methods, non-manifold features such as holes, boundaries and intersections can be produced correctly by our method. Compared with the original algorithm, our approach generalizes the interpolation step during polygonization by adopting the novel look-up table called the *vertex generation diagram* which makes it possible to generate an interpolated vertex at an arbitrary position on the edge of the hexahedral cell.

2 Related work

The implicit surface is one area of geometric modeling and has been studied widely by many researchers [2]. The application ranges over such fields as modeling, deformation, animation, blending and rendering. Since the implicit surface is defined as the isocontour of a continuous function in three-dimensional space, it is inherently restricted to the manifold surface. Therefore most of the existing research assumes that the surface is manifold.

When we process a given surface model by the volumetric representation, first it is necessary to convert the model into volumetric data set by means of the distance transformation. However, the computational cost for calculating the exact distance is high since the distance transformation is a representation of the Voronoi division of the space.

The major approach to achieving a fast distance transformation algorithm is to use approximations of the Euclidean metric, such as the chessboard metrics [19], and the chamfer distance [4]. With these metrics, the value associated with a voxel can be derived from the values of its neighbors, which allows these algorithms to work in two raster scans over the data set. One major problem of these algorithms is the low accuracy of the result, but this can be improved by considering the vector distance as proposed in [7, 8, 12].

The data set created by the methods listed above is the signed distance field, which can represent only the closed manifold model, as explained in section 3.1. In order to represent a non-manifold surface correctly in the framework of the distance field, we employ not the signed distance but the *segmented distance* described in section 3.3.

While the implicit representation itself can be processed directly for such operations as deformation and rendering, surface-based operations such as shape optimization or modeling often need explicit parameters which define the shape and position of the surface. Bloomenthal and Ferguson [3] proposed a method for the polygonization of non-manifold implicit surfaces using tetrahedral cell decomposition. Dae et al. [13] also used tetrahedral cells to model a non-manifold surface by the accumulative construction scheme similar to CSG representation. Hege et al. [11] described a method for generating non-manifold surfaces from a scalar field using hexahedral cell decomposition based on the marching cubes method [15]. All of these methods take multiple classification into account and treat the intersection of surfaces correctly, however the other types of non-manifold features, such as

the discontinuity of surface, are not considered. In our method, such degenerations of the surface can be successfully represented by restricting the interpolation of the field function, as described in section 4.2.

3 Implicitization

3.1 Implicit surface

Given a surface $S \subset \mathbf{R}^3$, the *implicit surface* representation of S is the combination of the scalar value t and function f such that

$$p \in S \quad \Leftrightarrow \quad f(p) = t, \tag{1}$$

where $p \in \mathbf{R}^3$ is the point in three-dimensional space.

f can be defined as a distance between p and the surface, which means f is not unique for a given S. One common choice is to use the *signed distance* and build a *signed distance field* where each point p is assigned a distance from the surface. The function returns a real-valued distance of positive sign when p is outside the region enclosed by S and negative sign when p is inside S. When used in the computer, the implicit fields are usually represented by not a function but a set of discrete points on a regular grid in three-dimensional space.

An example of the signed distance field in two dimensions is shown in Figure 1. Broken lines denote the original surface S. For each grid point p the nearest point q on S is calculated as illustrated by arrows, then the distance between p and q is assigned the same sign as the dot product of the normal of the surface at q and vector $p - q$.

For geometric modeling and processing, it is useful to approximate the surface with an explicit representation, such as the triangular mesh. The marching cubes method [15] and its extensions [5, 16] are the standard techniques used to extract the isosurface from a scalar field. It enumerates all cases of surface intersection with the cell composed of eight adjacent voxels, and then produces triangulated isosurface patches by means of a look-up table and linear interpolation. This method guarantees that the topology of the resulting isosurface is *manifold*, except the boundary of the scalar field.

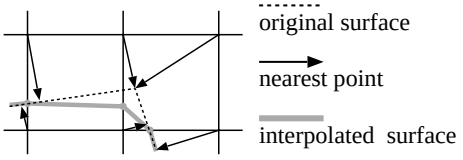

Figure 1: The isosurface reconstructed from the sampled distance field by means of linear interpolation can differ from the original.

3.2 Non-manifold surface

In three-dimensional space, surface S is two-manifold if and only if the infinitesimal neighbourhood around any point on S is topologically equivalent to a two-dimensional disk. In other words, exactly two different directions can be chosen as the axes of a two-dimensional local coordinate system for any point on the surface.

Conversely, S is non-manifold if and only if the number of directions which can be chosen on a point on S is less than two, or more than two. Therefore surface S is regarded as non-manifold in one of the two following cases.

1. **Discontinuity of surface connection:**
   There is at least one point $p \in S$ around which only one or no direction can be chosen in the local coordinate system. This case occurs when a surface has boundaries or holes. An example in two dimensions is illustrated on the left in Figure 2.

2. **Ambiguity of surface orientation:**
   There is at least one point $p \in S$ around which more than two directions can be chosen in the local coordinate system. This case occurs when the surface has branches or dangling faces. An example in two dimensions is illustrated on the right in Figure 2.

3.3 Enhancement of distance field

The approximate surface extracted from the sampled distance field is not necessarily close to the original because the geometric parameters of the surface are estimated by linear interpolation of the discretized distance field which usually does not vary linearly, particularly when the surface is not

differentiable. An example in two dimensions is shown again in Figure 1 where the interpolated surface is not close to the original surface in the neighborhood of a sharp edge, but the topology of the interpolated surface is still the same as the original. Although generating the isosurface by linear interpolation may deteriorate the accuracy of geometry, the fidelity of topology can be preserved as long as the interval of sampling is sufficiently small. In addition, the accuracy of geometry in this case can be improved by adopting the enhanced distance field proposed by Kobbelt et al. [14]

On the other hand, if the original surface is non-manifold, the generated surface is drastically modified and far from the original with respect to not only geometry but also topology.

In Figure 2, the left figures show examples of non-manifold surfaces due to the discontinuity of surface connection in two dimensions. In the upper left, the boundary point where the original line breaks off is incorrectly extended by unnecessary interpolation, which results in a completely different shape. The unnecessary vertex and faces connected to it should be removed to keep the topology correct, as shown on the lower left in Figure 2. This modification can be accomplished by not applying interpolation to the pair of regions whose distances differ by more than the width of a voxel.

The right drawings in Figure 2 illustrate the examples of non-manifold surfaces due to ambiguity of surface orientation. Because the implicit surface assumes binary classification of the field, it cannot treat the branches of the surface inherently. Introducing multiple classification makes it possible to treat such situations as pointed out by Bloomenthal [1]. In this method, the additional information on classification is attached to the field function. Different from Bloomenthal's work, in this paper, we propose a novel distance which enables multiple classification in the framework of the single real-valued distance function.

Provided that the distance field can be separated into regions in which the original surface forms a portion of the boundaries, each region can be distinguished by assigning a real number belonging to a different range to each region.

The signed distance is one such distance that distinguishes two different regions, namely, interior and exterior. The value in the distance field can

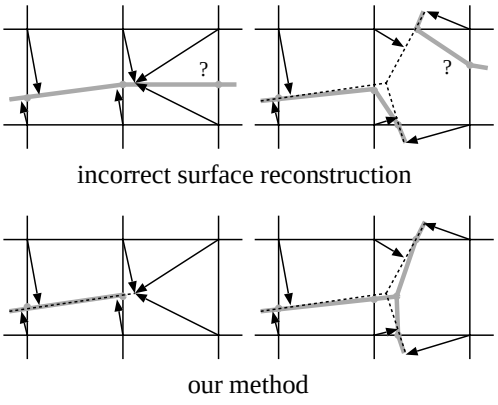

Figure 2: The upper left shows an example of incorrect surface reconstruction due to the discontinuity of the surface connection, which has been corrected by preventing unnecessary interpolation in the lower left. The upper right shows another example of incorrect reconstruction due to the ambiguity of surface orientation. This problem is caused by binary classification of the distance field, and is successfully solved by considering multiple classification by introducing the segmented distance field which is explained in section 3.3.

be classified into positive and negative regions. The absolute value of distance is equal to the Euclid distance from the boundary. Viewed from the representation of real values in the computers, this fact corresponds to the sign bit (MSB) of a signed distance being assigned to represent the indices of regions.

For the multiple classification, we must adopt another distance which makes it possible to distinguish the arbitrary number of regions. Accordingly we introduce a distance defined only within some intervals whose values are real numbers, which we call a *segmented distance*. In the computer, the segmented distance can be represented by using some higher bits of a number as the index of the region. This assignment is equivalent to modifying the distance so that it starts from a certain value which differs for each region and increases in accordance with the distance from the boundary of the region.

The segmented distance must be defined for each model according to the number of regions into which the field can be classified. Since each point on the surface can exist on the boundary between

different regions unless the point is the junction of the surface, we can classify the field by segmenting the surface into patches which have no junction edge. The classification which satisfies the abovementioned condition is not unique. It may be feasible to compose the segmented distance manually if the surface model is sufficiently simple, but for complex models, it is difficult or almost impossible to determine the number of regions necessary to define the segmented distance. Therefore we describe a simple method of classifying the field automatically.

First, if the junctions exist on the surface, we divide the surface into several patches along the diverging lines. Then both front and back faces for each patch are assigned different indices which indicate the classification. When the segmented distance field is calculated, the distance function $f(p)$ for the point p in the field is defined by both the index n of the nearest point and the distance D_n from the point. Suppose that a sufficiently large number d is selected as the interval of the region, then the distance is calculated by

$$f(p) = min(D_n, d) + r_n \tag{2}$$

where $r_n = d \cdot n$. f can be regarded as the distance with the offset r_n assigned to each region. d should be larger than the width of the distance field, and r_n must differ by at least d between one another in order to identify each region by f uniquely.

As a result of the discontinuity of the segmented distance, some values for the segmented distance can be undefined. In addition, such numerical operations as addition, subtraction and scaling of the segmented distances may yield undefined values, even if r_n and d are properly defined. Consequently the interpolation which is necessary to extract smooth surfaces cannot be performed on the distance values themselves. Therefore the segmented distance field must be constrained not to be interpolated, by controlling the process of distance interpolation, as described in the next section.

4 Polygonization

4.1 Definition of terms

Our polygonization algorithm is based on the marching cubes technique which sequentially processes the cells composed of eight adjacent voxels, as shown in Figure 3. To clarify our explanation, we define the following terms before we discuss the polygonization of the distance field.

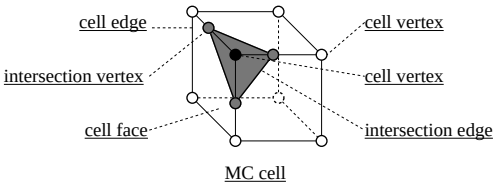

Figure 3: MC cell

MC cell: logical cube created by eight adjacent voxels

cell vertex/edge: vertex/edge of MC cell

intersection vertex/edge: approximate intersection point/line between MC cell and the actual surface

4.2 Interpolation using VGD

Polygonization in the methods based on the marching cubes algorithm essentially consists of two steps. In the first step, an isocontour value is defined by the user and all cubes that are intersected by the surface are identified. In the next step, those cubes in the boundary set are examined and a set of connected polygons are produced. Although some of the polygonizing methods assume binary data sets and yield triangles only with the precalculated normal [10, 18], it is preferable to interpolate field values to generate smoothly connected triangles.

The position of the intersection vertex is determined by linear interpolation of field values assigned to cell vertices at both ends of the MC edge. The interpolation can be written as

$$p = \begin{cases} \dfrac{t-u}{v-u} & \text{if} \quad t \in [u, v],\ u \neq v, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where t is the isosurface value, u and v are field values on each MC vertex, and p is the position of the intersection vertex, which is measured from the MC vertex whose value is u. p is normalized to satisfy $0 < p \leq 1$, and $p = 0$ means that no intersection vertex is generated on the edge.

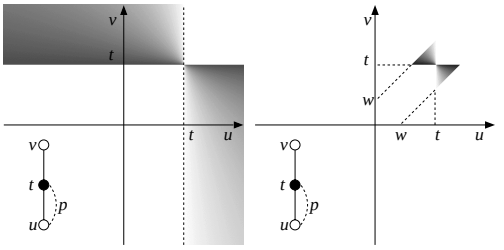

Figure 4: The vertex generation diagrams. Left: standard marching cubes method. Right: marching cubes method considering discontinuity of field function.

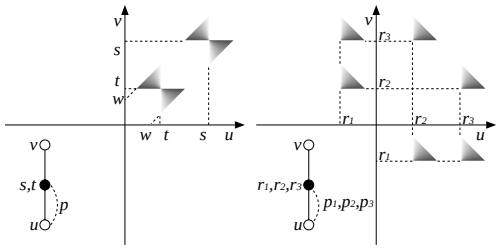

Figure 5: Left: A vertex generation diagram to generate boundary edges. Right: A vertex generation diagram to distinguish three different regions and generate the surface on their boundaries.

As mentioned in section 3.3, if we do not take discontinuity of the distance field into account, the resulting isosurface may have incorrect topology as well as incorrect geometry. Therefore we constrain the interpolating process by introducing the *vertex generation diagram (VGD)*, that is, the two-dimensional chart which determines the place at which to generate an intersection vertex on the MC edge. The diagram takes two field values u,v and yields the position of the intersection vertex in the form of the normalized parameter p described above.

The VGD corresponding to equation (3) is drawn on the left in Figure 4. The dark shading of the diagram indicates the position of the interpolated vertex within the edge.

Discontinuity of the implicit field, as mentioned in section 3.2, can be represented by cutting the diagram off with a certain threshold w, as shown on the right in Figure 4. The equation corresponding to this diagram is

$$p = \begin{cases} \dfrac{t-u}{v-u} & \text{if} \quad t \in [u,v],\ 0 < |v-u| < w \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where w is regarded as the tolerance of continuity, which is typically set to the width of one voxel.

Two or more VGD can be superimposed to extract multiple isosurfaces for different isosurface values. The left diagram in Figure 5 shows a VGD used to extract two different isosurfaces from a signed distance field. It is worth noting that two or more intersection vertices can be produced on a MC edge if the diagrams overlap one another.

An example VGD for a segmented distance which has three segments is presented on the right in Figure 5. Three regions are distinguished in this diagram, and the boundary surface for each pair of regions can be extracted. This diagram corresponds to the equation

$$p = \begin{cases} \dfrac{u-r_i}{(u-r_i)+(v-r_j)} & \text{if} \quad \begin{array}{l} u \in [r_i, r_i+d], \\ v \in [r_j, r_j+d], \\ 0 < (u-r_i)+(v-r_j) < w, \end{array} \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

where $i, j = 1, 2, 3$ and $i \neq j$ since the field is classified into three regions. To extend the equations to n regions, we should define the segmented distance such that r_i differs by more than the size of the field d for each $i \in [1, n]$, then the corresponding VGD can be generated by equation (5).

4.3 Triangulation of MC cell

After determining on which edges intersection vertices are generated, a set of triangles are produced for each MC cell. When the volume data set has n different regions, the fact that eight cell vertices can take at most n different states means that there are $min(n, 8)^8$ different ways in which a surface can intersect a MC cell.

In our method, however, the classification of cell vertices cannot determine the connectivity of intersection vertices uniquely because the vertex generation diagrams do not guarantee that the cell edge which intersects different regions has intersection vertices. Hence we adopt another method. Provided that an implicit surface actually exists on the

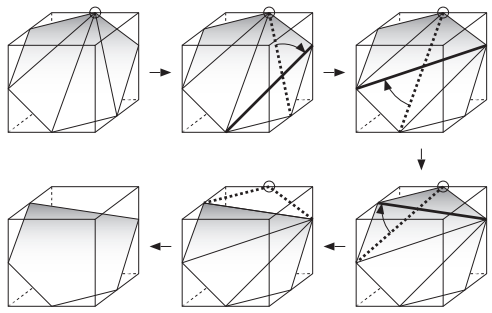

Figure 7: Candidates of intersection vertices are first checked to determine whether there are actual intersection vertices. If a vertex is determined to be removed, the edges connected to the vertex are flipped so that the vertex has only two edges. Then the triangle which is connected to the vertex is removed.

boundary between two different regions, the triangular patch to be generated is part of the surface patch generated using the existing cell-based triangulation method, such as the marching cubes method [15] for binary classification, and the generalized marching cubes method [11] for nonbinary classification. Therefore, first we generate all vertices on the cell edges which intersect different regions, and then remove any that are not actual intersection vertices. In our current implementation, we use the look-up table based on the generalized marching cubes method proposed by Hege et al. [11]. The look-up table for binary classification is show in Figure 6.

The simplest way of removing nonintersecting vertices from the initial vertices is to remove all faces which include the nonintersecting vertices. This method, however, may decimate all faces in the MC cell in the worst-case scenario, which causes jagged boundaries of the surface. Therefore, we must prevent triangles from being excessively eliminated by edge flipping. Vertex removal proceeds in a vertex-by-vertex manner, as shown in Figure 7, and the detailed algorithm is described as follows.

Triangle Removal in MC Cell

for all vertices *v* on a cell edge **do**
  if *v* is NOT an intersection vertex **then**
    while *v* is shared by more than two edges **do**
      Select one edge *e* which is connected to *v* and shared by more than two triangles.
      Flip *e*.
    end while
    Remove the triangle *t* connected to *v*.
  end if
end for

According to the look-up table shown in Figure 6, we must generate the intersection vertex inside the MC cell, for example, in the case of the lower rightmost diagram in Figure 6. The vertices inside the cell should connect directly to the intersection vertices so that no isolated triangle is generated during triangle removal.

5 Implementation details

5.1 Fast distance transformation

The Euclid distance is often employed as the distance between a point and a surface, because it spreads smoothly in every direction, which is often convenient for performing such operations as blending or deformation. Computing the Euclid distance accurately, however, leads to a prohibitive computational cost which increases cubically to the size of the data set. For the purpose of isosurface extraction with respect to a fixed isosurface value, the Euclid distance is not necessarily required. A simpler distance, such as the distance measured along a certain direction, may be sufficient.

We implemented a simple and effective implicitizer which converts a surface model into a segmented distance field, where the distance is determined by the depth measured from different views of the original surface model. This approach can make the best use of hardware-accelerated z-buffers which are available on a standard PC graphics card.

Assuming that the entire surface to be converted into the distance field is enclosed by a bounding box, a depth buffer is generated for each face of

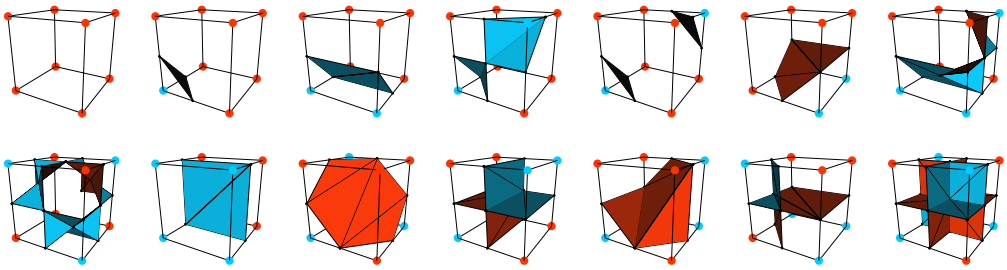

Figure 6: If the field function can be classified into three regions or less, the initial triangles in the MC cell are generated using a look-up table which has been created by symmetrically modifying the look-up table for the original marching cubes algorithm. The triangles can be modified and removed according to the field values assigned to the MC vertices. This figure shows the look-up table used in our method for binary classification, which is a portion of the table proposed in [11]

the box by rendering the model onto it using orthographic projection. Then we can determine the distance at the position of each voxel from the six z-buffers by taking the minimum positive value as the distance to the surface. Negative distance is always unreliable because the surfaces occluded by another surface are ignored. One solution to this problem is to assign a weight to each depth measured in six directions according to both the sign of the distance and the normal of the surface, as proposed by Curless et al. [**?**]

Since positive distance is always accurate in our situation, we adopt one of the distances with positive sign at each voxel where the normal of the surface is the nearest to the viewing direction, while the weighted sum of distance is used if all distances are negative. The normal of the surface is also estimated, using graphics hardware, by shading the surface by placing a point light source far in the viewer's direction. The brightness of the rendered pixel is approximately in proportion to the cosine of the angle made by the surface normal and the view direction, which can be used as the weight in the estimation of distance at the voxel.

In order to distinguish the boundaries in a field classified into more than two regions, we also use the color of the surface. According to the classification, the surface is rendered using different colors assigned to each side of the surface for each pair of regions. Before the surface is implicitized, the field must be classified correctly either manually or automatically by using existing methods such as that proposed in [17]. As for the triangular meshes, we assume that the degeneration described in section 3.2 occurs only on the edge of the faces. For instance, two triangles may intersect each other along a line other than an edge, which, however, can be detected and solved by subdividing the triangle in advance.

This method may fail to yield the exact distance of a complicated or concave surface. If part of the surface is not visible, the area will not be properly measured. However most surfaces will be converted successfully. If the result is not acceptable, adopting a more general solution such as that proposed by Chen et al. [6] may be possible.

5.2 Using VGD as a look-up table

The calculation of intersection vertices can be accelerated by using discretized VGD as a look-up table, since a large portion of the time in the marching cubes algorithm is spent on the interpolation of MC vertices.

Although the range of the field function value is generally too wide to keep the tabular VGD sufficiently small to store in the computer, we can discretize the VGD in some situations.

- The volumetric data set obtained by scans such as CT imaging or MRI are often defined as a scalar field represented by 8- or 16-bit integers, in which case we can utilize a tabular VGD to accelerate surface extraction.

- If the VGD has spatial coherency, we can compress the size of the diagram by segmenting it

into small blocks. For example, as shown on the right diagram in Figure 5, a large portion of the diagram is empty and can be ignored. Since the effectual values are distributed in the same manner around each intersection point of r_i and r_j for $i \neq j$, it is sufficient to store the block only once. If the size of the block is sufficiently small, it is feasible to create a look-up table by sampling only the block.

6 Experimental result

Figure 8 shows the results of implicitization and polygonization of three different surface models. Although our method of implicitization is not restricted to polygonal models, these models are all composed of triangular meshes because the implicitization described in section 5.1 utilizes the graphics library which can only render triangles at present.

The left models in Figure 8 are the original surface models and the right ones are the resulting meshes. The original models are first converted into volumetric data sets and then triangulated into polygonal models by the method proposed in this paper.

The model of a cat shown in the first row was generated by scanning a real object and then converted into a triangular mesh. There is no face at the bottom of the original model. The boundary edges are successfully reproduced in the generated model. In the second row, the model of a monitor created using a 3D modeler for the use of rendering is presented. The model has more than one component and has boundary edges. In addition, as is often the case for models created by CAD modeler, the model has numerous small holes on the surface because detailed shapes are modeled separately and then put into a common space without information concerning their connectivity. The holes can be eliminated by appropriate sampling into volumetric representation. In the third row, a sample model with three different regions is presented. The branches of the surface are successfully reproduced as shown in the right figure.

7 Conclusions and future work

We have presented an enhancement of the implicit surface to handle two-dimensional non-manifold surfaces embedded in three dimensions. The difficulty of non-manifold representation in the framework of an implicit surface is classified into two cases: discontinuity of surface connection and ambiguity of surface orientation. Our method can treat a discontinuous distance field properly and prevent undesirable interpolation. Generalizing the process for interpolation by introducing the vertex generation diagram (VGD) enables not only boundary representation but also nonbinary classification which is necessary to represent branches of faces. Also we presented an effective method of implicitizing given surface models easily by taking advantage of hardware acceleration.

One problem in the polygonization methods based on the marching cubes algorithm is that the accuracy of geometry and topology depends on the coordinate system of sample points as well as their intervals. The geometry of the surface can be improved by introducing the directed distance field representation proposed by Kobbelt et al. [14]. As for topology, topological feature tracking can be taken into account by, for example, the method proposed in [20]. Both problems still remain for future work.

Although the accuracy of geometry and topology can also be improved by increasing the sampling ratio of the distance field, processing a volumetric data set with high quality is often intractable because the data size increases cubically. To avoid this problem, it is effective to introduce hierarchical representation such as the octree. There is also an advantage in introducing hierarchical representation with respect to adaptive refinement of detail. We are planning to modify our polygonization method to process the octree correctly in a similar manner as in [9].

For the use of rendering, an implicit representation of a surface can be displayed not only by conversion into parametric representation but also directly, for instance, by ray tracing. The discontinuous implicit field and the vertex generation diagram are applicable to these methods.

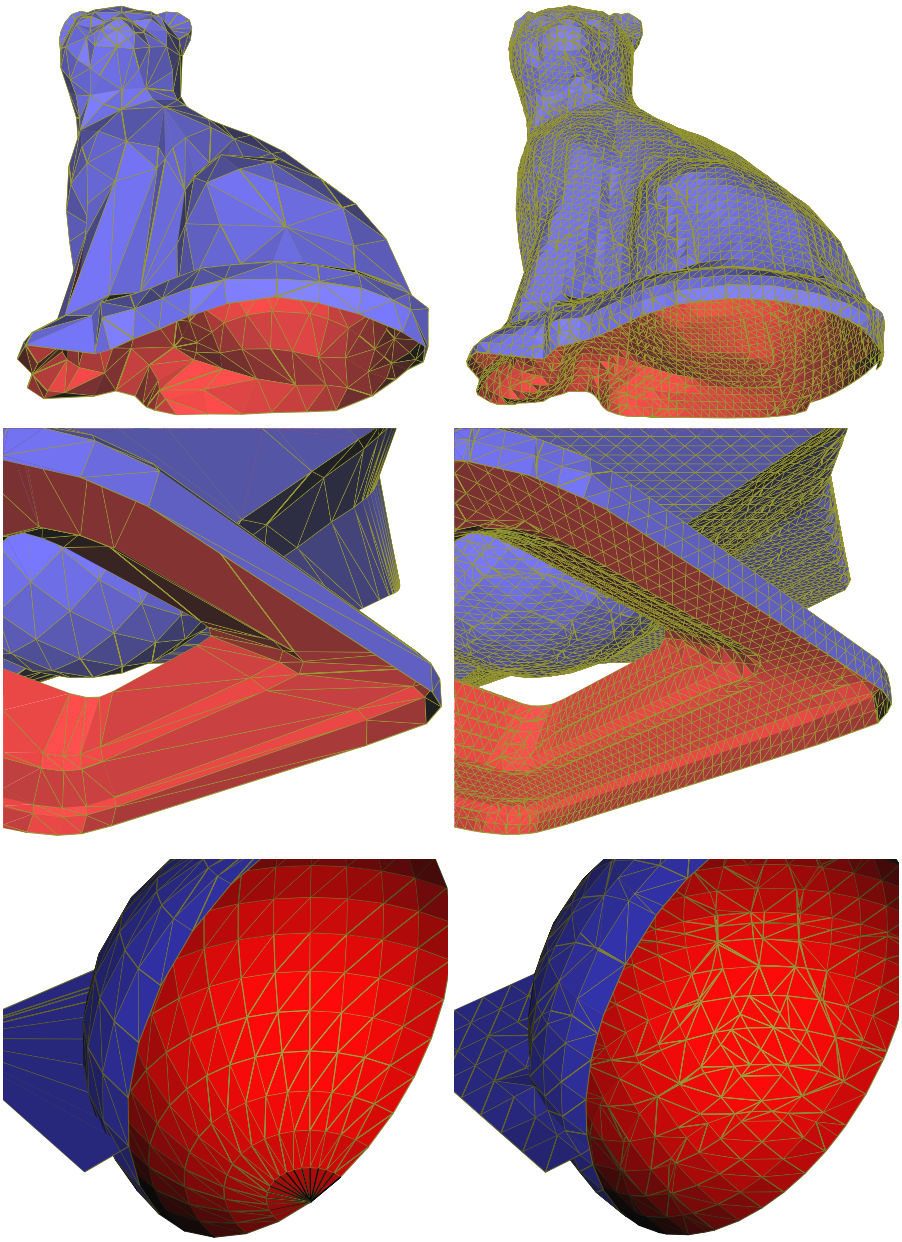

Figure 8: Left: the original models. Right: the polygonized implicit surfaces generated by our method. The original models are first converted into volumetric representation by the discontinuous implicitization, then polygonized again into the triangle models shown on the right. In the top row, boundary edges could be successfully reproduced. In the middle row, two separate surfaces, one manifold and the other non-manifold, are reproduced. In the third row, three different regions are successfully separated.

8 Acknowledgement

We thank Akitake Makinouchi, Hiromasa Suzuki, Takashi Kanai and Yoshinori Teshima for their comments and guidance. We are indebted to RIKEN and Japan Science and Technology Corporation for their support of this research.

References

[1] Jules Bloomenthal. Polygonization of implicit surfaces. *Computer Aided Geometric Design*, 5:341–355, 1988.

[2] Jules Bloomenthal. *Introduction to Implicit Surfaces*. Morgan Kaufmann Publishers, Inc., 1997.

[3] Jules Bloomenthal and Keith Ferguson. Polygonization of non-manifold implicit surfaces. In *Proc. SIGGRAPH*, pages 309–316, 1995.

[4] G. Borgefors. Distance transformations in digital images. *CVGIP*, 34:344–371, 1986.

[5] Montani C., Scateni R., and Scopigno R. A modified look-up table for implicit disambiguation of marching cubes. *Visual Computer*, 10(6):353–355, 1994.

[6] Hongsheng Chen and Shiaofen Fang. Fast voxelization of three-dimensional synthetic objects. *Journal of Graphics Tools*, 3(4):33–45, 1998.

[7] O. Cuisenaire and B. Macq. Fast and exact signed euclidean distance transformation with linear complexity. In *ICASSP '99*, volume 6, pages 3293–3296, 1999.

[8] P. E. Danielsson. Euclidean distance mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.

[9] S.F. Frisken, R.N. Perry, A.P. Rockwood, and T.R. Jones. Adaptively sampled distance fields: A general representation of shape for computer graphics. In *Proc. SIGGRAPH2000*, pages 249–254. ACM, July 2000.

[10] D. Gordon and J.K. Udupa. Fast surface tracking in 3D binary images. *Computer Vision, Graphics and Image Processing*, 1989.

[11] Hans-Christian Hege, Martin Seebas, Detlev Stalling, and Malte Zockler. A generalized marching cubes algorithm based on non-binary classifications. Technical report, Konrad-Zuse-Zentrum fur Informationstechnik (ZIB), 1997.

[12] M. W. Jones and R. Satherley. Voxelisation: Modelling for volume graphics. In *Vision, Modeling, and Visualisation 2000*, pages 319–326, 2000.

[13] Dae Hyun Kim, Ulf Doering, and Beat Bruderlin. Polygonization of non-manifolds with the aid of interval operators. In *Proc. Implicit Surfaces*, pages 145–151, 2000.

[14] Leif Kobbelt, Mario Botsch, Ulrich Schwanecke, and Hans-Peter Seidel. Feature sensitive surface extraction from volume data. In *Proc. SIGGRAPH*, 2001.

[15] William Lorensen and Harvey Cline. Marching cubes: a high resolution 3D surface reconstruction algorithm. In *Proceedings of the SIGGRAPH annual conference on Computer graphics*, 1987.

[16] Nielson G. M. and B. Hamann. The asymptotic decider: Resolving the ambiguity in marching cubes. In *Visualization '91*, pages 83–91, 1991.

[17] Alan P. Mangan and Ross T. Whitaker. Partitioning 3D surface meshes using watershed segmentation. *IEEE Transactions on Visualization and Computer Graphics*, 5(4), 1999.

[18] C. Montani, R. Scateni, and R. Scopigno. Discretized marching cubes. In *Proc. Visualization*, pages 281–287, 1994.

[19] A. Rosenfeld and J. L. Pfaltz. Distance functions on digital pictures. *Pattern Recognition*, 1(1):33–61, 1968.

[20] Barton T. Stander and John C. Hart. Guaranteeing the topology of an implicit surface polygonization for interactive modeling. In *SIGGRAPH*, 1997.