Protrusion Based Segmentation of Complex 3D Shape Models

Masaki Aono*, Shiro Wakida*, and Atsushi Tatsuma*,
* Toyohashi University of Technology, Aichi, Japan
E-mail:aono@tut.jp Tel: +81-532-44-6764

***Abstract*—3D shape models have been used in many fields. Segmentation of 3D model is a technology that allows us to split the model into multiple semantically meaningful parts, for reuse and partial shape retrieval. Our proposed method hypothesizes that any complex 3D shape models consist of "protruded" parts and the remaining body. To determine the border between "protruded" parts and the remaining body, we need the distance measure. In this paper, we first describe a couple of well-known distance measures, then we propose a new method for approximating a geodesic distance using "landmarks", which we prove to be much efficient than other distance measures. We also propose a new method for computing boundaries between segments. We have used a segmentation benchmark data set to evaluate the validity of our methods. The results we have obtained so far are very encouraging.**

I. Introduction

3D digital models have been used in many fields including industrial product design and simulation, virtual medical operations, education, and entertainments such as movies and games. However, it is very expensive to create a new 3D model from scratch, and it has been mostly neglected to reuse existing 3D models initially made for other purposes. For this reason, research on 3D shape retrieval has been actively investigated in the past decade. Yet not much attention has been paid to 3D partial shape retrieval to date, compared to the whole shape retrieval. To make it possible to do partial shape retrieval, it is very important to subdivide a whole 3D shape models into a collection of meaningful parts. The subdivision of 3D shape is commonly called "3D segmentation". Ever since Princeton University made public "3D segmentation benchmark" [1], the research on segmentation has become much easier, since we can take advantage of the manually tagged "correct" segmentation results in order to evaluate the results. The segmentation of a 3D model takes two inputs (3D shape and k, the number of segments to be produced) and outputs k-grouped segments, where each segment is assumed to consist of a collection of 3D polygonal meshes.

In this paper, we hypothesize that a 3D shape to be segmented is usually complex and has one or more "protruded" parts within itself. Under this assumption, we propose an algorithm to repeatedly extract a segment based on protrusion hypothesis. We also propose an approximate method for computing a geodesic distance, by introducing the notion of "landmarks", needed to determine the end points, which are supposed to locate the outermost part of the protruded segment in concern. The evaluation of segmentation is done by several standpoints. One of them is the beauty of the boundary between segments. In this respect, we propose a method for determining the candidates polygons on the boundary, by considering both distance histograms and the distribution of sliced areas from one "landmark" to the neighboring "landmarks".

II. Related Work

In his survey, Shamir [11] divided 3D segmentation research into 5 groups. Chen et al [1] described comparison of 7 different segmentation algorithms using 3D segmentation benchmark. Our purpose in this section is not to do comprehensive survey, but to focus on some of prominent approaches, from which we can extract two of methods; the one for a known superior method and the other for a baseline method.

- Region growing based [7]
- Topology based [14]
- Clustering based [12]
- Graph partitioning based [4]
- Spectral analysis based [9]

Katz et al [7] proposed a segmentation method by first transforming an input model based on the theory of multi-dimensional scaling (MDS) [3] into a pose-invariant representation. Second, this representation facilitates the robust extraction of prominent feature points. Finally, they extracted the core component of the mesh by using a spherical mirroring operation for segmentation.

Tierny et al [14] took a different approach to 3D shape segmentation. They first extracted *enhanced topological skeletons*, and these skeletons were also used to determine the core shape geometry. Then a semantic-oriented hierarchical segmentation process was carried out. Their method has a limitation that it is difficult to distinguish features when features form a compact connected component like a closed fist, where we cannot tell the difference between two skeletons of similar shapes.

Shlafman et al [12] proposed a surface decomposition by using an algorithm similar to k-means clustering algorithm, where given a polyhedral surface S with n vertices, they compute a decomposition of S into k disjoint patches $S_1, S_2, ..., S_k$. To determine the segment border, they employed the dihedral angles between adjacent faces, so that if two adjacent faces are close to being coplanar to each other, they are more likely to belong to the same patch. The distance between two adjacent faces was thus defined as $Distance(F_1, F_2) = (1 - \delta)(1 - \cos^2(\alpha)) + \delta Phys\_Dist(F_1, F_2)$, where δ is a weight parameter and

$Phys\_Dist$ denotes the Euclidean distance between the centers of gravity of two adjacent faces. On the other hand, the distance between non-adjacent faces was defined as $Distance(F_1, F_2) = \min_{F_3 \neq F_1, F_2}(Distance(F_1, F_3) + Distance(F_3, F_2))$. This amounts to a close approximation to a geodesic distance.

Golovinsky et al [4] proposed a hierarchical decomposition procedure that uses a set of randomized minimum cuts to guide placement of segmentation boundaries. They classify this method into "Randomized cuts" to discriminate it from "Normalized cuts", which they propose in the same paper. With "Randomized cuts", they first decimate the mesh (to 2,000 triangles), and they then proceed top-down hierarchically, starting with all faces in a single segment and iteratively making binary splits. For each split, they compute a set of randomized cuts for each segment, and then they identify for each segment which cut is most consistent with others in the randomized set. Amongst this set of candidate cuts, they choose the one that results in the minimal normalized cut cost (called "MinCut" algorithm). The algorithm terminates when a user-specified number of segments has been reached. There is a limitation of this approach in that MinCut algorithm produces unstable results for symmetric objects.

Liu et al [9] proposed a mesh segmentation algorithm via recursive bisection where at each step, a sub-mesh embedded in 3D was first spectrally projected into the plane and then a contour was extracted from the planar embedding. They rely on two operators to compute the projection: the graph Laplacian and a geometric operator designed to emphasize concavity. They achieved invariance to shape bending through multi-dimensional scaling (MDS[3]) based on the notion of inner distance.

III. Protrusion Based Segmentation

Our proposed hypothesis is stated as follows: an arbitrary complex 3D shape model should consist of a rounded core part and one or more "protruded" parts, with each "protruded" part may be further made of a rounded core part and one of more finer "protruded" parts, and so on in a nested fashion. A simple example is illustrated in Fig. 1, where a teddy bear is regarded as semantically being made of a rounded body, illustrated in solid curves, with five "protruded" parts (four legs and one head), shown in dotted curves, and the head is further divided into a rounded part and two "protruded" ears.

In order to translate our hypothesis into practical algorithms, we consider the following two steps. The first step is to detect and identify the shape of "protrusion", and the second step is to determine the boundary between the rounded part and one or more protruded parts. Since we assume that any complex 3D shape has a nested protruded parts, it is natural to apply the above steps iteratively until no evidently protruded parts remain. The net segmentation algorithm is based on this assumption except that we have to conform to the number "k" for stopping criterion of the iteration, where "k" is the desirable number of segments a priori given for each 3D shape.

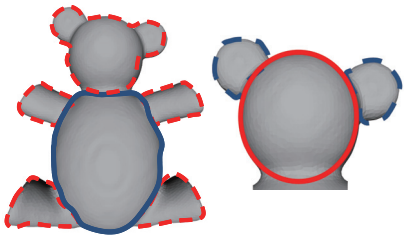

Fig. 1. Sample model having a nested protrusion

Note that each segment constitutes a set of polygonal meshes, and so does each boundary between segments.

A. Overall Segmentation Algorithm

The overall flow of our protrusion based segmentation algorithm is shown in Fig. 2. It consists of four major steps. In the first step, we compute the distances along geodesic path of the model. They are needed in steps 2 and 3. Step 2 selects "tips" of protruded parts (k-1) times, where tips are called "landmarks". During Step 2, we determine the tip of protrusion based on some evaluation criterion, and eliminate implausible tips from consideration. After a tip of protrusion is selected, we compute the distance between the tip and a boundary between segments in Step 3. The boundary determination itself will be elaborated later. In Step 4, we mark the same segment ID to all the meshes falling into the distance between the tip of protrusion and the boundary. We also put a different segment ID to the other meshes. During steps 2 and 3, we may have "landmarks" that do not meet a condition, which have to be removed. Due to this consideration, it may occur after Step 4 that we would not satisfy the number k of segments to be produced. In this case, we pick up the segment that has the largest area and repeat from Step 2 until k number of segments are produced.

B. Distance Computation

In order for our protrusion based segmentation algorithm to work out, it is essential to define the distance traversed along a real surface, made of polygonal meshes, instead of direct Euclidean distance as shown in Fig. 3. The most faithful definition of this distance can be defined by geodesic distance along the polygonal meshes connecting two end points as shown in Fig. 3.

To specify points on the surface, we will employ a centroid of each convex polygon, which will be regarded as a node, and a path between two nodes of neighboring polygons will be regarded as a link, making up a connected graph as a whole.

The actual distances we consider are as follows:

1) Diffusion Distances: Diffusion distance, hereafter we call DD for short, is the distance measured with Diffusion maps [2], [8]. Diffusion maps can be regarded as a non-linear dimensional reduction method, based on unsupervised manifold learning [6]. Like other manifold learning, diffusion maps employ graph theoretic approaches. Specifically, given a weight matrix W of a graph G, and a diagonal matrix D,

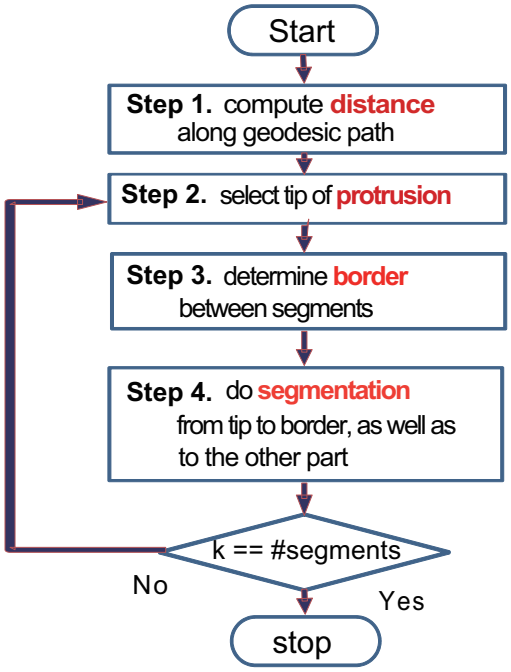

Fig. 2. Flow of protrusion based segmentation algorithm

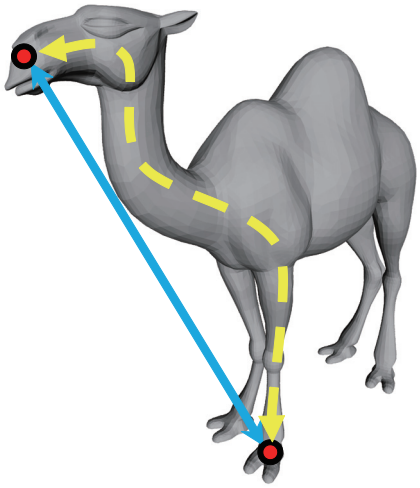

Fig. 3. Distance along a mesh surface and Euclidean distance

defined as $d_{i,i} = \sum_j w_{i,j}$, we first define an operator P as follows:

$$P = D^{-1}W$$

The operator P is regarded as a transition matrix, representing a random walk on the graph, whose transition probability is given by W. The transition probability defines how the data is propagated, which in turn defines the diffusion distance. Assume that P^t represents the probability after t steps of random walks, the diffusion distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ is expressed by the following:

$$D_t^2 = \sum_z (p(t, \mathbf{z}|\mathbf{x}_i) - p(t, \mathbf{z}|\mathbf{x}_j))^2 w(\mathbf{z}),$$

where $p(t, \mathbf{z}|\mathbf{x})$ denotes the probability to reach $\mathbf{z}$ after t steps. This distance can be rewritten by applying dimensional reduction after eigen-decomposition of P as below:

$$D_t^2(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k \geq 1} \lambda_k^{2t} (\psi_k^{(i)} - \psi_k^{(j)})^2,$$

where λ_k is the eigenvalue of the transition matrix P with $1 = \lambda_0 > \lambda_1 \geq \lambda_2 \geq ...$, and ψ_k^i is the i-th element of k-th eigenvector, corresponding to k-th eigenvalue λ_k.

2) Geodesic Distances: Given a connected graph, it is possible to compute a geodesic distance between two nodes with Dijkstra's shortest path algorithm [13]. They showed that the running time of computing geodesics on meshes was $O(N^2 \log N)$, where N denotes the number of nodes in a graph. When N is large, the cost of geodesic computation is prohibitive.

3) Landmark-based Geodesic Distances: To alleviate the expensive cost of computing either the geodesic distance or the diffusion distance, we propose an approximate method by introducing "landmarks", as a small subset of nodes randomly scattered on a selected mesh. The geodesic distance is now replaced by landmark-based geodesic distance, which we call "LGD" (Landmark-based Geodesic Distance) hereafter. The LGD is computed by the following steps:

1) Select N_L landmarks randomly from a model of N meshes
2) Compute distance between landmarks
3) Keep track of normal nodes, and their distances to nearest landmarks
4) Compute LGD, as the sum of distances between normal nodes, where each distance is defined by the distance between a normal node and the nearest landmark plus the distance between normal nodes

Note that "normal" nodes refer to nodes not belonging to landmarks. Since Dijkstra's shortest path algorithm is repeated only N_L times, the computational cost of LGD is therefore $O(N_L(N+E)\log N)$, where E and N_L denote the number of edges in the graph, and the number of landmarks, respectively. Because N_L is much less than N (i.e. $N_L \ll N$), the distance computation is greatly reduced.

C. Selection of Protrusion

Our hypothesis is that an arbitrary complex 3D shape consists of a central rounded part and one or more protruded parts in a nested way. To produce k number of segments, it is straightforward to generate make (k-1) protruded parts, and one remaining central part. The initial choice of protrusion is made by finding the mesh (or equivalently the node in a graph) that satisfies the following equation:

$$\underset{j}{\text{argmax}} \sum_{j=1}^{N} dist(i, j) \quad (1)$$

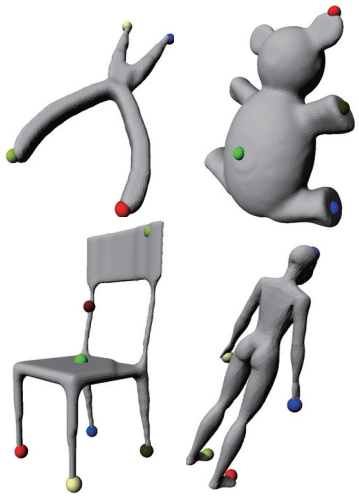

Fig. 4. Examples of tips of protrusion represented by spheres

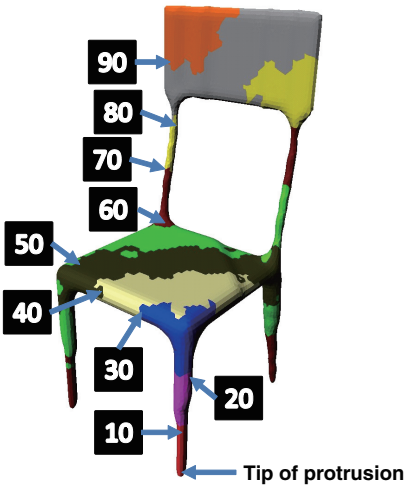

Fig. 5. Distance slots from the tip mesh of protrusion to the other meshes in a distance histogram

The choice after the first mesh is made by finding the mesh that satisfies the following equation:

$$\underset{i}{\text{argmax}}\{\min(dist(p,i))\}, \quad \forall p \in P \tag{2}$$

where P denotes a set of protrusions, already found so far. The idea behind the selection of protrusion with the above equations is based on our observation that a protruded part tends to be evenly distributed apart from the central part. This is true for 3D shapes with well-balanced protrusions such as four-leg desks and animals.

Fig. 4 illustrates how we select protrusions with the above-mentioned algorithm, where the selected mesh is denoted by spheres. For example, the plier shape on the left top in Fig. 4 has four spheres chosen at four tips of protrusion. Similarly the human shape on the right bottom has five spheres at five tips (two at fingertips, two at tiptoes, and one at top of head) of protrusion. On the other hand, the chair on the left bottom, and the teddy bear on the right top, may not produce good tips of protrusion. However, during border computation step (i.e. Step 3), since erratic tips will not produce good borders, they will be disregarded.

D. Border Computation

Generating "good" and smooth-curved borders between segments is an important but seemingly formidable task. The quality and the beauty of borders depend both upon how the shape is made of polygonal meshes of different sizes and upon what level of detail is specified in terms of mesh resolution. Since we do not tell in advance anything about these meshing details, we propose to determine the border meshes using a method independent of the size of meshes and the meshing resolution. Specifically, we introduce two statistics; one is a distance histogram that is constructed by taking the statistics of the LGD distance from the tip mesh of protrusion to the other meshes. The other is a statistics of diameters of sectional area from the tip mesh of protrusion to the other meshes. We will describe these in turn.

1) Distance Histogram: To identify borders, it is natural to think that we should extract one or more features that suggest the existence of border with high probability. The first feature we consider is a distance histogram. This histogram is constructed in such a way that starting from the tip mesh of protrusion, we gradually proceed to distant meshes along LGD path. We subdivide the farthest distance from the tip into ten groups for coloring with the same distance range. Each group consists of ten intervals, which in total amounts to hundred intervals. We will call "slot number" for referring to the interval from 1 to 100. The vertical axis is the frequency of polygonal meshes belonging to each interval. Fig. 6 illustrates this distance histogram for a chair shown in Fig. 5. Our observation from this histogram is that a segment border is likely to be placed at an inflection point of the outer shape of the histogram, where the flatter contour becomes a sudden increase or decrease. For example, from slot numbers 21 to 30, there seems to be an inflection point. Note that slot numbers are the same with the number attached to Fig. 5.

2) Sectional Area Diameter Histogram: The second feature we consider in order to identify the border of segments is a distribution of diameters of the sectional area from the tip of protrusion, assuming that any dissecting plane to the 3D shape has intersections with the polygonal meshes at least two or more points.

Fig. 7 shows an example of this histogram, corresponding to the same data as in Fig. 6. A similar inflection point is observed between slot number 21 and 31.

3) Border Approximation Algorithm: Two previous histograms make it possible to estimate candidates of the borders between segments. Specifically, we first generate candidate slot number intervals that include inflection points from distance

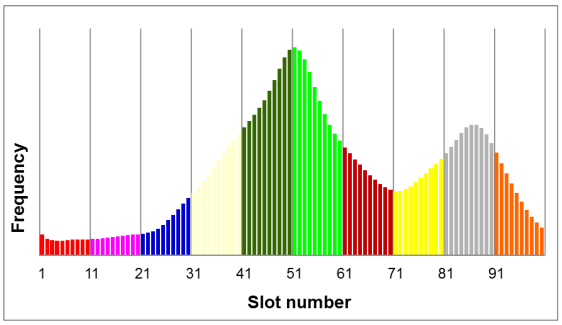

Fig. 6. Histogram of distances from the tip of protrusion

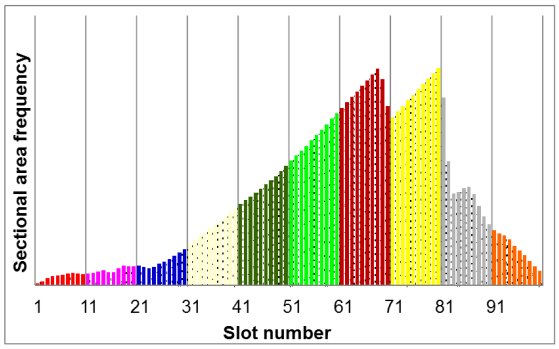

Fig. 7. Histogram of diameters of sectional area from the tip of protrusion

histograms. We then attempt to find out the most prominent inflection point and the associated slot number from sectional area diameter histogram. If the slot number is included in the candidate intervals from the distance histogram, we output the slot number as a border between segments. If the slot number is not included, we proceed to the second most prominent inflection point and the associated slot number from sectional area diameter histogram.

For example, from distance histogram 6, slot number intervals such as (23,68) and (71,80) are computed as candidate borders, while from sectional area diameter histogram 7, slot number 23 is computed as the most prominent inflection interval. Since slot number 23 is included in the interval (23,68), we will generate slot number 23 as a border candidate.

IV. Experiments and Evaluations

In this section, we will describe two separate experiments. The first experiment concerns the measure of distances, while the second experiment concerns the comparative experiments of segmentation, where our method (simply called "Protrusion") incorporates the LGD (Landmark-based Geodesic Distance), necessary to estimate the border of segments, after showing that the LGD results in the good balance between the fast computation and the accurate segmentation.

A. Distance experiments

The experiments have been conducted under a PC with Intel Core i7 920 CPU, 12GB memory, and Debian GNU/Linux operating system with g++ compiler v4.3.2. Eigenvalue computation needed in Diffusion Distance (DD) has been done by using CPP Lapack++ [15]. For computational complexity between geodesic distance and LGD, which has been discussed in *Distance Computation* section. Here, by taking two 3D models, an octopus consisting of 19,510 meshes and a camel consisting of 2,682 meshes, chosen from 3D segmentation benchmark we will describe in the subsequent section, we compare the time needed to estimate the border by repeatedly using the distance measure. Tables I and II summarize the result. Specifically, Table I describes the meaning of the names of distance. For example, GD denotes Geodesic Distance, DD denotes Diffusion Distance, while LGD denote our proposed Landmark based Geodesic Distance. Table II shows the time to compute the border of segments by repeatedly using the distance. Together with other experiments using different models, we have determined to adopt LGD(5), the Landmark based geodesic distance with 5% of the original number of meshes. This selection makes it possible to take reasonable trade-offs between fast computation and accurate segmentation due to the approximation of geodesic distance. The trade-offs will be elaborated in later section in more detail.

TABLE I
Distance names and their meanings

distance name	meaning
GD	Geodesic Distance
DD(k)	number of eigenvectors is the same with k
DD(5)	number of eigenvectors = 5
DD(10)	number of eigenvectors = 10
LGD(1)	1 % of number of meshes (N)
LGD(5)	5 % of number of meshes (N)
LGD(10)	10 % of number of meshes (N)

TABLE II
Distance comparison

distance name	Time (octopus)[sec]	Time(camel)[sec]
GD	12.9	850.5
DD(k)	17.4	105.8
DD(5)	17.7	105.4
DD(10)	18.3	106.3
LGD(1)	0.44	32.9
LGD(5)	0.96	68.2
LGD(10)	1.67	135.4

B. Segmentation experiments

For comparative experiments on segmentation, we have used a benchmark data set called 3D Mesh Segmentation Benchmark [1], MSB for short. MSB consists of twenty classes, each of which has twenty different 3D models, resulting in four hundred 3D models in total. Since plural manual segmentations (i.e. tentative answers) to each of the 3D models in MSB are provided with MSB, we could regard the manual segmentations as sample distributions over how humans decompose each 3D model into functional parts. Thus, they could be served as a kind of "ground truth".

C. Evaluation measures for segmentation

For comparison, we employed "Randomized Cuts" [4] (hereafter we call "RandCuts" for short) as a known superior method, and K-means method [7] as a baseline method. Evaluation measures include cut discrepancy [5], Hamming distance [5], and RandIndex [10].

Cut discrepancy is an evaluation measure of segmentation, indicating how badly the border between segments is cut. If this value is small, it means that the border is cut as close to a direct line. Cut discrepancy is sensitive to the granularity of segmentation. Given two segmentations, Hamming distance evaluation measure attempts to establish a correspondence between the two, computing the intersection between the two regions. Hamming distance measure is the normalized intersection. The smaller the degree of mismatch, the better the segmentation becomes. RandIndex is another region-based evaluation measure for segmentation, proposed by Rand [10]. It represents the ratio of duplication of the same location, and we use the value by subtracting one minus this ratio. In effect, the resulting segmentation is better when the value is smaller.

D. Results

Fig. 8 shows some of the results with protrusion based segmentation we propose. Tables III, IV, and V demonstrate the results based on the three evaluation criteria for 12 representative classes out of all 20 classes with RandCuts, K-means, and our proposed Protrusion methods. The data classes of these tables are sorted by the values of RandCuts method in an ascending order, where RandCuts is chosen as the best known previous method for 3D segmentation. Although the three evaluation criteria are different aspects of segmentation errors, there is a common characteristic that the smallest value in the same data class represents the best method among the three. We shade the cell of the best method in yellow for clarity. The names of data classes with an *italic* text represent the fact that our proposed (Protrusion) method exhibits the best among the three.

For data classes *Table* and *Plier*, our proposed method outperformed the other two in all the three evaluation criteria, while for classes including Teddy, our method was the worst among the three methods. We conjecture that this is in part because "protruded" parts of Teddy do not have outstanding sharp and well distinguished tips, and in part because distance and diameter histograms do not have well discriminated inflection points for border computation.

To be more concrete, Table III shows the comparative results using cut discrepancy evaluation measure. In addition to *Table* and *Plier*, data classes *Octopus*, *Cup*, and *Mech* exhibit the best results with our proposed method. Table IV shows the comparative results using Hamming distance evaluation measure. In addition to *Table* and *Plier*, data classes *Fish*, *Cup*, and *Mech* exhibit the best results with our proposed method. Table V shows the comparative results using RandIndex evaluation measure. In addition to *Table* and *Plier*, data class *Octopus* turns out to be the best result with our proposed method.

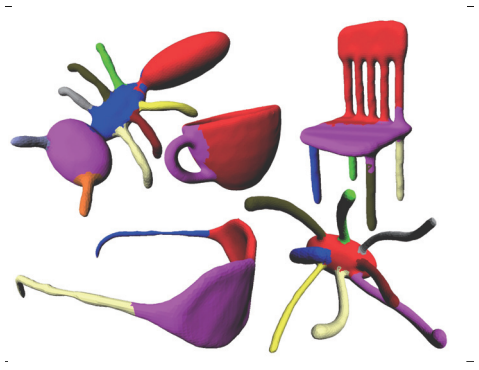

Fig. 8. Sample segmentation results with protrusion based segmentation

TABLE III
COMPRISON WITH CUT DISCREPANCY

Data	RandCuts	K-Means	Protrusion
Teddy	0.048	2.639	4.818
FourLeg	0.120	0.246	0.664
Ant	0.137	2.562	0.548
Glasses	0.155	1.430	0.190
Chair	0.750	1.208	1.344
Octopus	0.890	2.522	0.889
Plier	0.999	4.423	0.242
Fish	1.214	1.126	3.080
Cup	1.418	1.860	1.079
Mech	1.875	2.894	1.000
Table	2.181	2.453	0.717

E. Trade-offs

As we have mentioned in *Distance experiments* section, we have adopted LGD(5), Landmark based Geodesic Distance with 5% of the number of meshes for landmarks. We found from Tables I and II, that LGD(5) proved to be ten to fifteen times faster in computation than GD (Geodesic Distance) which was based on the Dijkstra's algorithm with $O(N^2 \log N)$ time complexity, where N denotes the number of meshes. For simplicity, we will exclude DD (Diffusion Distance) in this section.

As indicated in the preceding section, we thought there were trade-offs between the fast compuation and the accurate segmentation. We have attempted to verify the presumed trade-offs between speed and accuracy.

Among four hundred models with twenty classes, we selected typical two data with relatively large number of meshes; one from an *Octopus* class (an octopus with 30,982 meshes (model No.123)) and another from a *Table* class (a table with 26,746 meshes (model No.148)) to verify the trade-offs. Table VI shows the results of evaluation measures (Hamming Distance (HD), RandIndex (RI), and Cut Discrepancy (CD)) of segmentation with respect to several methods of ours having different computational costs shown in Table II, including

TABLE IV
COMPRISON WITH HAMMING DISTANCE

Data	RandCuts	K-Means	Protrusion
Glasses	0.034	0.852	0.577
Teddy	0.058	5.099	4.024
Ant	0.160	2.877	1.173
FourLeg	0.252	0.784	0.275
Cup	0.436	2.173	0.123
Octopus	0.538	2.286	0.734
Mech	0.654	2.319	0.612
Plier	0.695	2.869	0.276
Fish	0.962	2.716	0.673
Chair	1.234	2.561	1.266
Table	2.532	3.243	0.310

TABLE V
COMPRISON WITH RAND INDEX

Data	RandCuts	K-Means	Protrusion
Glasses	0.039	0.764	0.782
Teddy	0.083	2.689	7.712
Ant	0.156	3.418	1.873
FourLeg	0.187	0.313	0.826
Plier	0.545	2.717	0.149
Cup	0.645	2.224	0.922
Fish	0.918	1.678	1.311
Chair	1.127	1.398	1.568
Mech	1.160	2.245	1.866
Octopus	1.812	3.226	1.172
Table	3.010	3.067	0.470

LGD(1), LGD(5), LGD(10), and GD, using an octopus model No.123. RandCuts and K-Means are also included for the sake of comparison, where the values are extracted from the data attached with MSB. It is noted that the computational costs of RandCuts and K-Means are not provided with MSB. The value with an asterisk in Table VI means that it is the smallest (i.e. the best) among the segmentation methods. Table VII shows the similar experimental results, using a table model No. 148.

Fig. 9 visually demonstrates the different segmentations with respect to an octopus model No.123. It is interesting to note that GD, LGD(10), LGD(5), and LGD(1) look all similar, even though the computing time of LGD(1) (84 sec) is approximately 18 times faster than that of GD (1527 sec). Fig. 10 visually demonstrates the different segmentations with respect to a table model No.148. In Fig. 10, GD, LGD(10), and LGD(5) look similar to each other, whereas LGD(1) has apparent bad segment border depicted by the region in blue color. Tables VI and VII indicate that coarser models in terms

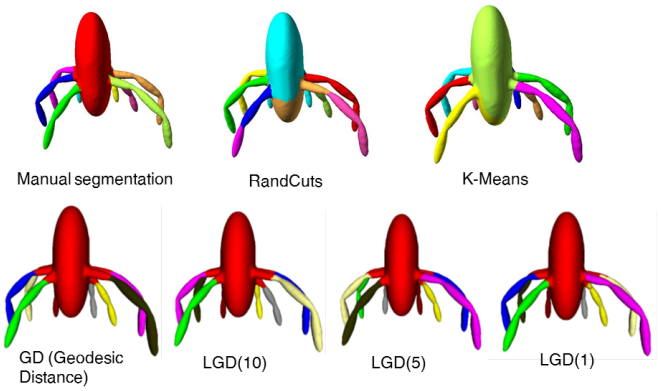

Fig. 9. Comparison with different segmentations for an onctopus model No.123

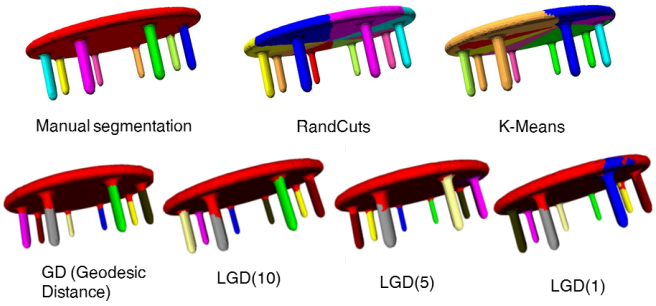

Fig. 10. Comparison with different segmentations for a table model No.148

of the number of landmarks do not necessarily attribute to less accurate segmentation with respect to HD, RI, and CD. Nevertheless, based on this observation as well as on the quality of segmentation seen from Fig. 9 and Fig. 10, the adoption of LGD(5) seems reasonable in terms of the trade-offs between speed and accuracy.

TABLE VI
COMPARISON OF EVALUATION MEASURES OF SEGMENTATION WITH AN OCTOPUS (NO.123) MADE OF 9 SEGMENTS

method	HD	RI	CD
RandCuts	0.203	0.218	0.116*
K-Means	0.190	0.175	0.233
GD	0.086	0.108	0.285
LGD(1)	0.067*	0.081*	0.191
LGD(5)	0.076	0.093	0.255
LGD(10)	0.085	0.106	0.285

V. CONCLUSION

We proposed a protrusion based segmentation method for a complex 3D shape model. To make it possible for our algorithm to work, we described a method to identify tips of protrusion by using geodesic distance between meshes. Specifically, we first computed all combinations between arbitrary

TABLE VII
COMPARISON OF EVALUATION MEASURES OF SEGMENTATION WITH A TABLE (NO.148) MADE OF 9 SEGMENTS

method	HD	RI	CD
RandCuts	0.341	0.526	0.326
K-Means	0.358	0.551	0.537
GD	0.029	0.064	0.122
LGD(1)	0.036	0.070	0.112
LGD(5)	0.030	0.057	0.093
LGD(10)	0.024*	0.051*	0.090*

two meshes, and attempted to find out a mesh with largest sum of distanced from the mesh, which will be the first selected ("Landmark") mesh. From the second candidate, we repeat a similar process trying to find a mesh that is farthest from all the landmarks found so far, until the number of landmarks become $k-1$, where k is a desired number of segments.

Upon completion of candidate landmark computation, we proposed another method for determining the border between segments. To do so, we proposed two clues: the one is a distance histogram from the candidate landmark, and the other is a diameter histogram from the candidate landmark. If these two histograms share a common slot of having the inflection point, showing the rapid increase of the number of meshes within the same slot, we assume that the segment border should be placed around the meshes.

Comparative experiments based on 3D Segmentation Benchmark data set having 20 classes, each of which contain 20 shape models, have been carried out. For classes with protrusions well observed, we have better results than k-means type segmentation as well as the randomized cuts algorithm which exhibited the best average performance in the previously developed methods. For classes with protrusion not well identified, however, we still leave further improvements to be desired.

ACKNOWLEDGMENT

This research was conducted by the Strategic Information and Communication R&D Promotion Programme (SCOPE 112306001) of Ministry of Internal Affairs and Communications (MIC) Japan.

REFERENCES

[1] Xiaobai Chen, Aleksey Golovinsky, and Thomas Funkhouser. A Benchmark for 3D Mesh Segmentation. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3):73:1–73:12, 2009.

[2] Ronald R. Coifman and Stephane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21:5–30, 2006.

[3] Trevor F. Cox and Michael A. A. Cox. *Multidimensional Scaling, Second Edition*. Chapman & HALL/CRC, 2001.

[4] Aleksey Golovinsky and Thomas Funkhouser. Randomized Cuts for 3D Mesh Analysis. *ACM Transactions on Graphics (Proc. SIGGRAPH ASIA)*, 27(5):145:1–145:12, 2008.

[5] Qian Huang and Byron Dom. Quantitative Methods of Evaluating Image Segmentation. *Proceedings of the 1995International Conference on Image Processing (ICIP'95)*, pages 53–56, 1995.

[6] Alan Julian Izenman. *Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning*. Springer, 2008.

[7] Sasi Katz, George Leifman, and Ayellet Tal. Mesh segmentation using feature point and core extraction. *The Visual Computer (Pacific Graphics)*, 21(8-10):649–658, 2005.

[8] Stephane S. Laffon. Diffusion Maps and Geometric Harmonics. *Ph.D. Dissertation, Yale University*, 2004.

[9] Rong Liu and Hao Zhang. Mesh Segmentation via Spectral Embedding and Contour Analysis. *EUROGRAPHICS 2007*, 26(3):10, 2007.

[10] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, pages 846–850, 1971.

[11] Ariel Shamir. A survey on Mesh Segmentation Techniques. *Computer Graphics Forum*, 27(6):1539–1553, 2008.

[12] Shymon Shlafman, Ayellet Tal, and Sagi Katz. Metamorphosis of Polyhedral Surfaces using Decomposition. *EUROGRAPHICS 2002*, 21(3):10, 2002.

[13] Vitaly Surazhsky, Tatiana Surazhsky, Danil Kirsanov, Steven J. Gotler, and Hugues Hoppe. Fast Exact and Approximate Geodesics on Meshes. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 24(3):553–560, 2005.

[14] Julien Tierny, Jean-Philippe Vandeborre, and Mohamed Daoudi. Topology driven 3D mesh hierarchical segmentation. *IEEE International Conference on Shape Modeling and Applications (SMI'07)*, page 6, 2007.

[15] M. Ueshima. CPPLapack Tutorial. *available at http://cpplapack.sourceforge.net/*, 2004.