A New Mesh Simplification Algorithm Based on Triangle Collapses

PAN Zhigeng (潘志庚)[1,2], ZHOU Kun (周 昆)[1] and SHI Jiaoying (石教英)[1]

[1]*State Key Laboratory of CAD & CG, Zhejiang University, Hangzhou 310027, P.R. China*

[2]*Department of Computing, Hong Kong Polytechnic University, Hong Kong, P.R. China*

E-mail: {pzg, kzhou, jyshi}@cad.zju.edu.cn

Received April 9, 1999; revised April 15, 2000.

Abstract In this paper a new mesh simplification algorithm based on triangle collapses is presented. The algorithm can provide efficient error management and simplify the original mesh greatly. Progressive meshes may be constructed with triangle collapsing operation. To make continuous transition between level of detail (LOD) models possible, a method for interpolating is also presented. Examples illustrate the efficiency of the algorithm.

Keywords mesh simplification, triangle mesh, level of detail, progressive mesh

1 Introduction

Representing complex models is still a challenge in spite of the research achievements of powerful graphics hardware. Many computer graphics applications require complex, highly detailed model to maintain a convincing level of realism. Consequently, models are often created or acquired at a very high resolution to accommodate this need for detail. However, the complexity of such models is not always necessary, and the computational cost of using a model is directly related to its complexity, so it is useful to have simpler versions of complex model[1]. Recent work on surface simplification has focused on this goal.

Surface simplification algorithm may be categorized into 3 classes: adaptive subdivision[2], geometry removal[3–10] and re-sampling[10–12]. Many algorithms fall into the second category. They may be broadly categorized into 3 sub-classes.

• *Vertex or triangle decimation*[3,9]: Schroeder[9] described an algorithm which selects a vertex for removal iteratively, removes all adjacent faces, and re-triangulates the resulting hole.

• *Vertex clustering*: Rossignac[12] presented a surface simplification algorithm based on vertex clustering. In his algorithm, a bounding box is placed around the original model and divided into a grid. Within each cell, the cell's vertices are clustered together into a single vertex, and the model faces are updated accordingly. This process can be very fast and make drastic topological alternative to the model.

• *Edge contraction or Edge collapse*[4,6,7]: Several algorithms simplify models by interactively contracting edges. The essential difference between these algorithms lies in how they choose an edge to contract. Garland's algorithm[4] can rapidly produce high quality approximations of polygonal models. It uses iterative contractions of vertex pairs to simplify models and maintains surface error approximations using quadric matrices.

In polygonal simplification algorithms based on vertex or triangle removal, the re-triangulation process is required to apply to the hole left by the removed vertices or triangles. It is obvious that the re-triangulation is a time consuming operation. On the other hand, the simplification algorithms based on edge collapse or triangle collapse do not require re-triangulation (See Fig.1 and Fig.2).

This research work is supported by the Fellowship of Hong Kong Polytechnic University and the National Natural Science Foundation of China (Nos. 69823003, 60083009).

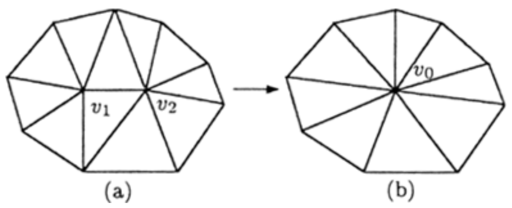

Fig.1. Edge collapse operation, (v_1, v_2) is collapsed into v_0. (a) Before collapse. (b) After collapse.

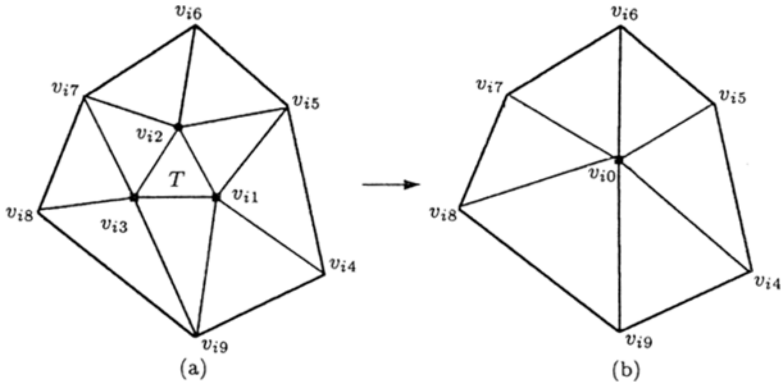

Fig.2. Tgriangle collapsing operation, (v_{i1}, v_{i2}, v_{i3}) is collapsed into v_{i0}.
(a) Before collapse. (b) After collapse.

The basic operation in Garland's algorithm[4] is edge collapse. The algorithm computes error matrix for every vertex in the original mesh. The location of the new vertex is determined by computing the cost for contracting edges. The error matrix of the new vertex is the sum of error matrices of the two vertices of the edge. However, as pointed out by the authors, the error computing process may result in accumulated error. The primary simplification operations of the algorithm presented by Isler *et al.*[8] are edge collapse and triangle collapse. Fig.2 illustrates the basic idea of a triangle collapsing operation. In their algorithm, a triangle is collapsed into one of the vertices of the triangle.

In this paper we present a new mesh simplification algorithm based on triangle collapse. Compared with the algorithm of Isler *et al.*[8], the location of the new point generated by the contracting operation is the optimized point according to the computation of distance of point to plane. We give an efficient method for transmitting simplification error, which can control the error between the simplified mesh and the original mesh. Compared with the algorithm of Garland[4], the effect of one triangle collapsing operation in our algorithm is equal to that of two edge collapsing operations in Garland's algorithm. In addition, accumulated error resulted from the computation of the new point is avoided.

The rest of this paper is organized as follows. Section 2 describes the simplification algorithm. Section 3 gives the method for constructing progressive meshes based on triangle collapsing operation. To support the continuous transition between different LOD models, we present a method for interpolation in Section 4. In Section 5, the implementation techniques and experimental results are presented. Finally, conclusions are drawn in Section 6.

2 Algorithm Description

To facilitate the description of our algorithm, we first introduce some basic symbols and concepts.

2.1 Definitions

Definition 1. *Given a set of triangles, if each triangle has a common edge with every adjacent triangle, then these triangles are called triangle mesh (TM), TM may be expressed by a vertex set V and a triangle set T, where $V = (V_1, V_2, \dots, V_n)$, $T = (T_1, T_2, \dots, T_m)$.*

Definition 2. *For an edge in TM, if it is shared by only one triangle, then this edge is called boundary edge, and the vertices of this edge are called boundary vertices, and the triangle containing this edge is called boundary triangle.*

Definition 3. *For a vertex V_i in TM, the union of which contain vertex V_i are called the relative triangle set of vertex V_i.*

Definition 4. *For a triangle T_i in TM, the union of triangles interrelated t the vertices of T_i is called the interrelated triangle set C_j of the triangle T_i.*

2.2 Triangle Collapsing Operation

The basic simplifying operation in our algorithm is triangle collapse. In Fig.2(a), for triangle $T_i = \{v_{i1}, v_{i2}, v_{i3}\}$, we can find out the interrelated triangle set of this triangle T_i, apply triangle collapsing operation to T_i, thus three vertices are contracted into one vertex v_{i0}. The result is shown in Fig.2(b).

It is obvious that there are two problems to be solved for triangle collapses. The first is the cost for triangle collapses, i.e., the error resulted from the contracting operation. The second is how to choose the location of v_{i0} in order to minimize the error of triangle contracting operation.

To solve these problems, we allocate a 4×4 symmetric matrix Q_i for each triangle T_i in the original mesh. If the triangle T_i is collapsed into $v_{i0} = [x_{i0}\ y_{i0}\ z_{i0}\ 1]^{\mathrm{T}}$, then the cost or the error for this contracting operation is determined by the following equations.

$$\begin{aligned}\varepsilon(T_i) = v_{i0}^{\mathrm{T}} Q_i v_{i0} = & q_{i11}x_{i0}^2 + 2q_{i12}x_{i0}y_{i0} + 2q_{i13}x_{i0}z_{i0} + 2q_{i14}x_{i0} + q_{i22}y_{i0}^2 \\ & + 2q_{i23}y_{i0}z_{i0} + 2q_{i24}y_{i0} + q_{i33}z_{i0}^2 + 2q_{i34}z_{i0} + q_{i44} \end{aligned} \tag{1}$$

There may be several choices for the location of v_{i0}. The simplest one is to choose the center of the triangle T_i, and the best one is to choose the location of v_{i0} which minimizes $\varepsilon(T_i)$. We evaluate the partial derivatives of x_{i0}, y_{i0} and z_{i0} in (1), and set them to 0, i.e., $\partial\varepsilon(T_i)/\partial x_{i0} = \partial\varepsilon(T_i)/\partial y_{i0} = \partial\varepsilon(T_i)/\partial z_{i0} = 0$. Thus we can obtain the following linear equations.

$$\begin{bmatrix} q_{i11} & q_{i12} & q_{i13} & q_{i14} \\ q_{i12} & q_{i22} & q_{i23} & q_{i24} \\ q_{i13} & q_{i23}. & q_{i33} & q_{i34} \\ 0 & 0 & 0 & 1 \end{bmatrix} v_{i0} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{2}$$

If the linear equations are solvable, then the location of v_{i0} is determined by (3); otherwise, we choose one point from the center of triangle T_i, three vertices, and the midpoints of three edges in the triangle as the location of v_{i0} to minimize (1).

$$v_{i0} = \begin{bmatrix} q_{i11} & q_{i12} & q_{i13} & q_{i14} \\ q_{i12} & q_{i22} & q_{i23} & q_{i24} \\ q_{i13} & q_{i23} & q_{i33} & q_{i34} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{3}$$

2.3 Computation of Error Matrices and Error Propagation

In the previous subsection, the 4×4 error matrix Q_i may have different choices according to the error criteria selected. In our algorithm, we choose the distance between the vertex and the average plane[4] as the error measuring method.

For each triangle T_i in the original mesh, we may obtain the interrelated triangle set C_i by evaluating the union of triangles which are interrelated to its three vertices. The approximation error (say $\varepsilon(T_i)$) resulted from the triangle collapsing process is defined as the maximum value of the distances of the new vertex ($v_{i0} = [x_{i0}\ y_{i0}\ z_{i0}\ 1]^{\mathrm{T}}$) to each triangle in the triangle set. We have:

$$\varepsilon(T_i) = \sum_{p \in C_i} (p^{\mathrm{T}} v_{i0})^2 \tag{4}$$

where $p = [a\ b\ c\ d]^{\mathrm{T}}$ represents the triangle in triangle T_i's interrelated triangle set C_i, whose plane is defined by the equation $ax + by + cz + d = 0$, where $a^2 + b^2 + c^2 = 1$. This approximation error is similar to that of Garland's algorithm. (4) may be changed to the form of (1) accordingly.

$$\varepsilon(T_i) = \sum_{p \in C_i} ((v_{i0}^{\mathrm{T}} p)(p^{\mathrm{T}} v_{i0})) = \sum_{p \in C_i} (v_{i0}^{\mathrm{T}} (pp^{\mathrm{T}}) v_{i0}) = \sum_{p \in C_i} (v_{i0}^{\mathrm{T}} M_p v_{i0}) \tag{5}$$

where M_p is a 4×4 symmetric matrix. M_p is defined as error matrix Q_i; and we have

$$M_p = pp^{\mathrm{T}} = \begin{bmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{bmatrix} \tag{6}$$

For boundary triangle, the interrelated triangle set of the triangle cannot form a triangle loop. Thus, to preserve the boundary feature of the original model, for each boundary edge in the triangle set we generate a plane that meets the following constraints: (i) the boundary edge is on the plane; (ii) the plane is perpendicular to the boundary triangle where the boundary edge resides. These planes are considered when computing error matrices. We have found that this method works quite well for preserving the boundary discontinuity.

For each triangle in the original mesh, we may evaluate its error matrix according to (6). If we apply contracting operation to some triangles in original mesh, then the coordinates of those triangles which share common vertex with collapsed triangles (if exist) changed, and the problem is whether the error matrices need to be re-computed? In the simplest case, re-computation is not required, and the error matrix is considered as the local error matrix. But it cannot control the error between the simplified mesh and the original mesh.

However, we adopt the method of re-computation, and transmit the error generated by each triangle collapsing operation. In Fig.2, the triangle $T_i = \{v_{i1}, v_{i2}, v_{i3}\}$ is collapsed into vertex v_{i0}. It is easy to find out the physically interrelated triangle set of v_{i0} in Fig.2(b). In the view of error propagation, we consider the triangle set interrelated logically to v_{i0} as the interrelated triangle set to triangle T_i. Since the vertex v_{i0} is generated with the combination of vertices v_{i1}, v_{i2} and v_{i3}, we need to deal with the triangle set interrelated to vertices v_{i1}, v_{i2} and v_{i3} when we compute the error caused by collapsing operation corresponding to v_{i0}. Therefore, for triangles interrelated to v_{i0} physically after contraction of triangle T_i, it is required to re-compute the triangle set which is interrelated to them logically, and the error matrices are re-evaluated. By this means, we can propagate the error caused by each triangle collapsing operation. The experience shows that this method for propagating error is very efficient.

2.4 Algorithm Outline

Our mesh simplification consists of the following steps:

Step 1. Compute the error matrix Q_i for each triangle T_i.

Step 2. For each triangle T_i, compute the location of contracting vertex v_{i0} according to its error matrix Q_i, and compute its corresponding approximation error according to $v_{i0}^{\mathrm{T}} Q_i v_{i0}$.

Step 3. Arrange the triangles into a sequence according to the contracting cost or error (in ascending order).

Step 4. Fetch the triangle whose contracting error is the smallest (the first one in the triangle sequence), apply triangle contracting operation, and update corresponding information.

Step 5. If the triangle sequence is empty or the error requirement is met, then go to Step 6, else go to Step 4.

Step 6. Finish.

3 Construction of Progressive Mesh Based on Triangle Collapses

Hoppe[7] introduced the concept of progressive mesh representation mechanism. He also presented a method for constructing progressive mesh based on edge contractions. For an original mesh model, a simplified model is obtained by an edge contracting operation. And the edge contracting operation is recorded. When we get the most simplified version of the mesh, the sequence of edge contraction is obtained. If we apply the reverse operation (vertex split) to the simplified version of the mesh, we can get any LOD models between the original mesh model and the most simplified model[13]. Since Hoppe presented the energy optimization method[15] for choosing edges for contraction, the whole computing process is very time consuming.

In this paper, we also present a method for constructing progressive mesh based on triangle collapsing operation. In Fig.2, when we apply the triangle collapsing operation to the triangle T_i, the three vertices v_{i1}, v_{i2} and v_{i3} are collapsed into the new vertex v_{i0}. Similar to the method of Hoppe, we record the coordinates of vertices v_{i1}, v_{i2} and v_{i3}, their number in the vertex table, and the vertex number of surrounding vertices $v_{i4}, v_{i5}, v_{i6}, v_{i7}, v_{i8}$ and v_{i9}. With this information, we can get the model in Fig.2(a) from the model in Fig.2(b). For an initial mesh model M_n, assume n is the number of vertices in the mesh, and $0 < m < n$, when each triangle collapsing operation is applied, the corresponding collapsing information d_i is recorded, where $d_i = \{\{v_{i0}\}, \{v_{i1}, v_{i2}, v_{i3}\}, \{v_{i4}, v_{i5}, v_{i6}, v_{i7}, v_{i8}, v_{i9}\}\}$. Thus, when $(n - m)$ triangle contracting operations are performed, we can get the simplified model M_m, and also the sequence of contracting information $\{d_n, d_{n-1}, \dots, d_{m+1}\}$. With this sequence of information $\{d_n, d_{n-1}, \dots, d_{m+1}\}$, we can obtain the model sequence $\{M_n, M_{n-1}, \dots, M_{m+1}\}$ from the base model M_m.

Progressive meshes offer an elegant solution for a continuous resolution representation of polygonal meshes. They present a novel approach to storing, rendering, and transmitting meshes by using a continuous-resolution representation. It is obvious that this representing mechanism will be applied widely in the future.

4 Interpolation between Different LOD Models

Turk[16] presented an interpolation algorithm between two different LOD models. After a detailed analysis, we found the limitation of the algorithm. It is required that vertices on the coarse model should be a subset of the vertices on the detailed model, so it is not suit for LOD models generated by simplification algorithm based on edge collapses or triangle collapses. In addition, the implementation of the algorithm is very complex.

In this section, we present a very simple but efficient method for model interpolation. It works well for any kind of LOD models. The basic idea of our algorithm is to find out corresponding points in both models, and perform linear interpolation directly. The algorithm is described as the following.

Step 1. For every vertex in the coarse model, find the nearest vertex in the detailed model. Thus it builds the corresponding relation between the two vertices.

Step 2. For those vertices in the detailed model which do not have corresponding vertices, try to find the nearest vertex in the coarse model. It also builds the corresponding relation between the two vertices.

Step 3. Given an interpolation parameter t ($0 \leq t \leq 1$), for each vertex P_h in the detailed model, we assume that its corresponding vertex in coarse model is P_l, then the interpolated point is P ($P = tP_l + (1-t)P_h$) on the in-between model.

Step 4. For a given error measure ε, if the distance between two adjacent vertices is less than ε, then the two vertices are collapsed into one, and the degenerated triangle is removed.

Step 5. Finish.

It is obvious that our method is very straightforward. And we construct octree for the coarse model and the detailed model to accelerate the executing speed in Step 1 and Step 2.

5 Implementation and Experimental Results

5.1 Data Structures

Since the algorithm may deal with meshes with a large number of vertices, a well-designed data structure is required. By this means we can process very large models, and the simplifying speed is very fast.

In our algorithm, we employ the following data structures: (i) a vertex table; (ii) a triangle table; (iii) a triangle table composed by triangles which are interrelated to each vertex (including physical table and logical table); (iv) an error matrix table.

5.2 Experimental Results

The algorithm is implemented on Sun Sparc Workstation and IBM RS6000 Workstation with programming language C. It has been integrated into an LOD tool on PC. Some experimental results are shown in Fig.3 to Fig.6. From these figures, we can see that our algorithm is efficient for both smooth models and non-smooth models.

In Fig.3, the simplified model shown in Fig.3(b) is very similar to the original model. And the result is also acceptable when the triangles are decimated 99% (Fig.3(d)). (Please refer to the inside back cover for Fig.3)

In Fig.4, we show shaded pictures of a terrain model. Since the triangle number in the original model is not very large, the model can only be simplified to a certain extent (the maximum percentage is 79%). Most features are preserved in simplified models. In addition, our algorithm especially works well for surfaces composed of triangles. In Fig.5, (a) is the original model consisting of 5000 triangles. The simplified models at three levels of detail (75%, 95% and 99% triangles decimated) are shown in Figs.5(b), (c) and (d) respectively. The boundary features are well preserved. In Fig.6, (a) and (f) are the original coarse model and detailed model respectively, (b), (c), (d) and (e) are interpolated models with different interpolation parameters. (Please refer to the inside back cover for Fig.4, Fig.5 and Fig.6.)

Table 1. Execution Time Comparison between Our Algorithm and Garland's

Original model (Triangle number)	Simplified model (Triangle number)	Garland's algorithm[4] (s)	Our algorithm (s)
Surface (5000)	272	13.43	6.76
Terrain (8192)	1872	29.85	11.86
Bunny (69473)	2306	1777.18	332.10

6 Conclusion and Future Work

The polygonal simplification algorithm based on triangle collapses presented in this paper is very fast and can get better approximation quality. When the basic element of mesh model is polygons other than triangles, our algorithm can easily be extended to deal with the situation. It can be used to construct multiple LOD models that are widely used in interactive graphics application such as Virtual Reality and interactive visualization systems[1,14].

We may extend the idea of triangle collapses, and let more areas composed of triangles be collapsed, thus the execution time may be reduced further. Future work includes the following three aspects. (1) Find method for dividing the triangle mesh into areas for contraction efficiently. (2) Study more efficient error control method. (3) Apply the method to dynamic virtual environments.

Acknowledgments The authors are very grateful to the suggestions for improving the paper given by anonymous reviewers. Thanks also go to Graphics Research Group at the Computer Science Department in the University of Washington for providing the data models.

References

[1] Pan Zhigeng, Ma Xiaohu, Shi Jiaoying. The automatic generation of multiple levels of detail in virtual environment. *Journal of Software,* 1996, 7(9): 526–531. (in Chinese)

[2] DeHaemer Jr, Zyda M J. Simplification of objects rendered by polygonal approximations. *Computers and Graphics,* 1991, 15(2): 175–184.

[3] Cohen J, Varshney A, Manocha D. Simplification envelops. In *SIGGRAPH'96,* 1996, pp.119–128.

[4] Garland M, Heckbert P. Surface simplification using quadric error metrics. *Computer Graphics* (*SIGGRAPH Proceedings*), 1997, 31(3): 209–216.

[5] Hamann B. A data reduction scheme for triangulated surfaces. *CAGD,* 1994, 11(3): 197–214.

[6] Ronfard R, Rossignac J˙R. Full-range approximation of triangulated polyhedral. *Computer Graphics Forum,* 1996, 15(3): 67–76.

[7] Hoppe H. Progressive meshes. *Computer Graphics* (*SIGGRAPH Proceedings*), 1996, 3(30): 99–108.

[8] Isler Veysi, Lau R W H, Green Mark. Real-time multi-resolution modeling for complex virtual environments. In *Proceedings of VRST'96,* 1996, pp.11–19.

[9] Schroeder W J, Zarge J A. Decimation of triangle meshes. *Computer Graphics,* 1992, 26(2): 65–70.

[10] Cohen J, Olano M, Manocha D. Appearance preserving simplification. *Computer Graphics* (*SIGGRAPH Proceedings*), 1998, 3(32): 115–122.

[11] Zhou Kun, Pan Zhigeng, Shi Jiaoying. A new simplification algorithm based on vertex clustering. *Journal of Automation,* 1999, 25(1): 1–8 (in Chinese).

[12] Rossignac J, Borrel P. Multi-resolution 3D approximation for rendering complex scenes. In Geometric Modeling in Computer Graphics, Falcidieno B, Kunii T (eds.) Springer-Verlag, 1993, pp.455–465.

[13] Hoppe Hugues, Tony DeRose, Tom Duchamp *et al.* Mesh optimization. *Computer Graphics* (*SIGGRAPH Proceedings*), 1993, 3(27): 19–26.

[14] Pan Zhigeng *et al.* Time-critical computing in virtual environment. In *Proceedings of CAD/Graphics'95,* Wuhan, 1995.

[15] Erickson C. Polygonal simplification: An overview. Technical Report, TR96-016, Department of Computer Science, UNC-Chapel-hill, 1996.

[16] Turk Greg. Re-tiling polygonal surfaces. *Computer Graphics,* July 1992, 26(2): 55–64.

PAN Zhigeng is a professor at the State Key Lab of CAD & CG, Zhejing University, a member of the IS & T, and a senior member of the China Image and Graphics Association. He is a content editor for the International Journal of Virtual Reality, and on the Editorial Board of Journal of Image and Graphics. He has published more than 60 papers and is the author or co-author of three books. He acted as one of the Guest Editors for a Special Issue of Computers & Graphics. His current research interests include virtual reality, distributed graphics and multimedia.

ZHOU Kun is a graduate student of the Department of Computer Science and Engineering at Zhejiang University. His research interests include time critical rendering, multi-resolution modeling, image-based rendering/modeling. He has published around 10 papers in various journals and proceedings related to VR. In 1999, he was awarded Microsoft Fellowship.

SHI Jiaoying is a professor of the Dept. of Computer Sci. & Eng. at Zhejiang University, China. He is the Director of Academic Committee of State Key Lab of CAD & CG. He is a member of SIGGRAPH Education Committee, and on the Editorial Board of International Journal of Computers & Graphics. Since 1990 his work has been concentrated on ViSC and VR. He has published more than 100 papers and three books.

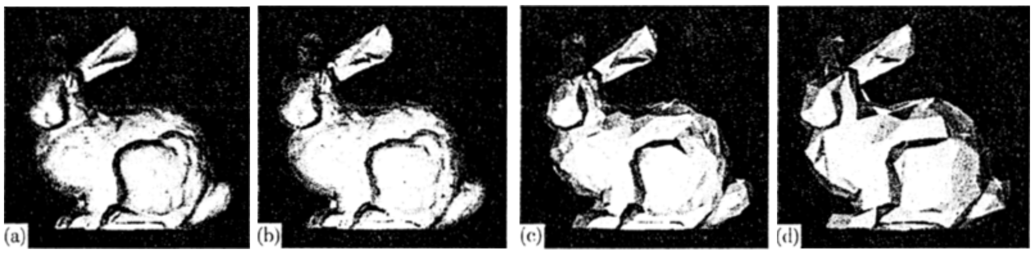

Fig.3. Simplification on bunny model. (a) Original model (69473 triangles). (b) Simplified model 1 (8310 triangles, 88% decimated). (c) Model 2 (1842 triangles, 97% decimated). (d) Model 3 (763 triangles, 99% decimated).

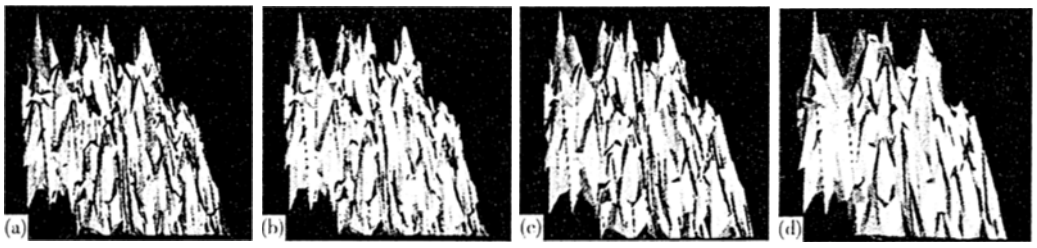

Fig.4. Simplification of terrain model. (a) Original model (6035 triangles). (b) Simplified model 1 (3704 triangles, 39% decimated). (c) Model 2 (2306 triangles, 62% decimated). (d) Model 3 (1268 triangles, 79% decimated).

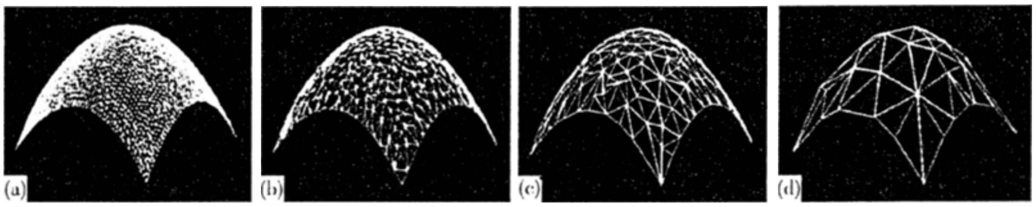

Fig.5. Simplification of surface model. (a) Original model (5000 triangles). (b) Simplified model 1 (1204 triangles, 76% decimated). (c) Model 2 (272 triangles, 95% decimated). (d) Model 3 (54 triangles, 99% decimated).

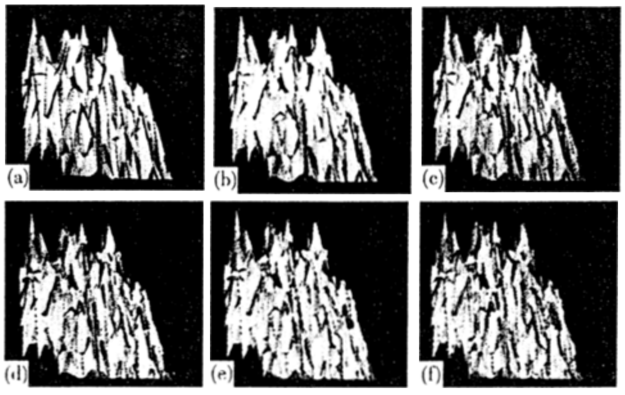

Fig.6. Interpolation between different LOD models. (a) Original model 1 (coarse model). (b), (c), (d) and (e) are in-between models (with $t = 0.2$, 0.4, 0.6, and 0.8 respectively). (f) Original model 2 (detailed model).