

Mesh Simplification using Four-Face Clusters

Luiz Velho

IMPA–Instituto de Matemática Pura e Aplicada,
lvelho@visgraf.impa.br

Abstract

In this paper we introduce a new algorithm for simplification of polygonal meshes. It generates a variable resolution structure called hierarchical 4-K mesh. This structure is a powerful representation for non-uniform level of detail that, among other things, allows simple and efficient extraction of conforming meshes.

Keywords: *variable resolution, 4-k meshes, level of detail, adaptation.*

1 Introduction

The simplification of large polygonal models is an important problem in Computer Graphics. There are, at least, two reasons that make its solution an essential component in graphical applications.

First, geometric models are becoming extremely complex, as the result of advances in CAD technology. Sophisticated 3D input devices can capture real objects at high resolution, producing very fine meshes with millions of faces. These models are highly oversampled, and have to be pre-processed before going into the graphics production pipeline. One of the main tasks in this pre-processing step is the elimination of redundant faces through mesh simplification [10].

Second, detailed geometric models can be quite large, even after redundancy is eliminated. The sheer size of huge polygonal representations can easily overwhelm most graphics programs, rendering impractical their use in applications. The strategy to overcome such limitations is based on multiresolution models, that allow processing geometry at multiple levels of detail. Simplification algorithms constitute an important component in the creation of multiresolution representations.

The relevance of simplification methods motivated intense research in this field. During the past years, many algorithms have been developed [22, 21, 2, 7, 14, 9, 17, 11, 5, 8, 4, 24, 18]. As a whole, they investigate various aspects of the problem, and provide solutions that contemplate differ-

ent practical trade-offs [3]. A shortcoming of most simplification methods is that they require a post-processing step in order to construct a multiresolution structure.

In this paper, we propose a new simplification algorithm that generates a variable resolution data structure well suited to adaptive level of detail operations. We developed this algorithm motivated by our previous work on hierarchical 4-k meshes [23]. Because of the characteristics of the 4-k mesh our solution addresses issues that were not considered in previous algorithms.

2 Background

A mesh simplification algorithm takes as input a polygonal surface description and outputs a simpler mesh representing the same surface. The simplified mesh should have fewer elements than the initial one and, at the same time, approximate the original surface.

There are several ways to specify this problem. Two common alternative criteria are based on mesh size or on geometric error. In the first case, the goal is to produce a mesh of size n , that gives the best geometric approximation of a surface. In the second case, the goal is to produce a mesh with smallest size, that gives an approximation within a tolerance of ϵ of the surface.

It is apparent that this is an optimization problem to obtain a surface approximation. Moreover, computing the optimal solution is NP-hard, since it requires time exponential in the number of vertices of the mesh [1]. For the above reasons, simplification algorithms resort to heuristics that produce sub-optimal solutions.

Most algorithms are based on iterative methods, where a *simplification operator* is repeatedly applied to the mesh according to some *optimality criteria*, until the desired result is obtained.

The simplification operator, usually consists of a local modification of the mesh that removes elements (i.e. vertices, edges and faces), while maintaining topological consistency. The optimality criteria is used to compare the effect of different candidate modifications.

The whole procedure can be summarized as:

while (requirements not satisfied) **do**

- (1) select candidate modifications
- (2) apply simplification operator

Note that, the above simplification algorithm, actually generates a hierarchy of meshes of decreasing size, (M^0, \dots, M^n) , since it starts with an initial mesh M^0 , and at each iteration a simpler mesh M^J is produced.

Simplification algorithms can be classified according to how they implement operations (1) and (2) in the main loop.

The *simplification operator* changes the connectivity and geometry of a region of the mesh. The domain of the operator consists of a submesh that is modified by it. Interior vertices of this submesh are eliminated and the local connectivity is reconstructed. In addition, the geometry of boundary vertices may be adjusted to improve mesh quality.

Some common simplification operators are: vertex decimation, vertex clustering, face merge, and edge collapse [20, 3]. It can be shown that any simplicial complex could be transformed into a simpler one by a sequence of edge swaps and edge collapses [13]. As a consequence, a topology preserving simplification operator can be implemented as the combination of these two basic operations.

In an *edge swap*, an edge shared by two faces is replaced by another edge linking the opposite vertices of the two faces. In an *edge collapse*, two vertices connected by an edge are replaced by a single vertex – the edge and two faces are removed.

In an edge collapse, there is a choice of where to position the new vertex. When the position of the new vertex is restricted to be one of the edge endpoints, the operator is called a *half-edge collapse*.

The decision of *where* the simplification operator should be applied in the mesh is based on an error metric. The metric gives an estimate of the distortion introduced by applying the modification to the mesh.

Several aspects must be considered, such as surface geometry, shape and attributes. Since the simplification operator affects the mesh locally, it makes sense to employ a criterion based on local surface properties. For this purpose concepts from differential geometry, adapted to the discrete setting, can be used to characterize the polygonal surface.

These properties are all defined in a neighborhood $N(p)$ of a vertex p of the mesh. They include, geometric error, distortion, and curvature [15].

The local geometric error, $E(p)$, measures the distance between the original surface and its approximation. Different metrics can be used to estimate this distance. For example, the Hausdorff distance or the mean square distance.

The local distortion, $R(p)$, can be estimated from the aspect ratio of faces.

The local curvature, $C(p)$, can be estimated from the dihedral angle between adjacent faces.

Note that the geometric error is an extrinsic property of the mesh related with surface approximation, while the distortion and curvature are intrinsic properties of the mesh, related with fairness of the surface.

The mesh quality criteria takes into account both approximation and fairness of the simplified surface. These components are combined into an adaptation functional with distinct weights for each of them

$$F(K) = \sum_{p \in K} \alpha E(p) + \beta R(p) + \gamma C(p) \quad (1)$$

There are two basic strategies to apply the simplification operator using the adaptation functional discussed above. They are: sequential and parallel application.

In the *sequential application*, one region of the mesh is selected at each step. In the *parallel application*, a set of independent regions that cover most of the mesh is selected at each step.

The main difference between these two strategies is the type of hierarchical structure that they generate. Sequential application produces a progressive mesh, in which the resolution changes locally, while parallel application produces a multiresolution mesh, in which the resolution changes globally.

3 Variable Resolution 4-k Meshes

A variable resolution structure allows non-uniform level of detail operations with a polygonal surface and it is well suited to adaptive geometric computations [20]. It consists of a directed acyclic graph (DAG) in which nodes represent local modifications to a mesh and arcs give the dependencies between modifications [19].

Level of detail operations fall into three categories: *mesh extraction*, *spatial search*; and *topological queries*.

The usefulness of a variable resolution representation depends on certain properties that guarantee efficient implementation of level of detail operations. These desirable properties are: *high expressive power*; *linear growth*, *bounded width*, and *logarithmic height* [6].

Expressive power is the number of different meshes that can be generated from the structure. This property is important for the adaptivity of all level of detail operations.

Growth rate is the ratio between the number of local modifications and the size of the mesh produced by them. This property is important for mesh extraction operations, such as view dependent mesh adaptation.

Structure height is the maximum length of a path from the top to the bottom of the dependency graph. This property is important for spatial search operations, such as point location.

Structure width is the maximum number of dependency relations of a local modification. This property is important for topological queries, such as neighborhood computation.

The properties of a variable resolution structure are determined by the methods used to construct it.

The $4-k$ mesh is a particular kind of variable resolution structure that exhibits most of the above desirable properties [23].

The power of the $4-k$ mesh comes largely from the fact that is built from two simple local modifications: edge swap; and degree 4 vertex removal. These two modifications form a complete set of topology preserving mesh simplification operators.

The basic block in a $4-k$ mesh is a submesh composed by two adjacent faces sharing an edge. This submesh can be produced either by an edge swap operation that transforms a two-face cluster into another two-face cluster (see Figure 1(a)), or by a degree 4 vertex removal, that transforms a four-face cluster into a two-face cluster (see Figure 1(b)). Figure 1 shows these two operations.

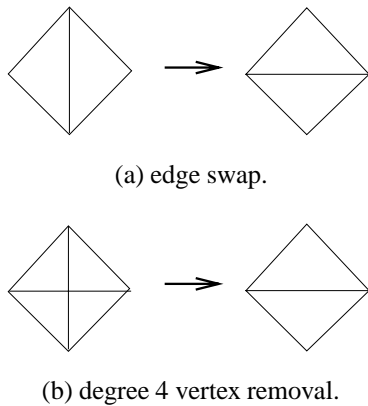


Figure 1. Local modifications of a $4-k$ mesh structure.

Note that the boundary of the submesh is not changed by both operators. This property is essential for building variable resolution structures.

Another important remark is that the degree 4 vertex removal is equivalent to a restricted half edge collapse in which one endpoint of the edge is a vertex of valence 4.

It is also easy to see that an edge collapse can be decomposed into a sequence of edge swaps followed by a degree 4 vertex removal. The purpose of the edge swap operations is to change the 1-neighborhood of the internal vertex of the submesh to have exactly 4 incident edges, so that the degree 4 vertex removal can be applied. Figure 2 shows an example of the decomposition of a general edge collapse into these basic operations.

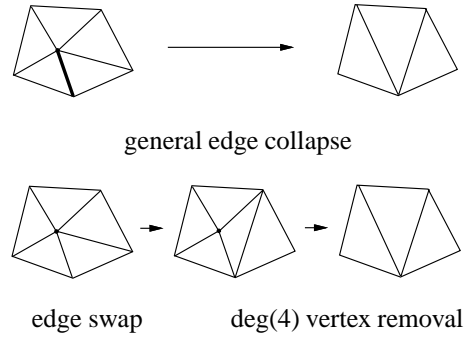


Figure 2. Decomposition of a general edge collapse

The combination of edge swap and degree 4 vertex removal operations is responsible for the bounded width of the $4-k$ mesh structure.

The logarithmic height property is achieved through a parallel application of local modifications such that they cover most of the mesh at each simplification step.

4 Simplification using Four-Face Clusters

The main goal of the four-face cluster simplification algorithm is to produce a variable resolution $4-k$ mesh. In order to accomplish this objective we use a simplification algorithm based on the parallel application of edge swaps and valence 4 vertex removal.

The outline of the method is as follows:

Repeat for N refinement levels:

1. Rank vertices based on mesh quality criteria;
2. Select an independent set of clusters that covers most of the mesh;
3. simplify clusters using edge swaps and degree 4 vertex removals;

Note that, because we are building a variable resolution mesh, the termination criteria is the height of the hierarchical structure, instead of mesh size or surface approximation. Nonetheless, the variable resolution structure can be used for extracting a mesh that fulfills either one of these two requirements.

The first step of the method classifies vertices of the mesh according to a mesh quality criteria that includes approximation error as well as surface fairness.

The error introduced by candidate simplifications is computed using a quadric error metric [8]. This metric gives an efficient way to estimate the geometric error between the original and simplified surfaces.

Below we summarize the main results of Garland and Heckbert concerning the quadric error metric.

A quadric error metric has the form:

$$Q(v) = v^T A v + 2b^T v + c \quad (2)$$

where the quadric q is given by $Q = (A, b, c)$, and A is a 3×3 matrix, b a 3-vector and c a scalar.

Each vertex, v of the mesh is associated with a set of planes p_i , of faces f_i adjacent to v . The sum of squared distances $d(w)$ of an arbitrary point w to this set of planes is

$$d(w) = \sum (p_i^T w)^2 \quad (3)$$

where $p_i = (n_x, n_y, n_z, d)$ represents the support plane of the face f_i . The quadric error is derived by rewriting equation (2) as

$$\begin{aligned} d(w) &= \sum (w^T p_i)(p_i^T w) \\ &= w^T \sum (p_i p_i^T) w \\ &= w^T \left(\sum K_i \right) w \end{aligned}$$

where K is related to the quadric Q

$$K = pp^T = \begin{pmatrix} n_x^2 & n_x n_y & n_x n_z & n_x d \\ n_x n_y & n_y^2 & n_y n_z & n_y d \\ n_x n_z & n_y n_z & n_z^2 & n_z d \\ n_x d & n_y d & n_z d & d^2 \end{pmatrix} = \begin{pmatrix} A & b \\ b & c \end{pmatrix}$$

The main advantage of using this scheme is that the quadric provides a mechanism to keep track of the history of local modifications to the mesh, by accumulating planes of the surface in the neighborhood of vertices of the mesh.

In that way, at each simplification operation, when a vertex is eliminated, its quadric is added to the quadrics associated with its neighbors. The quadric is also used to evaluate the cost of a simplification to the mesh.

The surface fairness is estimated using a measure of triangle compactness and dihedral angle. The triangle compactness measure, t_c is equivalent to the aspect ratio [9], and is given by

$$t_c = \frac{4\sqrt{3}a}{l_1^2 + l_2^2 + l_3^2} \quad (4)$$

where a is the area of a triangle and l_i are the length of its edges.

Taking into account the aspect ratio of triangles helps to select swap operations that improve mesh quality.

Now we explain in detail the three steps of the algorithm. Before entering the loop there is an initialization that assigns a quadric Q_v to every vertex, v , of the mesh. It is

computed as the weighted sum of the fundamental quadrics associated with faces f_i that are adjacent to v

$$Q_v = \sum a_i Q_i \quad (5)$$

where a_i is the area of face f_i , $Q_i = (A_i, b_i, c_i) = (n_i n_i^T, -A_i v, v^T A_i, v A_i v)$, and n_i the unit normal vector to f_i .

The first step computes the cost associated with removing vertices of the mesh. Since a simplification operation is a combination of edge swaps and degree 4 vertex removal, the error $E(v)$, incurred by removing a vertex v of the mesh is computed as the sum of the costs of performing these sequence of operations

$$E(v) = \alpha C(v) + \beta S(v) \quad (6)$$

where $C(v)$ is the cost of removing vertex v and $S(v)$ is the cost of edge swaps necessary to make v a vertex of valence 4. In our implementation $\alpha = 0.75$ and $\beta = 0.25$.

The cost of a vertex removal is defined as

$$C(v) = \min(Q_v + Q_u)(u) \quad (7)$$

where Q_u are the quadrics associated with the set of vertices, $u \in N_1(v)$, in the star of v . Note that this is equivalent to the vertex pair contraction cost used by Garland and Heckbert [8].

The cost $S(v)$ is the sum of the costs of edge swaps in the 1-Neighborhood $N_1(v)$ of the vertex v to make v a vertex of valence 4. The quadric metric is also used to compute the cost of swapping an edge $e = (u, v)$. It is defined as

$$S(u, v) = Q_s(t) \quad (8)$$

where s and t are the opposite vertices to the edge $e = (u, v)$ of the faces sharing e . The reason we employ this measure is because the quadric error $Q_s(t)$ can be interpreted also as the squared volume of the tetrahedron defined by the vertices (u, v, s, t) [16]. This gives an estimate of the error incurred by replacing the edge (u, v) by (s, t) .

We select a sequence of independent edge swaps based on $S(u, v)$ and t_c .

Note that the cost of vertex removal measures the approximation error, while the cost of edge swap also measures the change in surface fairness.

The second step of the method tries to cover most of the mesh with an independent set of four-face clusters. This is accomplished through a cluster marking strategy. Vertices are first sorted according to increasing error $E(v)$ into a priority queue. Then, while the queue is not empty, the vertex v with smallest error is extracted from the queue. If v is not marked, the sequence of edge swaps is performed and the resulting four-face cluster is marked, i.e., the four vertices in the star of v are marked. Note that the vertices taken out

of the star of v are not marked and, therefore, they remain as valid candidates for another simplification operation. (See Figure 3).

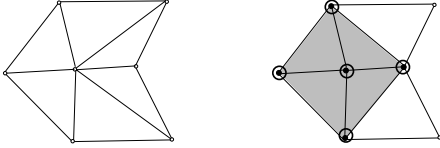


Figure 3. Cluster Marking.

The third step of the method corresponds to actually performing the vertex removal to simplify the four-face cluster selected in step (2).

The simplification operation removes the internal vertex, but there is a choice of how to triangulate the submesh to form the resulting two-face cluster. (See Figure 4).

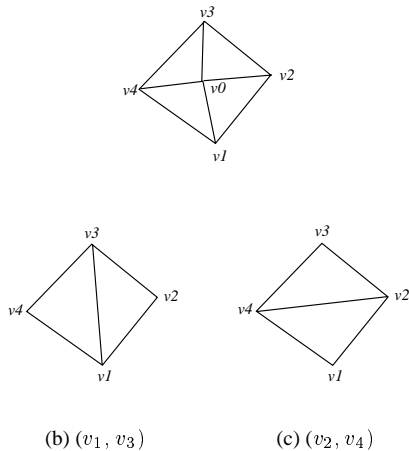


Figure 4. Options for vertex removal.

The option with smaller cost is the one that will be chosen. Hence, the internal edge after simplification is either (v_1, v_3) or (v_2, v_4) , depending of the costs $C(v_i)$, as given by equation 7.

After simplification, the quadric Q_0 , associated with the removed vertex v_0 , is added to the quadrics Q_a and Q_b of the endpoints of the new edge (v_a, v_b)

$$Q_i = (Q_0 + \delta_i Q_i), \quad i = a, b \quad (9)$$

where $\delta_i = 1 - \frac{C_i}{C_a + C_b}$.

The costs of the vertices $w \in N_1(v_a) \cup N_1(v_b)$, that are neighbors of v_a and v_b , need to be recomputed, and their position in the priority queue must be updated.

Steps (2) and (3) are repeated until the priority queue is empty, and the marked four-face clusters cover most of the mesh. The pseudo-code of the whole process is shown in Algorithm 1.

Algorithm 1 : Simplify_4k(M, n)

```

assign quadrics;
for all ( $v \in M$ ) do
  compute  $E(v)$ 
for ( $j = 1$  to  $n$ ) do
  put  $v \in V^j$  into queue
  while (queue not empty) do
    get  $v$  from queue
    if ( $v$  not marked) then
      perform edge swaps in  $N_1(v)$ 
      remove vertex ( $v$ ) and mark cluster
      recompute quadrics  $Q_a$  and  $Q_b$ 
      update queue for  $w \in N_1(v_a) \cup N_1(v_b)$ 

```

We remark that Algorithm 1 can be easily modified to apply the simplification operators sequentially, instead of in parallel.

5 Examples

Now we show some results of applying the four-face cluster simplification algorithm to various models.

The first example is a planar triangulation. Figure 5(a) shows the initial mesh at level 0 containing 186 triangles. Figures 5(b) through (f) show intermediate meshes at levels 1, 3, 5, 7, and 9, containing, respectively, 132, 69, 35, 19, and 6, triangles.

The second example is a height surface. Figure 6(a) shows the initial mesh at level 0 containing 2432 triangles. Figures 6(b) through (f) show intermediate meshes at levels 1, 2, 3, 4, and 5, containing respectively, 1594, 1103, 749, 500, and 338 triangles.

The third example is a cow model. Figure 6(a) shows the initial mesh at level 0 containing 5800 triangles. Figures 6(b) through (f) show intermediate meshes at levels 1, 3, 5, 7, and 9, containing respectively, 1200, 700, 400, 300, 200, triangles.

The last example is the Stanford Bunny. Figure 6(a) shows the initial mesh at level 0 containing 10000 triangles. Figures 6(b) through (f) show intermediate meshes at levels 2, 4, 6, 8, and 10, containing respectively, 4577, 2106, 988, 463, 245 triangles.

These examples show that our algorithm produces simplified meshes of similar quality of the ones generated by Garland and Heckbert. The main difference is that the variable resolution data structure constructed by our algorithm allows the extraction of conforming meshes based on any

adaptation criteria. The hierarchical 4- k representation ensures that the resulting meshes are free of any degeneracy. This is not true for other multiresolution structures, such as vertex hierarchies [24, 12].

Figure 9 shows two meshes extracted from the Bunny model using different adaptation criteria. In Figure 9(a) the criteria was surface curvature, while in Figure 9(b) the criteria was region selection.

6 Conclusions

We presented an algorithm for simplification of polygonal meshes. It is based on simple local operators for mesh modification: edge swap and degree 4 vertex removal. These operators are applied in parallel to an independent set of four-face clusters. The mesh quality criteria employs a quadric error metric and a triangle compactness measure. The method generates a variable resolution 4- k mesh structure.

Future work includes the use of spatial subdivision to handle very large meshes in the implementation of out-of-the-core simplification, and the analysis of local edge smoothness to create tagged meshes.

References

- [1] P. K. Agarwal and S. Suri. Surface approximation and geometric partitions. *SIAM J. Comput.*, 19:1016–1035, 1998.
- [2] M.-E. Algorri and F. Schmitt. Mesh simplification. *Computer Graphics Forum*, 15(3):77–86, August 1996. ISSN 1067-7055.
- [3] P. Cignoni, C. Montani, and R. Scopigno. A comparison of mesh simplification algorithms. *Computers & Graphics*, 22(1):37–54, February 1998. ISSN 0097-8493.
- [4] J. Cohen, M. Olano, and D. Manocha. Appearance-preserving simplification. *Proceedings of SIGGRAPH 98*, pages 115–122, July 1998. ISBN 0-89791-999-8. Held in Orlando, Florida.
- [5] J. Cohen, A. Varshney, D. Manocha, G. Turk, H. Weber, P. Agarwal, J. Frederick P. Brooks, and W. Wright. Simplification envelopes. *Proceedings of SIGGRAPH 96*, pages 119–128, August 1996. ISBN 0-201-94800-1. Held in New Orleans, Louisiana.
- [6] L. D. Floriani, P. Magillo, and E. Puppo. Efficient implementation of multi-triangulations. *IEEE Visualization '98*, pages 43–50, October 1998. ISBN 0-8186-9176-X.
- [7] K. Frank and U. Lang. Data-dependent surface simplification. *Eurographics Workshop on Visualization in Scientific Computing*, April 1998. Held in Blaubeuren, Germany.
- [8] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. *Proceedings of SIGGRAPH 97*, pages 209–216, August 1997. ISBN 0-89791-896-7. Held in Los Angeles, California.
- [9] A. Guéziec. Locally toleranced surface simplification. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):168–189, April - June 1999. ISSN 1077-2626.
- [10] P. Heckbert and M. Garland, editors. *Surface Simplification*, chapter Survey of Polygonal Surface Simplification Algorithms. Course notes of Siggraph 97. ACM SIGGRAPH, July 1997.
- [11] H. Hoppe. Progressive meshes. *Proceedings of SIGGRAPH 96*, pages 99–108, August 1996. ISBN 0-201-94800-1. Held in New Orleans, Louisiana.
- [12] H. Hoppe. View-dependent refinement of progressive meshes. *Proceedings of SIGGRAPH 97*, pages 189–198, August 1997. ISBN 0-89791-896-7. Held in Los Angeles, California.
- [13] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. *Proceedings of SIGGRAPH 93*, pages 19–26, August 1993. ISBN 0-201-58889-7. Held in Anaheim, California.
- [14] A. D. Kalvin and R. H. Taylor. Surfaces: Polygonal mesh simplification with bounded error. *IEEE Computer Graphics & Applications*, 16(3):64–77, May 1996. ISSN 0272-1716.
- [15] L. Kobbelt, S. Campagna, and H.-P. Seidel. A general framework for mesh decimation. *Graphics Interface '98*, pages 43–50, June 1998. ISBN 0-9695338-6-1.
- [16] P. Lindstrom and G. Turk. Evaluation of memoryless simplification. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):98–115, April - June 1999. ISSN 1077-2626.
- [17] P. Lindstrom and G. Turk. Fast and memory efficient polygonal simplification. *IEEE Visualization '98*, pages 279–286, October 1998. ISBN 0-8186-9176-X.
- [18] R. B. Pajarola. Large scale terrain visualization using the restricted quadtree triangulation. *IEEE Visualization '98*, pages 19–26, October 1998. ISBN 0-8186-9176-X.
- [19] E. Puppo. Variable resolution triangulations. Technical report, CNR, 1998.
- [20] E. Puppo and R. Scopigno. Simplification, LOD and multiresolution – principles and applications, 1997. Eurographics'97 Tutorial Notes.
- [21] R. Ronfard and J. Rossignac. Full-range approximation of triangulated polyhedra. *Computer Graphics Forum*, 15(3), Aug. 1996. Proc. Eurographics '96.
- [22] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen. Decimation of triangle meshes. *Computer Graphics (SIGGRAPH '92 Proc.)*, 26(2):65–70, July 1992.
- [23] L. Velho and J. Gomes. Hierarchical 4- k meshes: *Concepts and Applications*. Computer Graphics Forum, 2000.
- [24] J. C. Xia and A. Varshney. Dynamic view-dependent simplification for polygonal models. *IEEE Visualization '96*, pages 327–334, October 1996. ISBN 0-89791-864-9.

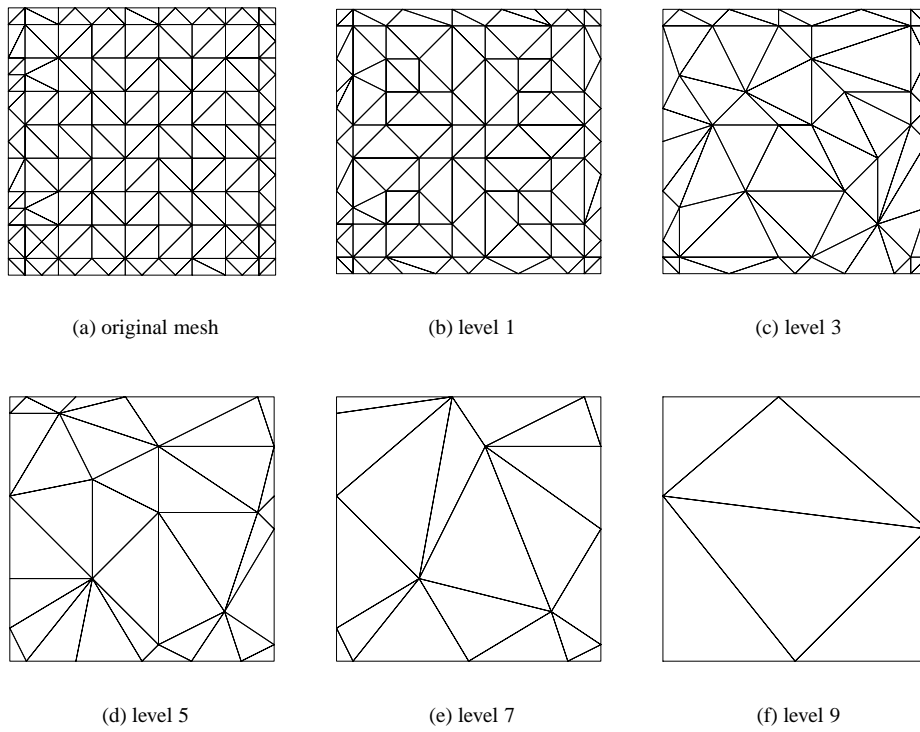


Figure 5. Planar triangulation. Simplified meshes with 186, 132, 69, 35, 19, 6 triangles

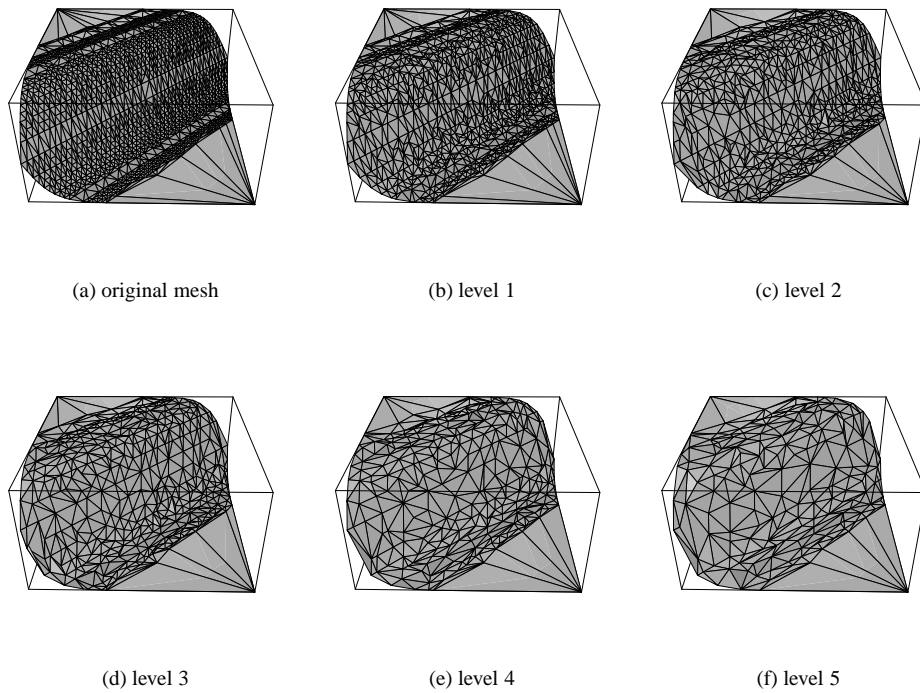


Figure 6. Height surface. Simplified meshes with 2432, 1594, 1103, 749, 500, 338 triangles

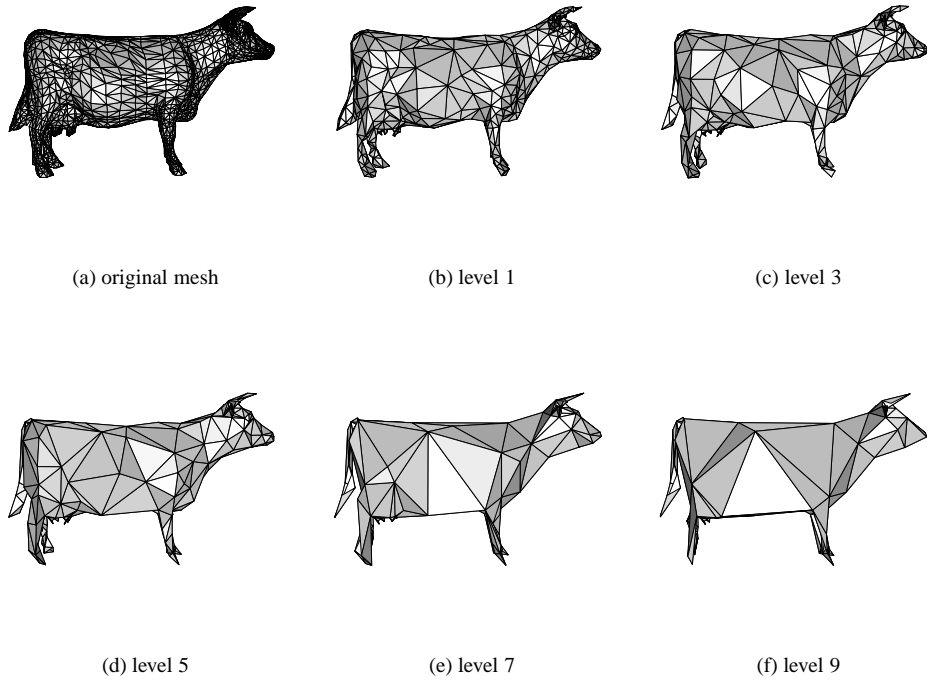


Figure 7. Cow model. Simplified meshes with 5800, 1200, 700, 400, 300, 200 triangles

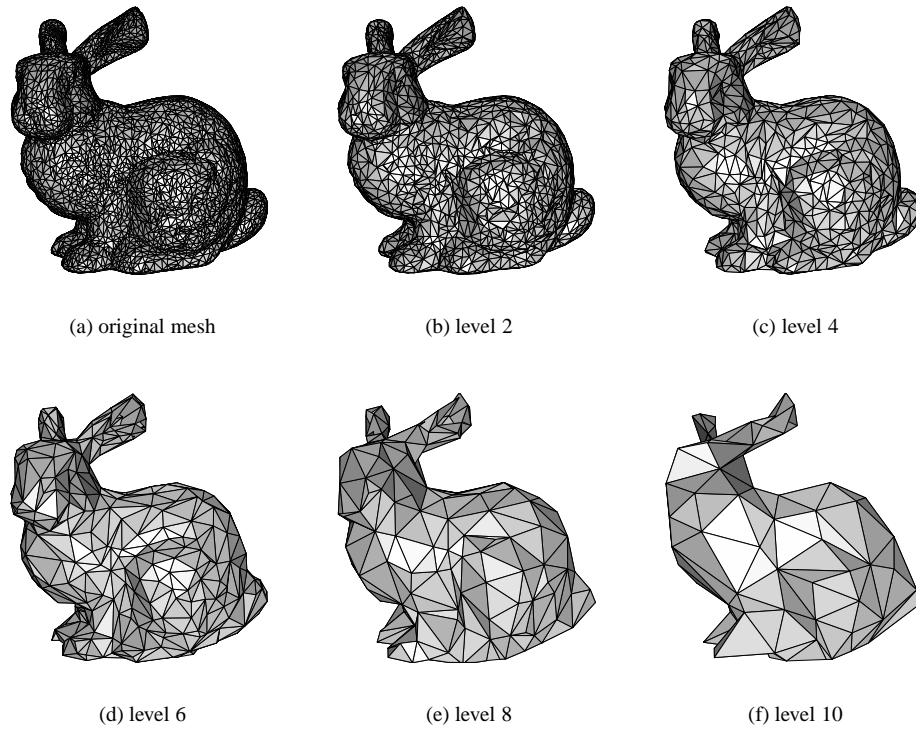
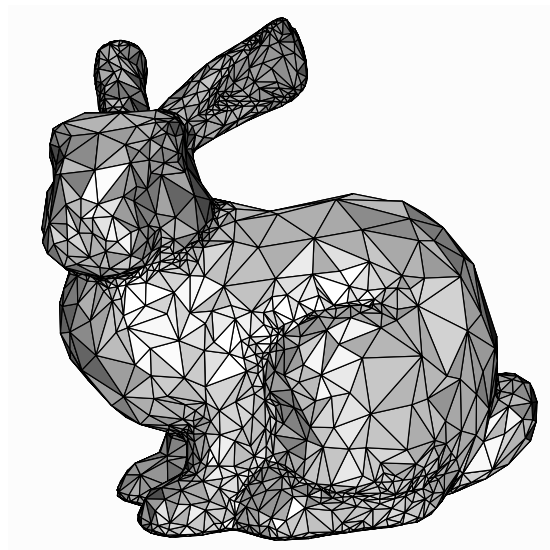
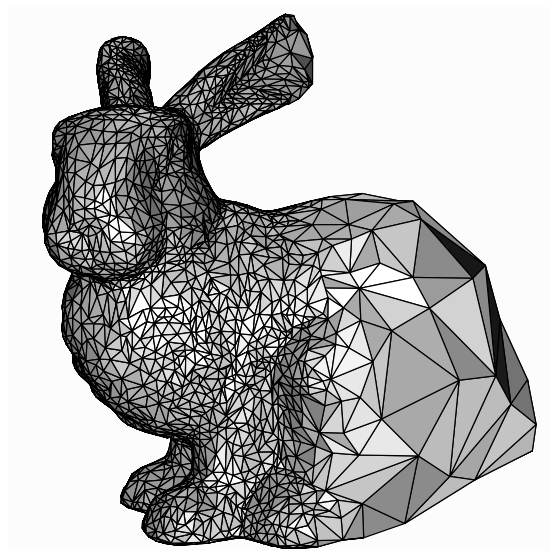


Figure 8. Stanford Bunny. Simplified meshes with 10000, 4577, 2106, 988, 463, 245 triangles



(a)



(b)

Figure 9. Adapted meshes