A Topology-Preserving Polygonal Simplification Using Vertex Clustering

Takayuki Kanaya[*]
Hiroshima International University

Yuji Teshima[†]
Shizuoka Institute of Science and Technology

Koji Nishio[‡]
Osaka Institute of Technology

Ken-ichi Kobori[§]
Osaka Institute of Technology

Abstract

Product developers have used a lot of polygon data, approximated from 3D-CAD data, as a collaboration tool on the Internet. It is difficult to deal with this data for example with respect to transmission, computational cost, or rendering, so simplification algorithms are required for data compression. In general, a vertex-clustering algorithm in simplification algorithms is very fast, although it has the problem that topology information is not preserved and for some applications, such as 3D-CAD, it is important to preserve topology information. We define topology information in this paper as genus on the polyhedron and 2-manifold. In this paper, we propose a topology-preserving simplification method using a depth-first search tree on a vertex-clustering algorithm. Our method does not lose the advantage that vertex-clustering algorithms are fast yet it solves the problem of lost topology information. Experimental results show that this method is effective and is easily adapted to space division algorithms of other vertex-clustering algorithms

Keywords: simplification, topology-preserving, vertex-clustering, polygon, level of detail

1 Introduction

In recent years, product developers have used the Internet as a collaboration tool, influenced by the globalization of development blocks. For example, the design data developed at a certain development block is shared in a design review through the Internet. In this case, 3D-CAD data is approximated by polyhedron models because it is difficult to deal with 3D-CAD data given computational cost, data processing, and so forth. Additionally, we often have gigabytes or terabytes of model data when modelling, for example, cars or airplanes. It is difficult to transmit model data fast on the network or render it in real time. Therefore technology is required to change the level of detail of multiresolution representations of models according to a user's needs. When treating product objects, topology information is important, with respect to the interference check of objects, the manufacture technology by a metallic mold, and so forth; hence topology information has to be preserved.

In this paper, we propose a technique for preserving topology information in the vertex-clustering algorithm by simplifying models which consist of a lot of polygon data without losing the speed advantage of the vertex-clustering alogorithms. We define a topology-preserving algorithm in this paper, as one that preserves the genus in a polyhedron model and will not generate nonmanifold surfaces such as those that intersect themselves.

[*]e-mail: t-kanaya@hw.hirokoku-u.ac.jp
[†]e-mail:teshima@cs.sist.ac.jp
[‡]e-mail:nishio@is.oit.ac.jp
[§]e-mail:kobori@is.oit.ac.jp

2 Related Work

The vertex-clustering algorithm has been proposed by Rossignac and Borrel [Rossignac and Borrel 1993]. This is the method where a cell, which includes all the vertices that exist in 3D space, is uniformly divided; all vertices within each grid cell are collapsed to a single representative vertex, which may be one of the input vertices or some weighted combination of the input vertices. Some triangles degenerate to edges or points. A number of other vertex-clustering algorithms have been proposed. Low and Tan have proposed a modified vertex-clustering algorithm using floating cells rather than a uniform grid [Low and Tan 1997]. The floating-cell clustering leads to more consistent simplification. Since the importance of vertices controls the positioning of clustering cells, the unpredictable simplification artifacts are greatly reduced. Luebke and Erikson have proposed the algorithm using a hierarchical grid in the form of an octree, which is called Tight Octree [Luebke and Erikson 1997]. This vertex tree allows for a dynamic, view-dependent adaptation of the level of detail. Kanaya and Kobori have proposed the improved Tight Octree algorithm allowing for cell size [Kanaya and Kobori 2001]. This method is easier to control the LOD level. Lindstrom has used QEM(quadric error metrics) [Garland 1999] to improve the positioning of the representative vertices [Lindstrom 2000]. Shaffer and Garland have proposed BSP-Tree partitioning to control the LOD level [Shaffer and Garland 2001]. Garland and Shaffer have used QEM to improve the positioning of the representative vertices, too [Garland and Shaffer 2002]. However, the simplified shape depends on how the user partitions the 3D space. These vertex-clustering algorithms feature low computational costs such as $O(n)$. Although these algorithms reduce the clustering errors, the topology preservation problem has not been tackled.

A lot of topology-preserving simplification algorithms on isosurface extraction from volume datasets have been studied. Gerstner and Pajarola have proposed an extraction of a simplified isosurface while preserving topology information [Gerstner and Pajarola 2000].This algorithm preserves the isosurface topology by finding critical points using a lookup-table method. Ju et al. have proposed an algorithm simplifying a volume grid tagged with material information [Ju et al. 2002]. There are existing techniques for isosurface and volume topology simplification. To simplify the topology, this algorithm usually alters the voxel values. Zhang et al. apply a similar idea to Ju et al.'s algorithm [Zhang et al. 2004]. They have adopted an enhanced cell, which has its own iniside/outside classification information. Also, Wood et al. have proposed an algorithm which makes a Reeb Graph of an original model and controls topology information by controlling the number of small handles [Wood et al. 2004]. The goal of these topology simplification algorithms is to extract a simplification while preserving its topology, so that each volume grid has some information, such as sign, inside/outside. Our algorithm is an extended vertex-clustering algorithm; every volume grid does not have information, so that our algorithm can be applied to other vertex-clustering algorithms.

3 The topology-preserving simplification

We now present an overview of our topology-preserving simplification algorithm. Our algorithm consists of 3 main parts: a space division for grading, a graph search for topology-preservation, and a simplification operator. We will describe each algorithm in the following subsections.

3.1 The space division algorithm

We applied the algorithm of [Kanaya and Kobori 2001], which we have proposed, to an arbitrary 3-D space division. It is believed that this algorithm can be applied even when space division algorithms of other vertex-clustering algorithms are used, because the model simplification is not influenced by other cells made by space division. The space division algorithm which we have outlined consists of the following five steps:

1. A cell, which includes all the vertices that exist in 3D space is generated. The cell is a minimized cuboid, which is divided by a domain parallel to an x-y plane, a y-z plane, and a z-x plane so that two or more vertices can be included.
2. The cell is equally divided by eight, and then eight cuboids are made.
3. The degree of the LOD (level of details) is determined by the longest edge of each cuboid.
4. The number of vertices belonging to each cuboid is equally divided into eight; dividing will end if the number of the vertices, which exists in the cuboid, is only one.
5. Otherwise, each remaining cuboid is changed into a cell and processing returns to step 2.

By this procedure, an octree, as shown in Figure 1, can be generated. Considering the degree of the LOD in this octree, the root node is a level 0 and the degree of the LOD becomes larger as the node is closer to a leaf node.

3.2 The topology-preserving algorithm

We now present the topology-preserving algorithm. It is a DFS (depth-first search) algorithm. We consider that the vertices and the edges, including the cell of arbitrary level i, are in the graph. When searching a graph consisting of a set of vertices and a set of edges

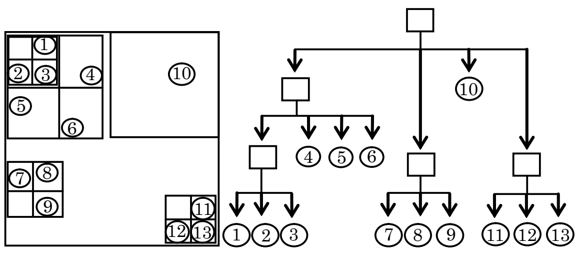

Figure 1: An example of generated octree (in 2D representation).

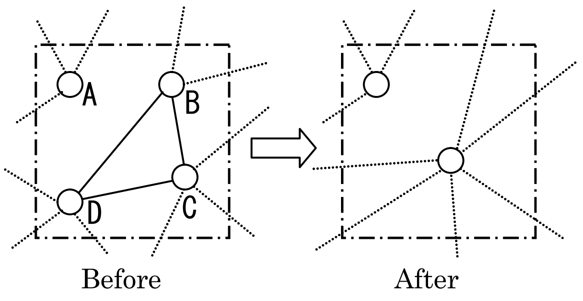

Figure 2: Candidate !œedges to be simplified (in 2D representation).

with relationships between the vertices at the arbitrary level i, according to a DFS algorithm, there are some connected components. Vertices in each connected component are replaced by a single representative vertex, as seen in Figure 2, according to a simplification operator described in the next subsection. The candidates for simplifying the model in level i on the octree are level i+1, because the model is recursively simplified from the greatest level of the octree to the user-defined level. The candidates of the model simplification are only vertices in the same cell and edges, which connect the vertices that exist in the same cell. As a result, the cells made by dividing space, which is the feature of the vertex-clustering algorithm, do not influence each other. The computational cost of this algorithm is $O(n)$.

We show an example using Figure 2. Consider an area that is surrounded with dash lines as a subdivided cell in a degree of the LOD. The circles are vertices. The solid lines and dotted lines are edges. Where the solid lines are edges, which connect the vertices that exist in the same cell, the dotted lines are edges, which connect the vertices that do not exist in the same cell. First, by searching the graph using the DFS algorithm, some connected components are found. In this case of Figure 2, there are 2 connected components, the vertex A and the connected component consisiting of vertices B, C, and D. Next, the model is simplified in each component. However, the candidates for simplifying the model are the only solid lines, so vertex A is not simplified. In the component consisiting of vertices B, C, and D, the only solid lines are simplified until collapsing the vertices onto a single representative vertex. When adapting conventional vertex-clustering algorithms to models, the vertices in each cell are replaced by a single representative vertex. However, when adapting our algorithm to models, the vertices in each cell are replaced by some representative vertices; as a result, the topology information can be preserved.

3.3 The simplification operator

We applied the QEM method, which has been proposed in [Garland 1999], as the algorithm of model simplification. First we will describe the outline of the QEM method. Model simplification in the QEM method is a vertex pair contraction operation. This is an operation, which collapses the two vertices (v_i, v_j) which are an edge, or which are not an edge but have the property $|v_i - v_j| < \tau$, where τ is a user-defined threshold. The operation which collapses only the two vertices connected by the edge is expressed as $\tau = 0$. The evaluation value $Q_{\bar{v}}$ is the sum of Q_{v_i} and Q_{v_j}, where, $\bar{v}$ is the vertex which is generated by collapsing the two vertices. The evaluation value Q is defined as the error at that vertex, the sum of the squared distances of the vertex to all the planes in the corresponding set.

We simplify models by the QEM method applied to every connected graph in each cell. If the vertex pair contraction operation is used, the two vertices, which are not connected in an edge, are also collapsed. This implies that topology information is not necessarily preserved. Therefore, the vertex pair contraction operation is adapted only for edges, that is τ= 0.

4 Experiment and Results

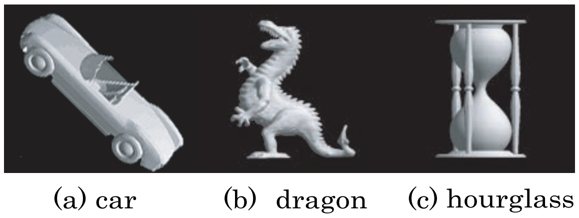

Figure 3: Experimental models.

Table 1: Number of polygons and processing time on experimental models.

Model	Original	OOC	Our method	Time
car	105,056	15,688	15,204	340 ms
dragon	108,586	19,030	18,976	330 ms
hourglass	32,320	6,260	6,020	40 ms

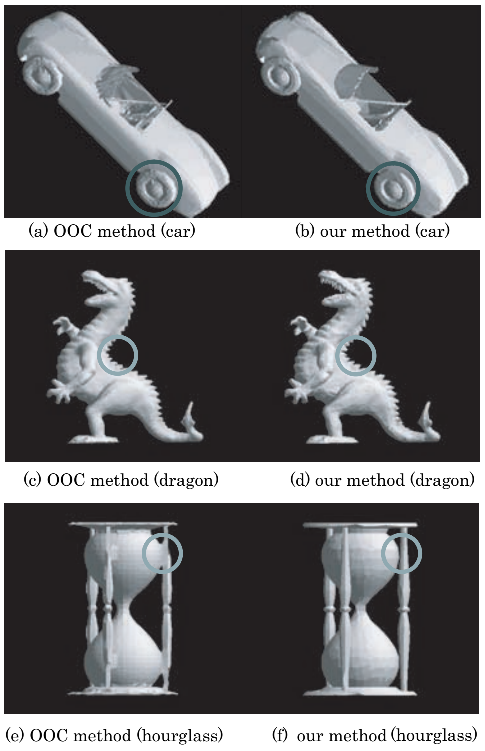

Figure 4: Simplified results using our method and the OOC method.

To evaluate the performance of our algorithm, it was compared to the algorithm proposed in [Lindstrom 2000] (OOC) using Metro tool, which has been proposed in [Cignoni et al. 1996]. Figure 3 shows some original models from the experiments. All simplifications were performed on the IBM PC compatibles (CPU: Pentium4 1.7GHz, RAM: 512MB, OS: Windows XP Professional). We

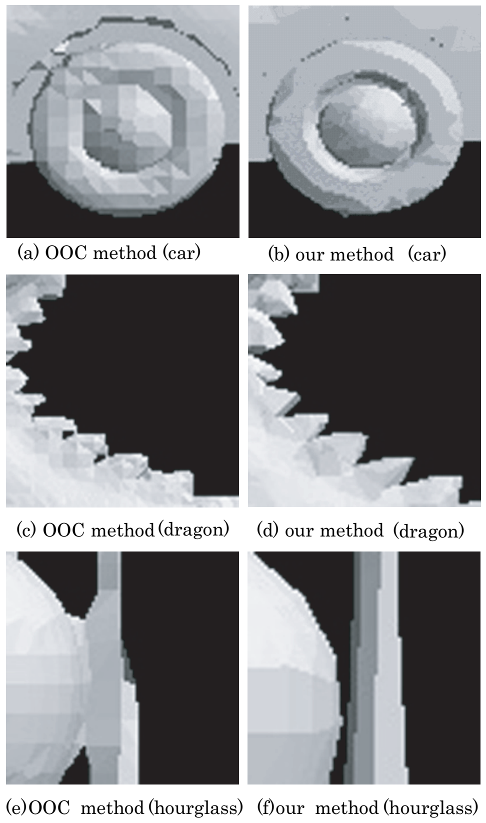

Figure 5: Magnified images of certain portion in Figure 4.

measured the difference between the original model and the model generated by our method and the OOC method. Table 1 shows the number of polygons in the original models, the number of polygons simplified by the OOC method, the number of polygons simplified by our method and the time difference between the two methods. Figure 4 shows the visual results generated by our method and by the OOC method. The number of polygons of each model is given in Table 1. Figure 5 shows the magnified images of certain portions shown in Figure 4. Figure 6 shows the mean error by Metro tool.

A visual comparison of the simplification results of our method and the OOC method in Figure 4 and 5 can be easily made. As you can see from Figure 5(a) and (b), the body and a tire simplified by the OOC method are joined, whereas the body and a tire simplified by our method are not joined. From Figure 5(c) and (d), the dorsal fins of the dragon simplified by the OOC method are connected, but the dorsal fins of the dragon simplified by our method are separated and maintain a sharp characteristic. From Figure 5(e) and (f), the hourglass simplified by the OOC method unified the glass and the pillar, although the hourglass simplified by our method is separated as in the original model. The experiments have shown that models simplified by our method preserve topology information and characteristics of shape. The measurements by Metro tool have shown that we have improved the accuracy quantitatively as well. The time difference between processing times of the topology-non-preserving and the topology-preserving algorithm was about 335ms in the case where the number of polygons was reduced by about 90,000 polygons from the original model. The time difference was about 40ms in the case where the number of polygons was reduced by about 2.6 million polygons. Therefore, we only sacrificed a miniscule amount of time.

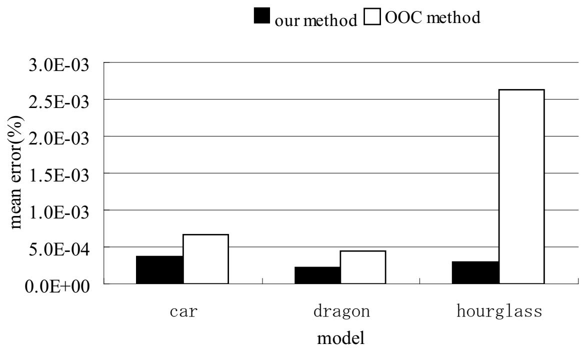

Figure 6: Comparison of our method and the OOC method. The mean error is absolute distance between meshes as a percentage of the diagonal of the mesh bounding box.

5 Conclusion

In this paper, we have proposed an optimal topology-preserving simplification on the vertex-clustering algorithm while still maintaining a high speed with only a minute difference in processing time. The measurements by Metro tool have shown that the errors of the models generated by our method have decreased in comparison with those generated by the OOC method. The experiments have shown that our method preserved topology information and the rapidity of the vertex-clustering algorithm.

For future work, we will consider the possibility for the user to control the number of the genus.

Acknowledgement

We would like to thank P. Cignoni et al. for the Metro tool. This work has been supported by Sasagawa Grants for Science Fellows of The Japan Science Society.

References

CIGNONI, P., ROCCHINI, C., AND SCOPIGNO, R. 1996. Metro: measuring error on simplified surfaces. Tech. rep., Paris, France, France.

GARLAND, M., AND SHAFFER, E. 2002. A multiphase approach to efficient surface simplification. In *VIS '02: Proceedings of the conference on Visualization '02*, IEEE Computer Society, Washington, DC, USA, 117–124.

GARLAND, M. 1999. *Quadric-based polygonal surface simplification*. PhD thesis, Carnegie Mellon University. Chair-Paul Heckbert.

GERSTNER, T., AND PAJAROLA, R. 2000. Topology preserving and controlled topology simplifying multiresolution isosurface extraction. 259–266.

JU, T., LOSASSO, F., SCHAEFER, S., AND WARREN, J. 2002. Dual contouring of hermite data. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 339–346.

KANAYA, T., AND KOBORI, K. 2001. Real time simplification in polygon environment. In *9th International Conference on Human-Computer Interaction, HCI International 2001, EP &CE UAHCI*, 223–224.

LINDSTROM, P. 2000. Out-of-core simplification of large polygonal models. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 259–262.

LOW, K.-L., AND TAN, T.-S. 1997. Model simplification using vertex-clustering. In *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*, ACM Press, New York, NY, USA, 75–82.

LUEBKE, D., AND ERIKSON, C. 1997. View-dependent simplification of arbitrary polygonal environments. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 199–208.

ROSSIGNAC, J., AND BORREL, P. 1993. Multi-resolution 3d approximations for rendering complex scenes. In *Geometric Modeling in Computer Graphics*, Springer-Verlag, New York, NY, USA, 455–465.

SHAFFER, E., AND GARLAND, M. 2001. Efficient adaptive simplification of massive meshes. In *VIS '01: Proceedings of the conference on Visualization '01*, IEEE Computer Society, Washington, DC, USA, 127–134.

WOOD, Z., HOPPE, H., DESBRUN, M., AND SCHRÖDER, P. 2004. Removing excess topology from isosurfaces. *ACM Trans. Graph. 23*, 2, 190–208.

ZHANG, N., HONG, W., AND KAUFMAN, A. 2004. Dual contouring with topology-preserving simplification using enhanced cell representation. In *VIS '04: Proceedings of the conference on Visualization '04*, IEEE Computer Society, Washington, DC, USA, 505–512.