Mesh Surface Simplification Based on Constrained Energy

Hidenori Sato Akira Onozawa Hitoshi Kitazawa

NTT Integrated Information & Energy Systems Laboratories
3-1, Morinosato Wakamiya, Atsugi-shi, Kanagawa Pref., 243-0198
Japan
{hide, onoz, kitazawa}@aecl.ntt.co.jp

ABSTRACT

A triangle-mesh-surface-simplification algorithm based on a constrained energy function is proposed. In the algorithm, vertices are deleted iteratively according to the dimension-less energy function, which represents the cost of surface modification by vertex deletion and subsequent hole re-triangulation. The energy function is used both for selecting a vertex with the minimum energy for deletion and for constraining the target curvature of the surface to be simplified. During simplification, the target curvature of the surface to be simplified is modified gradually by increasing upper limit constraints. The experimental results show that the proposed algorithm has the advantage that a set of parameters can be applied to various-scale data composed of a variety of sizes and shapes with boundary edges.

Keywords: computer graphics, triangle meshes, mesh simplification, level of detail

1 INTRODUCTION

In many computer graphics applications, triangle meshes are mainly used for representing geometric surface models in three-dimensional space. On the other hand, the models are becoming larger and more complex because of the speed-up of computer hardware and the development of modeling tools, such as computer aided design systems and laser digitizers. However, rendering a few large and complex 3-D objects simultaneously in near-real time is difficult because of the large computational cost. When an object is viewed from a distance, it is sufficient to use a simplified model that represents it's global shape. Therefore, it is useful to generate a *level-of-detail* (LOD) model that contains appropriate multiresolution models for a full-complexity model. An LOD model is also useful when transmitting objects through a network such as the Internet. In this case, users could view the object quickly if the lowest level model is transmitted first. Furthermore, rendering time can be shortened if the lowest level model is used when objects are moving in 3-D space.

Many mesh surface simplification algorithms have been proposed in recent years [Heckb97a]. The algorithms can be classified into two types: surface reconstruction and iterative mesh reduction. A typical surface reconstruction algorithm uses marching cubes [Shekh96a]. The algorithm reconstructs a simplified model from the original model by downsampling the marching cubes. The obtained model is not the subset of the original. On the other hand, iterative mesh reduction algorithms delete a unit iteratively, and the obtained model is a subset of the original. That means the *progressive mesh* (PM) representation [Hoppe96a] can be applied to store the obtained data. The PM data structure stores the continuous sequence of simplified data and represents the arbitrary reduction rate. The proposed iterative mesh reduction algorithms can be classified into following four types in terms of the unit of deletion:

- deletion of vertices [Klein96a, Schro97b],
- deletion of edges [Hoppe96a, Popov97a],
- deletion of faces [Gieng97a],
- deletion of vertex pairs [Garla97a].

Klein et al. proposed an algorithm that uses the Hausdorff distance between the original mesh surface and the simplified one [Klein96a]. Vertices whose Hausdorff distance doesn't exceed the user specified distance are deleted iteratively. Schroeder et al. proposed an algorithm in which vertices are deleted according to the result of classification of vertices and the distance from vertices to either average plane for simple vertices or line for edge and boundary vertices [Schro97b]. Hoppe et al. proposed algorithms based on the iteration of the deletion of edges. First, the problems are formulated as an optimization problem of triangle meshes [Hoppe96a]. The order of the deletion of edges is determined according to the energy minimization function, which is composed of 1) distance, 2) spring, 3) scalar, and 4) discrete energy terms. In subsequent work, they generalized the problem to arbitrary triangulation and energy terms [Popov97a]. An edge which yields a minimum cost is deleted iteratively. Gieng et al. proposed an algorithm based on the iteration of the deletion of meshes [Gieng97a]. Each of the triangles is assigned a weight, which is given as a linear combination of 1) area, 2)curvature, 3) interior angle, and 4) topological measure. Triangles are fully or partially collapsed continuously. Garland et al. proposed an algorithm based on the iterative contraction of vertex pairs [Garla97a]. The pairs to be contracted are not only connected vertices but also unconnected vertices separated by a specified distance. The pair to be contracted is the one which yields minimum quadric error metrics that is given as a sum of squared distances from the position of vertex to initial planes. The position of a vertex after contraction is determined so that the cost is minimized.

All of the above algorithms use either surface curvature or the distance from either the original or simplified vertex to specified planes as a basic criterion. On the other hand, most of them define various error metrics and a variety of classifications of vertices and boundaries to prevent undesirable simplification of data. Such algorithms have many user-specified parameters that must be optimized for global features, scale, and the parts composing the original data. To get a good result, users have to run the simplification algorithm many times with different parameters.

The goal of mesh surface simplification is to take a mesh model as input and generate a simplified mesh model as output. The output model must be a geometric approximation of an input model. In this paper, we assume the target model for simplification is produced by automatic modeling tools such as CAD systems and laser digitizers. Such a model usually consists of a large number of triangle meshes. The meshes have common edges and vertices and do not intersect. This assumption doesn't lose generality because any polygon

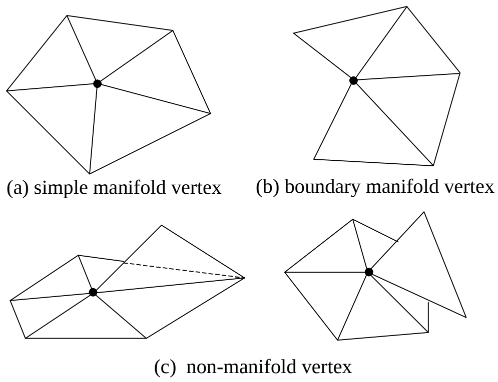


Figure 1: Vertex Classification

can be triangulated during a preliminary process.

Considering the optimization problem of proposed algorithms and working under the above assumption, we have developed a mesh surface simplification algorithm that has the following objectives:

- Applicable a set of parameters to various scale data with a variety of sizes and shapes.
- Generation of data at a arbitrary reduction rate.
- Deletion of simple and boundary manifold vertices.

In the algorithm, vertices are deleted iteratively according to a dimension-less energy function, which represents the cost of surface modification due to vertex deletion and resulting hole retriangulation. A vertex to be deleted is selected iteratively according to the energy. In addition, by dynamically constraining the energy during simplification, surface portions for simplification are changed accordingly from low to high curvature portions. The experimental results show that the proposed algorithm yields good results by using only a set of parameters for various scale data with a variety of sizes and shapes.

2 VERTEX CLASSIFICATION AND VDT

The topological characteristics of the vertices are classified into three types: *simple manifold*, *boundary manifold*, and *non-manifold* [Schro92a, Schro97b](see Fig. 1)[1]. A simple manifold vertex is a vertex that is connected by triangles whose outer loop makes a complete cycle, and every edge

[1]Although the category is simplified compared to those of original papers, our proposed algorithm uses our simplified one.

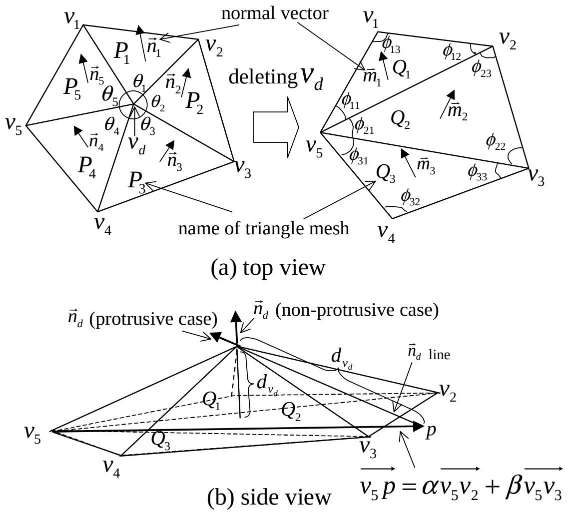


Figure 2: Example of VDT

in the loop shares just two triangles. A boundary manifold vertex is a vertex that is connected by triangles whose outer loop makes a semi-cycle, and two of the edges in the loop have only one triangle. A vertex that doesn't fall into these two categories is a non-manifold vertex. Modeling tools usually generate the simple and boundary manifold vertices.

Based on the above classification, our proposed algorithm deletes either a simple or a boundary manifold vertex and re-triangulates the resulting hole iteratively. We call this operation *Vertex Deletion and re-Triangulation* (VDT). Fig. 2 shows an example of VDT. The order of vertex deletion is determined from the VDT energy, which represents the cost of geometry modification by VDT. The function of VDT energy is given as a linear combination of energy terms which are calculated during VDT.

3 VDT ENERGY FUNCTION

The key to successfully obtaining a good geometric approximation is a good definition of VDT energy. It has to represent the geometry modification of VDT.

According to notations in Fig. 2, we define the VDT energy function, ΔE_{vdt}, as a linear equation with four energy terms:

$$\Delta E_{vdt} = a \cdot \Delta E_{vdist} + b \cdot \Delta E_{atotal} + c \cdot E_{maxdisp} + d \cdot E_{maxarea}, \quad (1)$$

where,

$$\Delta E_{vdist} = d_{v_d}/\sqrt{\overline{S}} + E_p, \quad (2)$$

$$\Delta E_{atotal} = |(S_Q - S_P)/S_P|, \quad (3)$$

$$E_{maxdisp} = \max_{Q_j}\left(\sum_i \left|\frac{\phi_{ji} - 60}{60}\right|\right), \quad (4)$$

$$E_{maxarea} = \max_{Q_j}(S(Q_j))/\overline{S}, \quad (5)$$

d_{v_d} : distance from v_d to Q_j which is in the direction of $\vec{n_d}$,
$\vec{n_d}$: normal vector of v_d ($\equiv \sum(\theta_i \cdot \vec{n_i})/\sum \theta_i$),
θ_i : vertex angle of v_d of P_i,
$S(P)$: the area of P,
S_Q : total area of $Q_i (\equiv \sum_i S(Q_i))$,
S_P : total area of $P_i (\equiv \sum_i S(P_i))$,
$\overline{S}$: average area of the initial meshes,
E_p : protrusive penalty for ΔE_{vdist},
p : intersection point of Q_j and the $\vec{n_d}$ line,
ϕ_{ji} : ith vertex angle of mesh Q_j,
$a, \cdots, d$: weights for each energy term.

The normal vector of v_d, $\vec{n_d}$, is calculated at one time in the first step of the algorithm and maintained during the simplification because it is important to keep the direction of the normal vector of initial surfaces during simplification. The vector is used for calculating ΔE_{vdist}.

ΔE_{vdist} is the distance from v_d to a generated triangle. It indicates the degree of the collapse of the mesh surface in terms of distance. The protrusive penalty for ΔE_{vdist}, E_p, is added to ΔE_{vdist} when $\vec{n_d}$ doesn't point in the direction of Q_i. E_p is given as $\alpha + \beta - 1$ according to the vector equation, $\vec{v_5 p}$, in Fig. 2. ΔE_{atotal} is the difference of total area. It indicates the degree of collapse of the mesh surface in terms of area. For simple manifold vertices, these two energy terms are related to the modification of the curvature value. For boundary manifold vertices, they are related to the preservation of boundary lines. $E_{maxdisp}$ is the maximum dispersion of the interior angles of generated triangles and is related to the maximum aspect ratio of generated triangles. It prevents the generation of slender triangles. $E_{maxarea}$ is the maximum area of generated triangles. It prevents the generation of large triangles. These two energy terms are related to the geometric shape, i.e. the quality, of generated triangles. Each energy term is divided by a value of the same dimension, such as the average value of the original meshes. Therefore, each energy has no dimension, and the weight of each term is independent of the scale of the original data.

4 ALGORITHM

One of the simplest but fastest ways to simplify the mesh surface is to select a vertex that yields the minimum VDT energy iteratively. However, the value of VDT energy depends on the weight of each energy term, and the weights have to be determined according to the feature of the original data. To avoid this problem, our algorithm selects one vertex which yields the minimum VDT energy whose terms are less than the specified upper limit constraints, and performs the corresponding VDT. The proposed algorithm is outlined in Fig. 3. It has two loops. We call the outer loop the *constrained energy loop* and the inner one the *minimum energy loop*. The *constrained energy loop* sets the upper limit constraints and makes the initial priority queue for the *minimum energy loop*. The *minimum energy loop* deletes vertices iteratively under the constraints.

In the first step, normal vectors of input vertices are calculated as described in the previous section. In the *constrained energy loop*, the user-specified upper limit constraints are updated first. Next, VDT energies of remaining vertices are calculated. Then, vertices having at least one energy term that exceeds the updated limits are discarded, and a vertex priority queue is made in which the vertex with the minimum VDT energy has the highest priority. The VDT energies for one vertex are calculated on the assumption that the vertex is deleted and the resulting hole is re-triangulated. Although there are many combinations of generated triangles for the deletion of one vertex, one VDT for one vertex deletion is determined after all. The problem with this re-triangulation is discussed in the following section. In the *minimum energy loop*, the vertex that has the highest priority is selected, and the corresponding VDT is performed iteratively. After that, the VDT energies of vertices affected by the preceding operation are re-calculated and the priority queue is updated. This loop is iterated until the priority queue is empty or the reduction rate satisfies the user-specified rate.

The VDT energy terms used for the upper limit constraints are ΔE_{vdist} and ΔE_{atotal}. As described in the previous section, these terms are related to the modification of the curvature value and boundary edges. Increasing the upper limit constraints indicates the increasing of the target curvature value of the mesh surface for simplification. In other words, our algorithm first simplifies near-flat surfaces and after that simplifies globally-curved surfaces, which results in preventing the global shape of the objects from collapsing in the early stage of simplification. That is, the proposed algorithm yields good simplified results for a model composed of various-shaped parts. As for the boundary edges, it is important to maintain the lines of them during the simplification

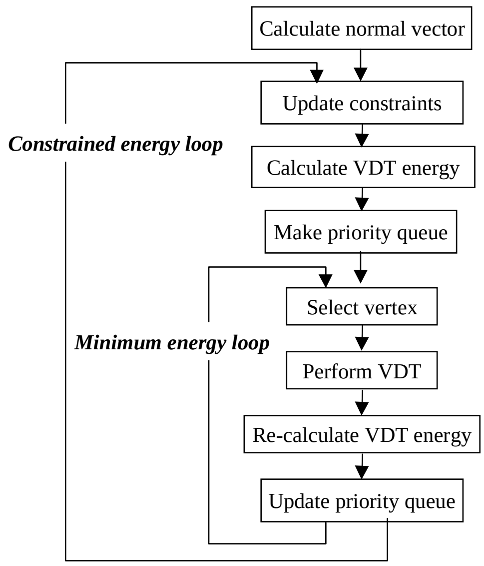


Figure 3: Flow of the proposed algorithm

because they are important feature lines of the original data. The upper limit constraints also have the advantage of preserving boundary lines.

5 TRIANGULATION AND VDT ENERGY

Hole re-triangulation after vertex deletion is one of the main problems for iterative vertex-deletion algorithms. Published algorithms separate vertex deletion and re-triangulation. Klein et al. used Delaunay triangulation [Klein96a]. Schroeder et al. also tried to use a simple criterion for this problem [Schro92a, Schro97b], but it doesn't guarantee an optimal solution. In [Schro92a], the outer loop on the average plane is divided iteratively by a recursive operation in regard to the aspect ratio. In [Schro97b], the shortest edge connected to a vertex to be deleted is deleted first, and remaining edges are re-connected to the other vertex of the deleted edge.

In our algorithm, the VDT energy function is defined so that it takes re-triangulation into account. When calculating VDT energies for one vertex, the VDT energies are calculated for all possible combinations of triangles when the number of the vertices of the outer loop is less than 8. Otherwise, VDT energies are calculated for combinations such that the number of edges connected to each vertex of the outer loop is averaged. Fig. 4 shows two examples of the tested combination of triangles when the number of the outer-loop vertices is 8. Note that the total number of tested combinations is 8, and the tested

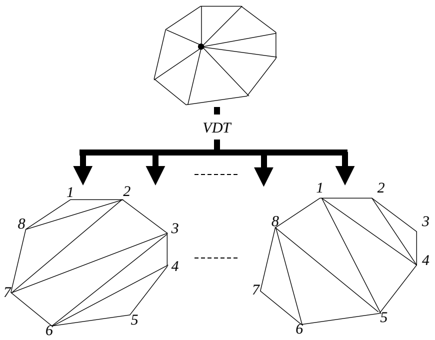


Figure 4: Examples of tested combination of triangles when the number of the outer-loop vertices is 8

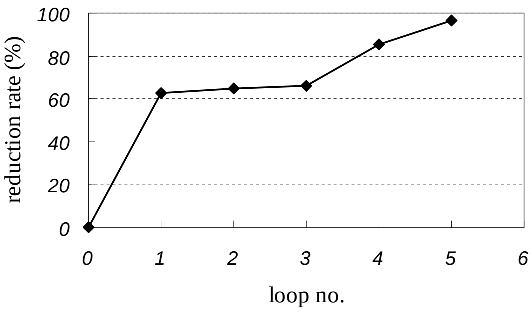


Figure 5: Variation of reduction rate for each *constrained energy loop* of cuboid and sphere model

combinations yield good triangles in terms of aspect ratios. Actually, it is desirable to calculate all the VDT energies for all the combinations of generated triangles for one vertex deletion. However, this is very difficult in reality when the number of the outer-loop vertices is large.

Although it doesn't guarantee an optimal solution like Schroeder's algorithms, our algorithm produces a near-optimal solution because more than 98% of vertices were deleted when the number of outer-loop vertices was less than 8 in our experimental data. (See experimental results.)

After calculating all VDT energies for one vertex deletion, we choose only one combination that yields a minimum VDT energy.

6 EXPERIMENTAL RESULTS

The proposed algorithm was implemented in VC++ on a PC/AT computer (Windows NT Pentium II, 266MHz with 128MBytes of memory).

First, we made a triangulated surface model of a cuboid and a sphere so that they were placed separately side by side, and examined the effect of the upper limit constraints. The produced model consists of 5242 vertices and 9812 triangles. The data is one object, although it was produced from two objects. The results are shown in Fig. 6. In Fig. 6, simplified results with the upper limit constraints are shown in (2), and the results without the upper limit constraints are shown in (3). In each figure, the reduction rate is the ratio of the number of deleted vertices to that of the original vertices. The weights for each energy term were set 0.2 for ΔE_{vdist} of simple manifold vertices, 1 for ΔE_{vdist} of boundary manifold vertices, 5 for ΔE_{atotal}, 5 for $E_{maxdisp}$, and 0.2 for $E_{maxarea}$. The upper limit constraints used in the *constrained energy loop* are shown in Table 1. In the table, "loop no." is the number of *constrained energy loop*'s. They were determined empirically. The results show that flat surfaces of the cuboid are simplified first when the upper limits are constrained, and the sphere's surface is simplified after that. On the other hand, both surfaces are simplified simultaneously when the upper limits are set to infinity. In addition, the optimal solution was produced for the cuboid for the constrained operation; that is, one rectangle composed of two triangles was produced. Figure 5 shows the variation of reduction rate when each *constrained energy loop* is finished. For the broken line in the graph, a high gradient indicates that many vertices were deleted in the loop and low gradient indicates that few vertices were deleted in the loop. The broken line shows clearly the change of target curvature value of surface during simplification.

Next, we tested the algorithm on a lamp model, a washstand model, and a commode model using the constraints data in Table 1. Simplified wire frame models of a lamp are shown in Fig. 7, and simplified flat shaded models of a washstand and a commode are shown in Figs. 8 and 9, respectively. The results show that proposed algorithm yields a good geometric approximation for all models. Especially important is that the outlines of boundary edges of lower part of the commode are maintained during simplification. However, the 90% reduction model of the washstand, around the sink and mirror, is out of shape. Furthermore, We tested our algorithm on more than 20 models using the same set of parameters. All results were good geometric approximations.

We also examined the rate of the number of deleted vertices when the number of the outer-

loop no.	1	2	3	4	5	6	7	8	9
ΔE_{vdist}	0.001	0.005	0.01	0.05	0.1	0.5	1	1	1
ΔE_{atotal}	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.2	-

Table 1: Data of the upper limit constraints for experiments

loop vertices is less than 8. The results are 100% for the cubiod and sphere model, 99.0% for the lamp model, 98.0% for the washstand and the commode model. Consequently, we can say the simplified results are either optimal or near-optimal in terms of our criteria that take into account the triangulation in the VDT energy function.

7 CONCLUSION AND FUTURE WORK

We have developed a mesh-surface-simplification algorithm based on the constrained energy function. The dimension-less VDT energy function takes re-triangulation into account and it represents the cost of vertex deletion and hole triangulation. In the algorithm, a simple or a boundary manifold vertex that yields minimum VDT energy whose terms are less than the specified upper limit constraints is deleted and corresponding VDT is performed iteratively. During simplification, the target curvature value of the surface to be deleted is increased gradually and the outline of boundary edges is preserved by increasing upper limit constraints.

The experimental results show that the proposed algorithm has the following advantages:

1. Re-triangulation is optimized in terms of the VDT energy for almost of the deleted vertices.

2. One set of parameters can be applied to all the various-scale data with a variety of sizes and shapes with boundary edges.

In future, we will extend our algorithm to non-manifold vertices in the same manner as described in [Schro97b]. We will also extend it to treat surface properties such as a color. We believe this is possible because the VDT energy terms are easy to add and an energy term that represents the difference of vertex color during VDT can be defined. In addition, the upper limit constraint of the color energy term is also considerable, and this prevents generating a vertex that has mixed color.

ACKNOWLEDGEMENTS

The authors would like to thank Mr. Nakazato, Mr. Minamiyama, Mr. Niikura and Mr. Yamashita of TOTO technical center for providing experimental data.

REFERENCES

[Hoppe96a] Hugues Hoppe: *Progressive Meshes*, *Proc. of SIGGRAPH '96*, pp. 99–108, 1996.

[Popov97a] Jovan Popovic and Hugues Hoppe: *Progressive Simplical Complexes*, *Proc. of SIGGRAPH '97*, pp. 217–224, 1997.

[Gieng97a] Tran S. Gieng and Bernd Hamann, Kenneth I. Joy, Gregory L. Schussman and Issac J. Trotts: *Smooth Hierarchical Surface Triangulations*, *Proc. of Visualization '97*, pp. 379–386, 1997.

[Garla97a] Michael Garland and Paul S. Heckbert: *Surface Simplification Using Quadric Error Metrics Proc. of SIGGRAPH '97*, pp. 209–216, 1997.

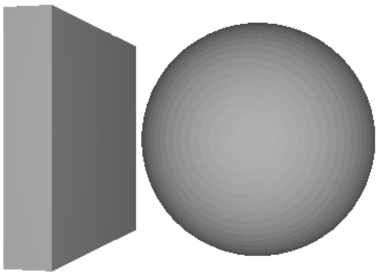
[Schro92a] William J. Schroeder, Jonathan A. Zarge and William E. Lorensen: *Decimation of Triangle Meshes*, *Proc. of SIGGRAPH '92*, pp. 65–70, 1992.

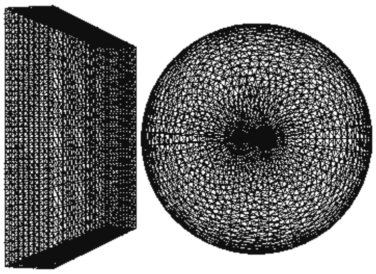
[Schro97b] William J. Schroeder: *A Topology Modifying Progressive Decimation Algorithm*, *Proc. of Visualization '97*, pp.205–212, 1997.

[Klein96a] Reinhard Klein, Gunther Liebich and W.Straßer: *Mesh Reduction with Error Control*, *Proc. of Visualization '96*, pp. 311–318, 1996.

[Heckb97a] Paul S. Heckbert and Michael Garland: *Survey of Polygonal Surface Simplification Algorithms*, *In Multiresolution Surface Modeling Course Notes. SIGGRAPH '97*, 1997.

[Shekh96a] Raj Shekhar, Elias Fayyad, Roni Yagel and J.Fredrick Cornhill: *Octree-Based Decimation of Marching Cubes Surfaces*, *Proc. of Visualization '96*, pp. 335–342, 1996.

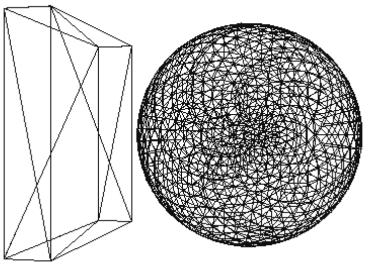


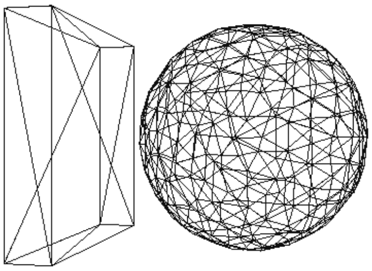


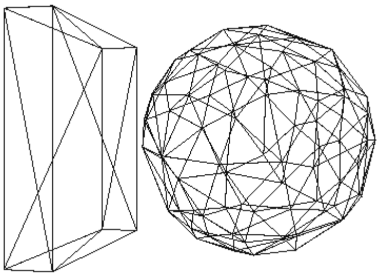
(1a) flat shaded model

(1b) wire frame model

(1) original model





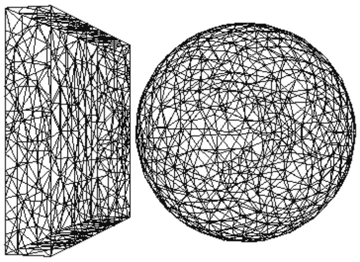


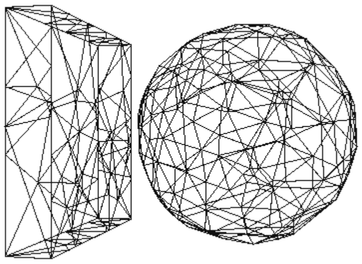
(2a) 80% reduction

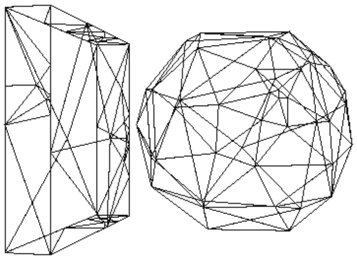
(2b) 95% reduction

(2c) 98%reduction

(2) simplified wire frame model **with** upper limit constraints







(3a) 80% reduction

(3b) 95% reduction

(3c) 98% reduction

(3) simplified wire frame model **without** upper limit constraints

Figure 6: Experimental results for cuboid and sphere model

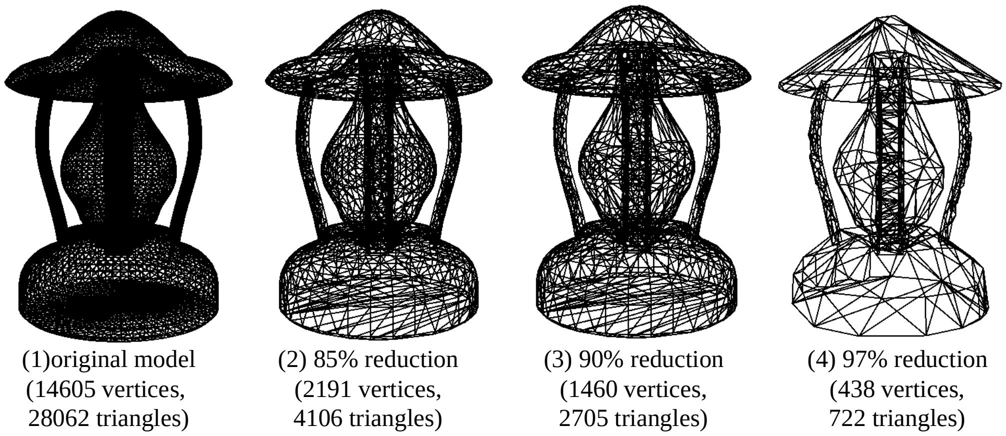


Figure 7: Wire frame model of simplified lamp

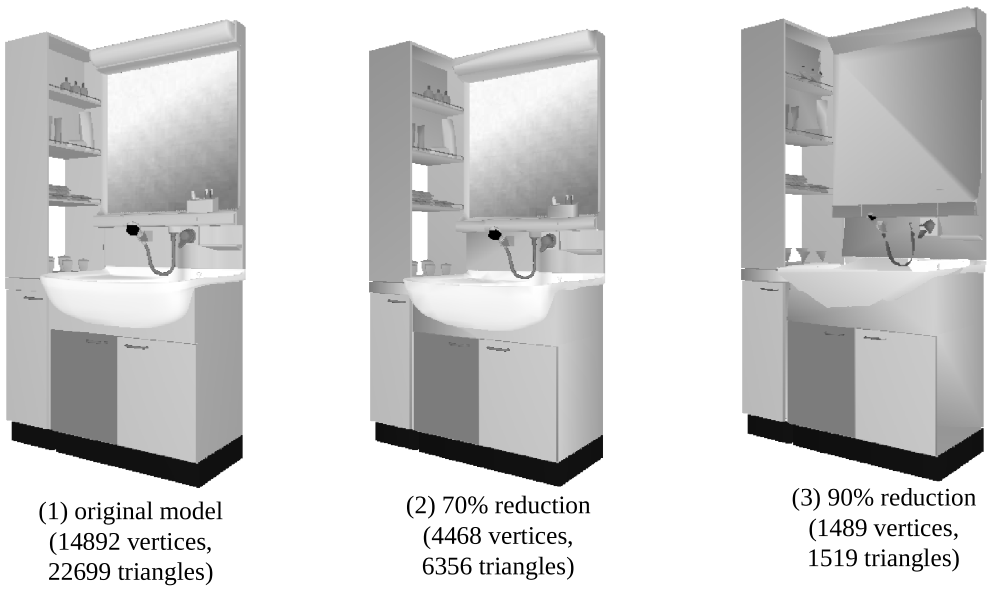


Figure 8: Flat shaded model of simplified washstand

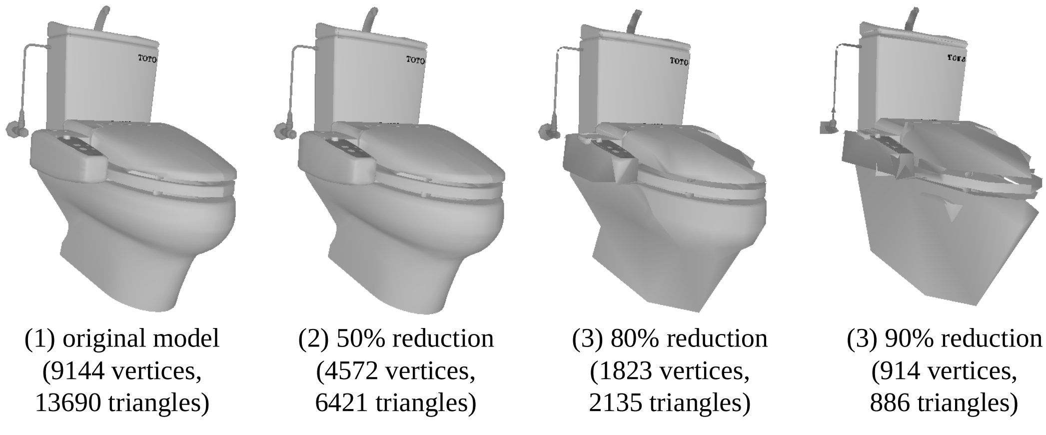


Figure 9: Flat shaded model of simplified commode