Hierarchical Mesh Segmentation Based on Global Sharp Vertices

Kwan-Hee Yoo, Chan Park, Young-Jin Park
Dept. of Computer Education and IIE
Chungbuk National University, chungbuk, Republic of Korea

Jong-Sung Ha*
Dept. of Game and Contents
Woosuk University, Chonbuk, Republic of Korea

ABSTRACT

In this paper, we propose a hierarchical method for segmenting a given 3D mesh, which hierarchically clusters sharp vertices of the mesh using the metric of geodesic distance among them. Sharp vertices are extracted from the mesh by analyzing convexity that reflects global geometry. As well as speeding up the computing time, the sharp vertices of this kind avoid the problem of local optima that may occur when feature points are extracted by analyzing the convexity that reflects local geometry. For obtaining more effective results, the sharp vertices are categorized according to the priority from the viewpoint of cognitive science, and the reasonable number of clusters is automatically determined by analyzing the geometric features of the mesh.

Keywords: *Mesh Segmentation, Hierarchical Clustering, Sharp Vertex, Geodesic Distance.*

1. INTRODUCTION

Mesh segmentation is an optimization problem that decomposes a given mesh into a disjoint set of sub-meshes while satisfying some criteria to the maximum. Since three dimensional meshes are represented with vertices and edges similarly to graph data structures, the mesh segmentation can be regarded as a graph partitioning. The optimization of mesh segmentation under some criteria is an NP-class problem [1] even if the problems are much limited. Furthermore, the mesh segmentation is also input-sensitive; the efficacy of a segmentation method varies from artifacts to natural objects. By those reasons, mesh segmentation methods take heuristic algorithms and need to be more optimized when they are combined to a specific application such as mesh modeling, metamorphosis, compression, simplification, shape matching, collision detection, texture mapping, and skeleton extraction.

Since a lot of heuristic mesh segmentation methods [1-15] can be categorized by many criteria or viewpoints, it is very difficult to formally summarize and quantitatively analyze all of the previous researches. The hierarchical clustering method of this paper is a kind of bottom-up approach [1-3] that merge the faces of a mesh increasingly, differently to the top-down approaches [4-6] that divide the mesh continuously. Our method treats only the sharp vertices extracted from the given mesh, which is based on the research result [16], [17] that the negative minimum of principal curvatures and the depth of concavity directly influence on identifying the boundary of some regions. This notion of cognitive science has been adopted also in Zhou and Huang [12], Katz *et al.* [14], and Lien and Amato [15].

Zhou and Huang [12] designates a root from extreme vertices arbitrarily, constructs the shortest path trees of geodesic distance from the root to all extreme vertices, computes local extreme points and saddle points, and then divide the given mesh with backward flooding algorithm. Katz *et al.* [14] transforms a mesh into another mesh by the multi-dimensional scaling of pose-invariant representation method, extracts feature points based on geodesic distance, and finds the core components with the spherical mirroring providing the duality between convexity and concavity. Lien and Amato [15] defines a measure for the quantity of concavity, and recursively decompose the mesh by finding the greatest concavity. In our method, rather than convexity reflecting local geometry [12], [14] and concavity [15], sharp vertices are extracted by analyzing convexity reflecting global geometry, which will be explained in Section 2. In Section 3, Mesh segmentation is performed by hierarchical clustering of the sharp vertices after assigning the priority numbers of cognitive importance to them. Especially, we automatically determine an appropriate value for the number of clusters by analyzing the geometric features of meshes in Section 4. We present our experiments and segmented results are visualized with the previous results in Section 5.

* *Corresponding author. E-mail : jshajsha@yahoo.co.kr*
Manuscript received Aug.31, 2009 ; accepted Nov. 05, 2009

2. EXTRACTING SHARP VERTICES

2.1 Local sharp vertices of approximate curvatures

Since the surface curvature is a very important factor for the recognition of 3D shapes, usually sharp vertices are extracted using the vertex curvatures of meshes The curvatures of mesh vertices cannot be computed mathematically differently to the curved surfaces, and the approximate curvatures of mesh vertices can be computed by two categories of methods; continuous approaches [18], [19] fitting the vertices into a parametric curved surface, and discrete approaches [20], [21] computing normal curvatures in the directions of edges incident to the vertices. The sharp vertices extracted using these curvatures are geometrically local, that is, there is a critical problem of local optima in mesh segmentation.

2.2 Global sharp vertices of approximate curvatures

The previous approaches [12], [14] for extracting sharp edges and vertices consider only the approximate curvatures. However, the curvature may reflect only the local geometry in a dense mesh, and the simplification of the dense mesh may lose the original shape. This paper extracts the sharp vertices using another geometric operation reflecting the global geometry.

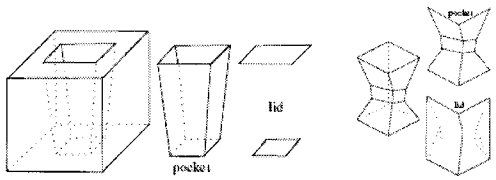


Fig. 1. Pockets and lids of polyhedrons

The typical vertices of global sharp geometry are the extreme points that can be obtained by the geometric operation of convex hull. From the similar observation, we develop a method for extracting and categorizing more global sharp vertices by observing and analyzing geometric features. As illustrated in figure 1, the pockets and lids of polyhedrons can be computed by the geometric operations of convex hull and regularized difference. It is reasonable to regard the vertices on the boundaries of the pockets and lids as the most important vertices representing the features.

Since 3D models are not simple in general, we decompose a mesh surface into sub-meshes before computing pockets and lids with geometric operations. A convex part is usually contained in a geometric feature, and it can be the smallest unit to be considered as a meaningful sub-meshes. A set of faces in a mesh is called the convex patch if they are connected and all of them exit on their convex hull. It is an NP-complete problem [1] to compute the minimum number of disjoint convex patches for a given mesh.

In this paper, we define and use another set of faces for representing a convex part, called the approximate convex patch (ACP), in which faces are connected via convex edges. An ACP can include some concave edges because it is sufficient for a pair of adjacent face if there is at least a convex edge between them. ACPs are always determinant, and can be computed easily with a graph searching technique of breadth-first-search or depth-first-search. A tolerance angle can be permitted in checking the convexity of an edge for allowing that a mesh is usually an approximation of a curved surface.

After decomposing a mesh into sub-meshes, we compute the pocket and lid of each sub-mesh. And then, all vertices are marked with global geometric properties as shown in Tab. 1, which are Boolean tags representing whether they are extreme points or they are contained in pockets, lids, hulls, and boundaries. The properties of Tab. 1 may be exclusive to each other or not, and a vertex can have multiple properties simultaneously. For example, if a vertex is EXTREME, it is also one of IN_HULL_FACE, IN_LID_FACE, and IN_BOUNDARY_EDGE. Reversely, if a vertex is IN_HULL_FACE or IN_LID_FACE, it is also EXTREME, but if the vertex is IN_BOUNDARY_EDGE, it is not necessarily EXTREME. It is possible that a vertex is IN_POCKET_FACE and also one of IN_HULL_FACE, IN_LID_FACE, and IN_BOUNDARY_ EDGE.

Table 1. Vertex marks

Marks for global properties	Descriptions of the properties
EXTREME	An extreme point of the convex hulls of ACPs
IN_POCKET_FACE	Contained in a pocket of ACPs.
IN_HULL_FACE	Contained in a hull of ACPs and also in an ACP
IN_LID_FACE	Contained in a lid of ACPs (in a hull but not in an ACP)
IN_BOUNDARY_EDGE	Contained in a boundary of ACPs

Instead of a complicated algorithm for marking the properties of table 1, we briefly describe a conceptual algorithm as follows.

Algorithm VERTEX_MARKING
input: M
output: M' with marked vertices

1. Partition M into approximate convex patches $\{ ACP_i \}$.
2. **for each** ACP_i **do**
3. Construct the convex hull CH_i of ACP_i.
4. Mark the extreme vertices of CH_i with EXTREME.
5. Construct the pocket P_i and the lid L_i with the deficiency of CH_i and ACP_i.
6. Mark the vertices in the boundary of ACP_i with IN_BOUNDARY_EDGE.
7. **for each** face $F_j \in P_i$ **do**
8. **if** ($F_j \in CH_i$) *tag* ← IN_HULL_FACE
9. **else** *tag* ← IN_POCKET_FACE

10. **for each** vertex $V_k \in F_j$ **do**
11. **if** (V_k is not marked with IN_BOUNDARY_EDGE)
12. Mark V_k with tag .
13. **end for each**
14. **end for each**
15. **for each** face L_i **do**
16. **if** ($F_j \in CH_i$) $tag \leftarrow$ IN_HULL_FACE
17. **else** $tag \leftarrow$ IN_LID_FACE
18. **for each** vertex $V_k \in F_j$ **do**
19. **if** (V_k is not marked with IN_BOUNDARY_EDGE)
20. Mark V_k with tag .
21. **end for each**
22. **end for each**
23. **end for each**

end algorithm

3. MESH SEGMENTATION WITH HIERARCHICAL CLUSTERING

3.1 Brief of hierarchical clustering

In our mesh segmentation method, we apply the hierarchical clustering technique of Johnson [22] into the extracted sharp vertices of the mesh, and then the other vertices are processed for being classified into clusters. This separated process of the two groups of vertices can obtain the effectiveness and efficiency since only a small set of vertices reflecting geometric characteristics is considered in complicated computations such as geodesic distance and clustering. The hierarchical clustering is processed as follows.

Step 1: Allocate a cluster to each sharp vertex, and compute the distance for each pair of clusters.

Step 2: Merge the two clusters between which the distance is the minimum among the computed distances.

Step 3: Update the distance for each pair of clusters.

Step 4: Repeat Step 2 to 3 until the distances are less than a reasonably determined threshold.

There are two major issues in the above: a definition of the distance metric between a pair of clusters, and a method for the hierarchical clustering of sharp vertices and the processing of other vertices, which will be described in Section 3.2, 3.3, and 4 respectively.

3.2 Geodesic distance between a pair of clusters

First, we define the geodesic distance between a pair of points before defining it between a pair of clusters. In general, the Euclidian distance is used as the distance metric between a pair of points in 3D space. However, in this paper, we use another metric composed of the geodesic distance on the surface and the angle between the normal vectors of vertices, since the mesh segmentation takes places on the surface of the mesh.

If we assign the Euclidian distance $ed(v_i, v_j)$ between a pair of adjacent vertices v_i and v_j to the edge incident to the two vertices in a mesh, the mesh is equivalent to a weighted graph. Then, the geodesic distance $gd(v_p, v_q)$ between a pair of vertices v_p and v_q is the one with the shortest length among the paths

$path_k = (v_{k1} = v_p, v_{k2}, \cdots, v_{km}) = v_q), k = 1, \cdots, n$

between the two vertices as follows.

$$gd(v_p, v_q) = \underset{k=1}{\overset{n}{Minimum}} \sum_{i=1}^{m} ed(v_{ki}, v_{k(i+1)}) \quad (1)$$

Since the geodesic distance $gd(v_p, v_q)$ is greater than zero, we can get the shortest path by applying the Dijkstra algorithm [23] after transforming the mesh into a weighted graph.

A distance metric between a pair of clusters c_i and c_j can be defined with the terms of $gd()$. We can consider two kinds of distance metric between the pair of clusters c_i and c_j : the shortest or the longest one among the geodesic distances between all pair of sharp vertices that are included respectively in a pair of clusters, or the average of the geodesic distances, which are respectively formulated as follows.

$gd(C_i, C_j)$ = Minimum/Maximum/Average of

$$gd(v_p, v_q) \text{ for all } v_p \in C_i, v_q \in C_j \quad (2)$$

In the implementation and experiments of this paper, the average metric is used for the mesh segmentation using the hierarchical clustering.

3.3 Hierarchical clustering of sharp vertices

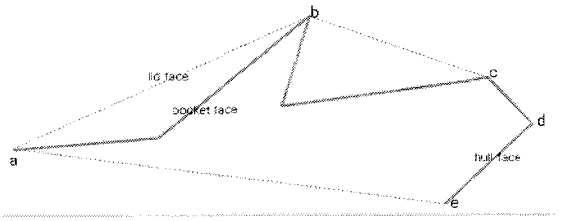


Fig. 2. Classification of Sharp Vertices

As explained in Section 2, sharp vertices for each sub-mesh ACP are computed independently. We explain the priorities of the sharp vertices with a sub-polygon in 2D that are similar to the cross-section of a sub-mesh ACP in 3D as illustrated on Fig. 2. As summarized in table 2, there are three kinds of faces: the hull face that is a part of convex hull and also an original face of the sub-polygon, the pocket face that is an original face of the sub-polygon but not in convex hull, and the lid face that is a new face created for constructing the convex hull. We assign the highest priority 1 to sharp vertices that are extreme and not

in hull faces as like the vertex 'b', since they looks very sharp and plays an important role for features. The next priority 2 is assigned to the vertex 'd' that is extreme and not in lid faces. The priority 2 is used for sharp vertices that are extreme and in both of a hull face and a lid face as like the vertex 'c'. The vertices 'a' and 'e' are the boundary of the sub-polygon. Table 2 shows our categorizing the global sharp vertices from observing such marks of vertices.

Table 2. Priorities of sharp vertices

Priorities of sharp vertices	The marks of a vertex
1	EXTREME and not IN_HULL_FACE
2	EXTREME and not IN_LID_FACE
3	EXTREME and IN_HULL_FACE and IN_LID_FACE

Since its possibility is very high that sharp vertices with the priority 1 and 2 may be near the centroids of segmented mesh features, such sharp vertices are hierarchically clustered by being regarded as the centroids. Sharp vertices with the priority 3 are used for the detailed segmentation as the secondary vertices of the segmented mesh features.

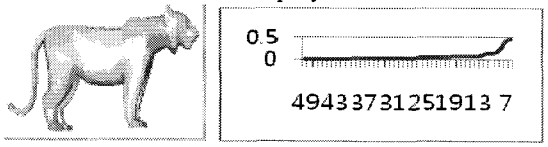
4. TERMINATING CONDITION AND PROCESSING OF NON-SHARP VERTICES

It is very important to determine the terminating condition of iterative hierarchical clustering. The clustering is repeated until the geodesic distances for all pair of clusters are less than a threshold that is determined by automatically computing a reasonable number of final clusters. In order to determine the number of final clusters, we import the similar method to Katz and Tal [6]. The main idea is to investigate the variation of a certain measure presenting the geometric features of clusters by decreasing the number of clusters. Let $G(h)$ be the minimum geodesic distances among the centroids of sharp vertices of clusters, where h is the number of clusters that is initially the number of sharp vertices with the priority 1 and 2.

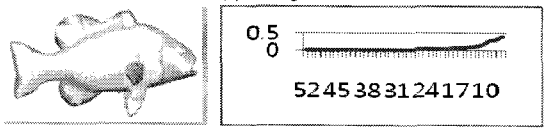
$$G(h) = \text{Minimum of } gd(v_{kj}, v_{ki}),\ i, j = 1, \cdots, h \quad (3)$$

where v_{ki} is the centroid of the sharp vertices of a cluster.

By repeating the clustering, we compute the variation of by decreasing one by one, and find the place when the measure is increased the most rapidly.



(a) a tiger model



(b) a fish model

Fig. 2 Variations of $G(h)$ for mesh models

Figure 2 illustrates examples for investigating the measures $G()$ for two models of animals, which are composed of faces with the number 956 and 1604 respectively. The horizontal and vertical axes represent the number of clusters represents respectively. In figure 2(a), the number of sharp vertices with the priority 1 or 2 is 49, and the $G()$ rapidly increased when is 13 that is just the reasonable number of clusters. Similarly in figure 2(b), the initial number of clusters is 52 and the chosen number of clusters is 15.

After clustering the sharp vertices, the mesh segmentation can be completed by processing other vertices except the sharp vertices. The boundary of each segmented sub-mesh, which is composed of the non-sharp vertices, is dependent on the surface curvature as well as the geodesic distance between the non-sharp vertices and the centroids of clusters of sharp vertices. The non-sharp vertices are classified and merged into the nearest cluster by the following distance metric.

$$d(v_p, v_q) = w_1 \cdot gd(v_p, v_q) + w_2 \cdot \left| C_{v_p} - C_{v_q} \right| \quad (4)$$

where w_i is the weight, and C_{v_p} is the vertex curvature.

As mentioned in Section 2.1, the curvatures of mesh vertices are approximately computed by the two methods of continuous approaches [18], [19] and discrete approaches [20], [21]. The continuous approaches are accurate in dense meshes, but unstable in sparse meshes. Many discrete approaches are stable in both of dense and sparse meshes. In order to compute the vertex curvatures approximately, we uses the method of Cohen-Steiner et al. [24], which employs the theory of normal cycle to compute curvature tensors in a uniform method for the both of curved and planar surfaces. The normal cycle is a theory that generalizes the technique of differential geometry for computing curvatures by considering the offsets of objects, since the technique has been applicable only to the convex objects but inapplicable to the objects with any dimension and condition. This method is able to take both of continuous and discrete approaches for the 3D meshes, efficient and reliable in the computation, and converged by restricting the error bound of approximate curvatures in the case of Delaunay triangulation of surfaces.

5. IMPLEMENTATIONS AND EXPERIMENTS

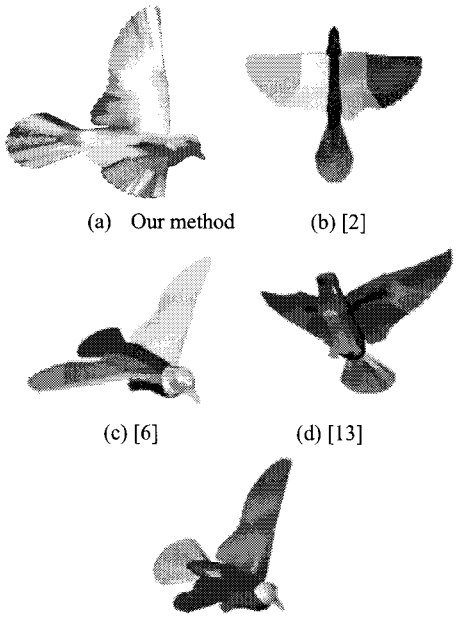
Our hierarchical clustering method for mesh segmentation has been implemented by using CGAL (computational geometry algorithm library) [25] in which the 3D models are represented half-edge data structures. Computer environments are Microsoft Windows and Visual Studio .NET 2003. Mesh data for testing our implementation were obtained almost from the Princeton shape benchmark [26]. The segmented meshes are visualized by programming with the libraries of OpenGL and GLUT.

Table 3 illustrates the original meshes of four animal models, the sharp vertices are extracted by the method of Section 2.2, and the number of clusters is computed by the method of Section 3. The numbers of clusters are from 11 to 18, and the sharp vertices are in the range of about 10~20% rate of all

vertices.

Table 3. Sharp vertices and cluster numbers in our experiments

models	# of vertices	# of sharp vertices				# of clusters
		priority 1	priority 2	priority 3	total	
fish	1604	51	1	55	107	15
bird	567	49	0	10	59	18
tiger	956	49	0	66	115	13
human head	428	19	12	64	95	11

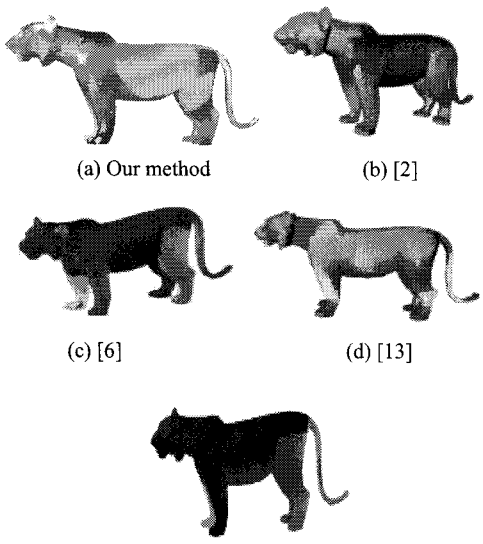


(a) Our method (b) [2]

(c) [6] (d) [13]

(e) [14]

Fig. 3. Mesh segmentation results for a bird model



(a) Our method (b) [2]

(c) [6] (d) [13]

(e) [14]

Fig. 4. Mesh segmentation results for a tiger model

Figure 3~4 compare the segmentation results of our method, Attene *et al.* [2], Katz and Tal [6], Mortara *et al.* [13], and Katz et al. [14], where the segmented sub-meshes are colored differently. In our method, the weight values w_1 and w_2 of Eqn. 4 are set to 0.4 and 0.6 respectively.

6. CONCLUDING REMARKS

We presented a hierarchical clustering method for 3D mesh segmentation. In order to resolve the sensitivity to the initial centroids of clusters and avoid falling into the local optima, first we extract the sharp vertices with priority numbers from the mesh by analyzing the global geometry such as convexity such as curvatures from the viewpoint of cognitive science. Next the hierarchical clustering method is applied to the sharp vertices, based on a metric including the geodesic distance between each pair of clusters. Since it is crucial to determine the segmentation resolution, we also presented a method to choose a reasonable number of clusters automatically. Finally the mesh segmentation is completed by merging other vertices except the sharp vertices into the nearest cluster by the distance metric composed of geodesic distances and curvatures.

The procedure of this paper is quite different to the one of another our approach [27] which use also extract global sharp vertices and automatically determine a reasonable number of clusters. In the k -means clustering [27], before iteration, the cluster number k is initially determined and k sharp vertices are arbitrarily selected for the centroids of clusters. On the while, in the hierarchical clustering of this paper, the number of clusters is not determined before iteration. The reasonable number k can be used only as one terminating condition, but not necessarily. At beginning, the sharp vertices are just the initial clusters, and the number of clusters is iteratively reduced by merging a selected pair of clusters.

Our method was experimented for four mesh models and compared with the previous methods. With current results, it is difficult to assert that the efficiency and the quality of our method is better than the previous methods, but only a new approach based on global geometry is proposed and put to the test. Since the mesh segmentation is input-sensitive as well as analyzing formally and quantitatively is a common concern in this research area, it is needed to develop a better method for analyzing the statistics of geometric properties and automatically determine the weight values.

ACKNOWLEDGEMENT

This work was partially supported by Woosuk University (2009), and by Korea Research Foundation Grant funded by Korean Government (MOEHRD) (The Regional Research Universities Program / Chungbuk BIT Research-Oriented University Consortium).

REFERENCES

[1] B. Chazelle *et al.*, "Strategies for polyhedral surface decomposition: an experimental study," *Computational Geometry: Theory and Applications, Vol.7*, 1997, pp. 327-342.

[2] M. Attene, S. Katz, M. Mortara, G. Patane, M. Spagnuolo, and A. Tal, "Mesh segmentation - a comparative study," *IEEE Int. Conf. on Shape Modeling and Applications*, 2006, pp. 14-25.

[3] D. L. Page, A. F. Koschan, and M. A. Abidi, "Perception based 3D triangle mesh segmentation using fast matching watersheds," *Conference on Computer Vision and Pattern Recognition*, 2003, pp. 27-32.

[4] T. Srinak and C. Kambhamettu, "A novel method for 3D surface mesh segmentation," *Proc. of the 6th IASTED International Conference on Computers, Graphics, and Imaging*, 2003, pp. 212-217.

[5] M. Garland, A. Willmott, and P. S. Heckbert, "Hierarchical face clustering on polygonal surfaces," *Proc. of the 2001 Symposium on Interactive 3D Graphics*, 2001, pp. 49-58.

[6] S. Katz and A. Tal, "Hierarchical mesh decomposition using fuzzy clustering and cuts," *ACM Transactions on Graphics (TOG), Vol.22, No.3*, 2003, pp. 954-961.

[7] T. Kanungo, D. M. Mount, N. S. Netanyahu, A. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient -means clustering algorithm: analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.24, No.7*, 2002, pp. 881-892.

[8] A. P. Mangan and R. T. Whitaker, "Partitioning 3D surface meshes using watershed segmentation," *IEEE Transactions on Visualization and Computer Graphics, Vol.5, No.4*, 1999, pp. 308-321.

[9] T. K. Dey, J. Giesen, and S. Goswami, "Shape segmentation and matching with flow discretization," *Proc. of the Workshop on Algorithms and Data Structures (WADS), Vol.2748*, 2003, pp.25-36.

[10] K. Wu and M. D. Levine, "3D part segmentation using simulated electrical charge distributions," *Proc. of the 1996 IEEE International Conference on Pattern Recognition (ICPR)*, 1996, pp.14-18.

[11] D. Cohen-Steiner, P. Alliez, and M. Desbrun, "Variational shape approximation," *Proc. of the 31st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2004, pp.27-34.

[12] Y. Zhou and Z. Huang, "Decomposing polygon meshes by means of critical points," *Proc. of MMM'04*, 2004, pp.187-195.

[13] M. Mortara, G. Panae, M. Spagnuolo, B. Falcidieno, and J. Rossignac, "Blowing bubbles for the multi-scale analysis and decomposition of triangle meshes," *Algorithmica, Special Issues on Shape Algorithms, Vol.38, No.2*, 2004, pp.227-24.

[14] S. Katz, G. Leifman, and A. Tal, "Mesh segmentation using feature points and core extraction," *Visual Computer Vol.21*, 2005, pp.649-658.

[15] J. M. Lien and N. M. Amato, "Approximate convex decomposition of polyhedra," *Technical Report TR06-002, Dept. of Computer Science, Texas A&M University, 2006.*

[16] D. D. Hoffman and W. A. Richards, "*Parts of recognition,*" *Cognition, Vol.18*, 1984.

[17] D. D. Hoffman and M. Singh, "Salience of visual parts," *Cognition, Vol.63*, 1997.

[18] F. Yamaguchi, Curves and surfaces in Computer Aided Geometric Design, *Springer-Berlag*, 1988.

[19] G. Taubin, "Estimating the tensor of curvatures of a surface from a polyhedral approximation," *Proc. of International Conf. on Computer Vision*, 1995.

[20] A. Rosenfeld and E. Johnston, "*Angle detection in digital curves,*" *IEEE Transactions on Computers, Vol.22*, 1973, pp.875-878.

[21] A.D.C. Smith, *The Folding of the Human Brain: from Shape to Function, University of London, PhD Dissertations*, 1999.

[22] S. C. Johnson (1967): "Hierarchical Clustering Schemes" *Psychometrika*, pp. 2:241-254

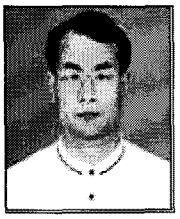
[23] E. W. Dijkstra, "A note on two problems in connection with graphs," *Numerische Mathematik, Vol.1*, 1959, pp.269-271.

[24] D. Cohen-Steiner and J.M. Morvan, "Restricted Delaunay triangulations and normal cycle", *Proc. of 19th Annual ACM Sym. on Computational Geometry*, 2003, pp.237-246.

[25] Computational geometry algorithm library, http://www.cgal.org.

[26] P. Shilane, M. Kazhdan, P. Min, and T. Funkhouser, "*The Princeton shape benchmark,*" *Proc. of Shape Modeling International*, 2004.

[27] Y. J. Park *et. al.*, "Mesh segmentation with the geodesic means of clustering of sharp vertices," *IASTED08*, 2008.



Kwan-Hee Yoo

He received the B.S in computer science and statistics from Chonbuk University, Korea in 1985, and also received M.S., Ph.D. in computer science from KAIST, Korea in 1988, 1995 respectively. He is the faculty in Chungbuk university from 1997. His main research interests include computer graphics, e-learning, medical and dental application, and 3D games.

Chan Park

He received the B.S., M.S in computer education from Chungbuk University, Korea in 2003, 2007 respectively. He is under the doctorial course in information and industry engineering from Chungbuk university. His main research interests include computer graphics, e-learning, and medical and dental application.

Young-Jin Park
He received the B.S., M.S in computer science from Chonbuk University, Korea in 1986, 1988 respectively and also received Ph.D. in information and industry engineering from Chungbuk university in 2009. His main research interests include computational geometry and computer graphics.

Jong-Sung Ha
He received the B.S in computer engineering from Seoul University, Korea in 1984, and also received M.S., Ph.D. in computer science from KAIST in 1986, 1996 respectively. He is the faculty in Woosuk University from 1990. His main research interests include applied computational geometry, computer graphics, and 3D contents.