Tree balancing for mesh simplification

Markus Grabner†

Graz University of Technology, A-8010 Graz, Austria

Abstract

In this paper, a tree balancing technique is presented that reduces the number of hierarchy levels in typical pair-contraction based mesh simplification methods. A well-balanced hierarchy is essential for data structures that explicitly represent vertices as the corresponding paths through the simplification hierarchy.
The quality of the generated meshes is evaluated by three methods. The deviation between the simplified and the original mesh is measured by the Hausdorff distance. The shape of the individual triangles is evaluated by Guéziec's compactness value. Finally, a plot showing the shape distribution of all triangles in the mesh is given. The experiments show that tree balancing has no significant impact on triangle quality.

Categories and Subject Descriptors (according to ACM CCS):
I.3.6 [Computer Graphics]: Graphics data structures and data types

1. Introduction

Due to the ever increasing sizes of 3D models, multiresolution techniques have been a major research interest during the past decade. View-dependent simplification has been identified as a powerful concept for applications requiring interactive walkthroughs of large virtual environments. Given the current viewpoint and viewing direction, the system can determine the subset of the whole scene that contributes most to the visual quality of the current frame. This can significantly reduce the amount of data to be transmitted and displayed, thus increasing the frame rate.

A common way to prepare an arbitrary triangle mesh for interactive visualization is to simplify it iteratively by collapsing pairs of vertices. Vertex pairs introducing the smallest approximation error after contraction are processed first. Then the simplification and refinement steps (see Figure 1) are reorganized to achieve any desired mesh resolution, which doesn't need to be uniform across the surface. A well-known method of this kind is the Progressive Meshes approach[6], which also supports view-dependent operation[7].

The simplification procedure defines an implicit hierarchy of pair contractions since a vertex needs to be present in the mesh before it can be collapsed itself with another vertex.

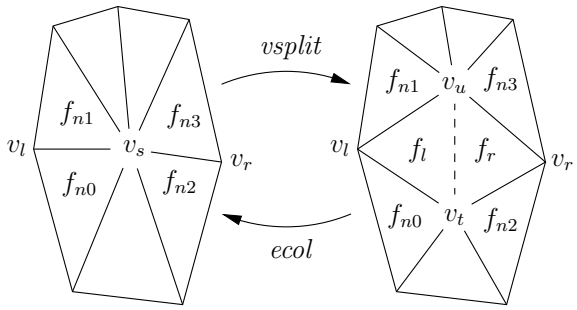


Figure 1: *Basic operations for mesh simplification (edge collapse) and refinement (vertex split) according to Hoppe[7], the more general case of simplification doesn't require the vertices to be connected by an edge (pair contraction, indicated by dashed line)*

This results in a binary tree which is uniquely defined by the (geometrically reasoned) sequence of pair contractions in the initial simplification phase. Methods that don't take into account the structure of this tree may produce a degenerate tree. However, a well-balanced hierarchy is important for at least two reasons:

- To obtain a portion of the mesh in highest detail, a complete path from the hierarchy root to the leaf nodes corre-

† grabner@icg.tu-graz.ac.at

sponding to the requested region must be traversed. In a degenerate tree, the length of this path is only bounded by the total number of vertices in the mesh. This can cause severe delays in transmission of the geometry, especially if the available bandwidth is limited.

- The recently introduced CAME data structure[4] identifies vertices by their paths through the simplification hierarchy, allowing compact storage and transmission of adaptive multiresolution meshes. As in most mesh representations, a fixed number of bits of main memory is assigned to vertex indices (i.e., paths) to avoid dynamic memory management overheads. With 64 bits per vertex index, meshes with up to $2 \cdot 10^{15}$ vertices can be represented[4]. However, this number is dramatically reduced if the hierarchy is not well-balanced. See Section 2.2 for details on this data structure.

In Section 3, we present a method based on a modified error metric that favors the creation of well-balanced trees to overcome the problems mentioned above. Since this can override the geometric decision on which pair to contract next, in Section 4 the quality of the simplified meshes is compared between balanced and unbalanced simplification.

2. Related work

A brief review of the work on mesh simplification is given in Section 2.1. We explain in more detail the CAME data structure[4] in Section 2.2 since it is one of the main applications of the method proposed in this paper. Finally some techniques for triangle quality evaluation are reviewed in Section 2.3.

2.1. Mesh simplification

Many view-dependent refinement schemes require a previous (non view-dependent) simplification procedure. We therefore separately consider the simplification process and the creation of a hierarchical data structure in the following subsections.

2.1.1. Simplification sequence

In contrast to early mesh optimization techniques, where only the final mesh is important[8], more recent algorithms try to achieve optimal quality at each simplification step.

In the Progressive Meshes approach[6], careful attention is paid to preserve mesh appearance during simplification. A sophisticated energy metric is defined, which takes into account the distance between the simplified and the original mesh and attributes such as color and surface creases.

The simplification procedure based on the *quadric error metric*[2] uses a local accumulative error criterion to select the next vertex pair to be contracted. While this method is simple to implement, it also allows topological modifications of the mesh.

2.1.2. Simplification hierarchy

Xia and Varshney propose the *merge tree*[13], which is built bottom-up by collapsing as many independent edges per level as possible. While this avoids excessive degeneration, the trees resulting from this method are rather sparse (see Section 4.2.1 and the original work[13] for examples).

The Progressive Meshes sequence defines an implicit hierarchy of simplification steps as demonstrated in [7]. Due to the less constrained simplification procedure, hierarchies are created with fewer levels than the merge trees[13].

To prevent mesh foldovers, additional dependencies have to be introduced. These can be stored explicitly[7], implicitly[1], or be incorporated into the hierarchy, forming a directed acyclic graph (DAG) instead of a binary tree[3]. Although the tree balancing method presented in this paper doesn't deal with those dependencies, it is still possible (and highly recommended) to incorporate them into the data structure used to represent the hierarchy.

Note that methods for regular meshes (such as Lindstrom's[9]) allow perfectly balanced hierarchies by definition. However, such methods are restricted to heightfields (e.g., digital terrain models).

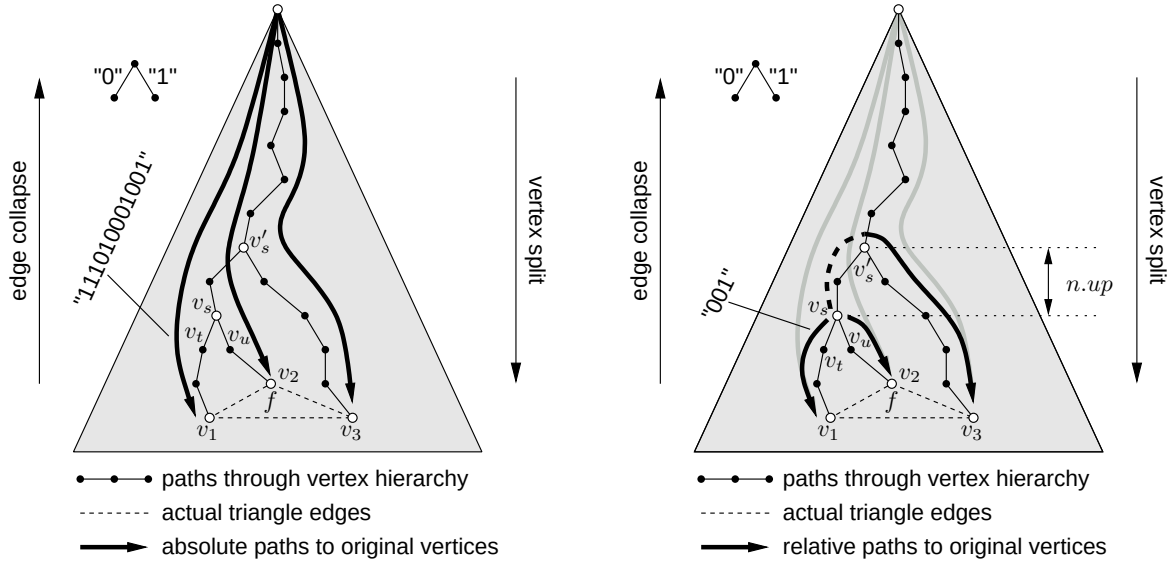
2.2. Compressed adaptive multiresolution encoding

In the CAME data structure[4], mesh vertices are referenced such that triangle adjacency relations are maintained implicitly. In fact, neither at the decompression stage nor at the rendering stage it is required to store information about a triangle's neighbors, allowing efficient storage and rendering.

In contrast to assigning more or less arbitrary indices to mesh vertices[6] or having vertex indices reflect simplification order[1], in CAME each vertex is identified by the path to be taken in the simplification hierarchy from the root to the corresponding node. These *node identificators* are simply bit strings with "0" for the left branch (v_t) and "1" for the right branch (v_u) as indicated in Figure 2(a).

However, the bit strings identifying the paths to the vertices v_1, v_2, and v_3 are highly redundant since they all have a common prefix. It is therefore sufficient to store only those portions of the strings *relative* to the node containing vertex v_s in Figure 2(b). This gives the *relative node identificators* n_1, n_2, and n_3, where n_3 also has to include the number of levels to go up in the hierarchy (dashed line in Figure 2(b)) to reach the common junction v_s' of all vertex paths of triangle f. Note that mesh simplification and encoding are interleaved, i.e. the triangle f is removed from the mesh when the edge between v_t and v_u (the ancestors of v_1 and v_2, respectively) is collapsed. At the same time, n_1, n_2, and n_3 are determined and stored.

Updating triangle vertices is completely separated from hierarchy traversal and doesn't require any additional information. The bit strings n_1, n_2, and n_3 are stored together with



(a) absolut (b) relative

Figure 2: *Triangle f and the hierarchy paths to its vertices v_1, v_2, and v_3*

the *vertex split record*[6] of v_s. Therefore any information required to reconstruct triangle f is known as soon as v_s is split (see Figure 2). By evaluating the relative paths n_i, each triangle can autonomously update its vertices. Moving up and down one level in the hierarchy simply corresponds to truncating and appending one bit, respectively, at the end of the bit string identifying a triangle's vertex in the current level of detail.

Although the relative node identificators *n* are kept on disk with a variable number of bits, they are stored in main memory using fixed-sized variables to avoid dynamic memory management overheads. Each *n* is packed into a 64 bit integer, using six bits for *n.up* (see Figure 2), six bits for the length of the bit string, and the remaining 52 bits for the bit string itself. Assuming a well-balanced hierarchy, this is sufficient to encode meshes with up to $2^{51} \approx 2 \cdot 10^{15}$ vertices.

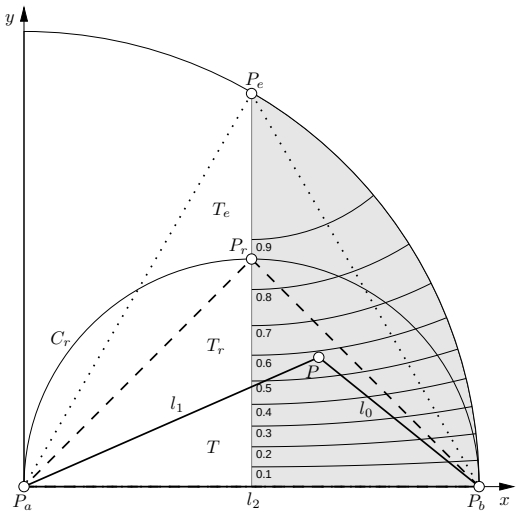


Figure 3: *Triangle shape plot*

2.3. Triangle quality

To achieve optimal image quality when rendering triangle meshes, "well-shaped" (i.e., near equilateral) triangles are desirable. The quality of a triangle with area a and side lengths l_0, l_1, and l_2 is commonly measured by its *compactness*

$$c = \frac{4\sqrt{3}a}{l_0^2 + l_1^2 + l_2^2}$$

as defined by Guéziec[5]. The compactness value lies between zero for a flat triangle (at least one inner angle is zero) and one for an equilateral triangle (all inner angles are $\frac{\pi}{3}$).

A method for visualizing the triangle shape distribution of the whole mesh has been proposed by Niepel et al.[10]. First, each triangle's sides are reordered such that $l_0 \leq l_1 \leq l_2$.

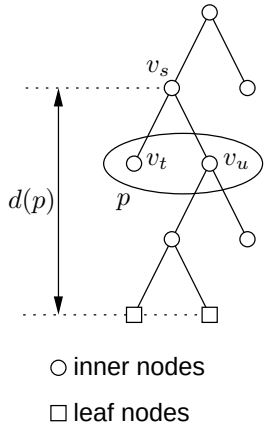


Figure 4: *Vertex pair p with associated subtree depth $d(p)$, leaf nodes are vertices of input mesh, inner nodes are vertices of simplified meshes, v_s, v_t, and v_u refer to Figure 1*

Then the triangle is scaled by $\frac{1}{l_2}$ and rotated such that the longest edge (which obviously has unit length after scaling) coincides with the unit vector along the *x*-axis (see triangle $T = (P_a, P_b, P)$ in Figure 3). After these transformations, the point P lies within the shaded area for each possible triangle T. By applying this method to all triangles in the mesh, we get an impression of the triangle shape distribution of the mesh. Throughout the rest of this paper, only the part corresponding to the shaded area in Figure 3 is shown when presenting triangle shape plots (see Figure 6 for examples).

Figure 3 also shows two special types of triangles: the equilateral triangle T_e with point $P_e = (\frac{1}{2}, \frac{\sqrt{3}}{2})^T$, and a right triangle T_r with point $P_r = (\frac{1}{2}, \frac{1}{2})^T$. Note also the circle C_r which gives the locations of all possible right triangles (theorem of Thales). Finally, we show how Guéziec's compactness value c is related to the shape plot diagram by a set of curves for $c = 0.1 \ldots 0.9$.

3. Tree balancing

3.1. Modified error metric

During simplification, we want to minimize at each step the newly introduced approximation error. This is achieved by storing the error metric values of all possible candidate simplification steps in a priority queue. The step corresponding to the head of the queue is performed, and the procedure is repeated until the queue is empty or any other user-defined stop criterion is fulfilled. This strategy is found in many simplification algorithms[6, 2, 1].

Since the value of the error metric guides the simplification process, we can easily incorporate other criteria by redefining the error metric. The main goal of this work is to avoid degenerate hierarchies, so we define an error metric that favors well-balanced hierarchies. Let $E(p)$ be the value of the error metric associated with contracting vertex pair p. We then define the modified error metric

$$E'(p) = 2^{bd(p)}\,(E(p) + w||p||), \qquad (1)$$

which contains the previous metric $E(p)$ and the Euclidean distance $||p||$ of the two vertices involved. A factor is included that penalizes deep hierarchies. The depth of the hierarchy subtree below the vertex v_s created after contraction of p is called $d(p)$ (see Figure 4), the value b is selected by the user. If contracting p would create a deep branch in the hierarchy, the term $2^{bd(p)}$ becomes large and increases $E'(p)$. This in turn prevents p from being placed at the top of the priority queue, allowing other pairs to be processed first to create a well-balanced hierarchy. The exponential penalty factor "overrides" geometric decisions for extremely ill-balanced meshes, even at the cost of some quality degradation of the resulting mesh (see also Section 4.2.2).

The pair distance $||p||$, weighted by another used-defined parameter w, is included for the following reason. Several mesh generation techniques (e.g., the marching cubes algorithm) create triangles lying in the same plane. If $E(p)$ is chosen to approximate the geometric deviation between the simplified and the original mesh (e.g., the quadric error metric[2]), then each pair contraction within the flat region would be assigned zero error. Therefore no reasonable sequence of pair contractions could be found, not even after applying the tree balancing term. Since we assume that the input mesh doesn't contain degenerate triangles, $||p|| > 0$ holds for all vertex pairs, allowing the tree balancing mechanism to work also in flat regions of the mesh†. Note, however, that we could choose $E(p) = ||p||$ (which is done by Xia and Varshney[13]), which would make the extra term $w||p||$ in Equation 1 superfluous.

For the experiments in Section 4 we use the parameter values $b = 1$ and $w = 0.01$ unless otherwise noted.

3.2. Optimally balanced hierarchies

To evaluate the effectiveness of our tree balancing scheme, we have to identify the optimal solution and compare it with our results. We make use of the following properties:

- The maximum number of nodes in a binary tree of depth d (i.e., $d+1$ levels including root and leaves) is $2^{d+1} - 1$.
- Since each inner node in the simplification hierarchy tree has exactly two children, the number of inner nodes equals the number of leaf nodes minus one. Moreover, the number of leaf nodes is the number V of vertices of the input mesh. Therefore the total number of nodes is $2V - 1$.

† For the purpose of tree balancing, $||p||$ could also be replaced by a global constant, but this would make it difficult to assign proper weights to the balancing term.

We define the *tree fill ratio*

$$r = \frac{2V-1}{2^{d+1}-1}, \tag{2}$$

which equals one for a perfectly balanced binary tree (i.e., a tree of depth d with 2^d leaf nodes) and is otherwise less than one. When applied to the CAME framework (see Section 2.2), it is very important to keep the tree fill ratio close to one, otherwise even meshes with low primitive count could exceed the fixed-length node identificators.

Requiring $r \leq 1$ in Equation 2 gives $d \geq \log_2 V$. Therefore the minimal depth of the simplification hierarchy required for an input mesh with V vertices is

$$d_{\min} = \lceil \log_2 V \rceil .$$

The experiments in Section 4.2.1 show the hierarchy depth and the tree fill ratio r for different models.

4. Results

The six data sets in Figure 5 were used in our experiments. Figure 5(a) is a digital city model where each building is represented as a protrusion of its ground polygon. The model in Figure 5(c) shows an antique building that has been created manually from reconstruction drawings. Figure 5(e) shows a digital terrain model at low resolution, Figure 5(f) is a flat square with a hole.

4.1. Input data evaluation

Before we examine the properties of mesh simplification algorithms applied to the meshes of Figure 5, we determine in Figure 6 their triangle shapes prior to simplification. Most notably, there are regular patterns in Figures 6(a) to 6(c). The circular arcs in Figures 6(a) and 6(c) are due to the fact that most of the faces in the scene are rectangles which are divided into two right triangles. Quantization in the recording process is the reason for the pattern in Figure 6(b).

Figure 5(d) is an irregular triangle mesh, therefore its shape plot (Figure 6(d)) doesn't show any regularity as well. In Figure 5(e), the number of different triangle shapes is reduced due to quantization, therefore Figure 6(e) is sparse. Finally, there is only one single triangle shape in Figure 5(f), resulting in a single dot in Figure 6(f).

4.2. Effects of tree balancing

4.2.1. Hierarchy depth

Now we examine the influence of the balancing parameter b to the hierarchy depth. Table 1 and Figure 7 show the tree depths depending on b for different models. Obviously the unbalanced algorithm runs into great troubles in the more regular cases, creating a tree with 217 levels (Figure 7(d)) for the flat model (Figure 5(f)). Similar results are obtained for the terrain model, where 105 levels are generated by the unbalanced algorithm (Figure 7(c)).

The problems are less severe for the other four models, where the trees achieved by unbalanced simplification are approximately one third deeper than the optimal solution (see also Figures 7(a) and 7(b)). However, tree balancing reduces the depth close to the theoretical minimum (see Section 3.2).

Table 2 shows the hierarchy depths for the bunny model simplified by different algorithms. The deep hierarchy produced by the merge tree approach[13] is due to the restrictive definition of legal edge collapse transformations (edges with overlapping regions of influence must not be collapsed at the same level). A different (less restrictive) method to avoid mesh folding is proposed by Hoppe[7] and used in the CAME framework[4] with some modifications[3], making hierarchies with significantly fewer levels possible.

4.2.2. Mesh quality

Figures 8 and 9 explain how tree balancing affects mesh quality in terms of triangle shapes. During simplification of the city and bunny model, snapshots of the simplified mesh were taken. For each of these meshes, triangle quality was computed by Guéziec's method, and the distributions are shown in the histograms in Figures 8(a) to 8(d) (x-axis is lower bound of compactness value interval, y-axis is normalized number of triangles within this interval). The triangle shape plots in Figures 9(a) to 9(d) and 9(e) to 9(h) refer to the simplified city model at 3000 faces and the simplified bunny model at 1000 faces, respectively, for different values of the balancing parameter b.

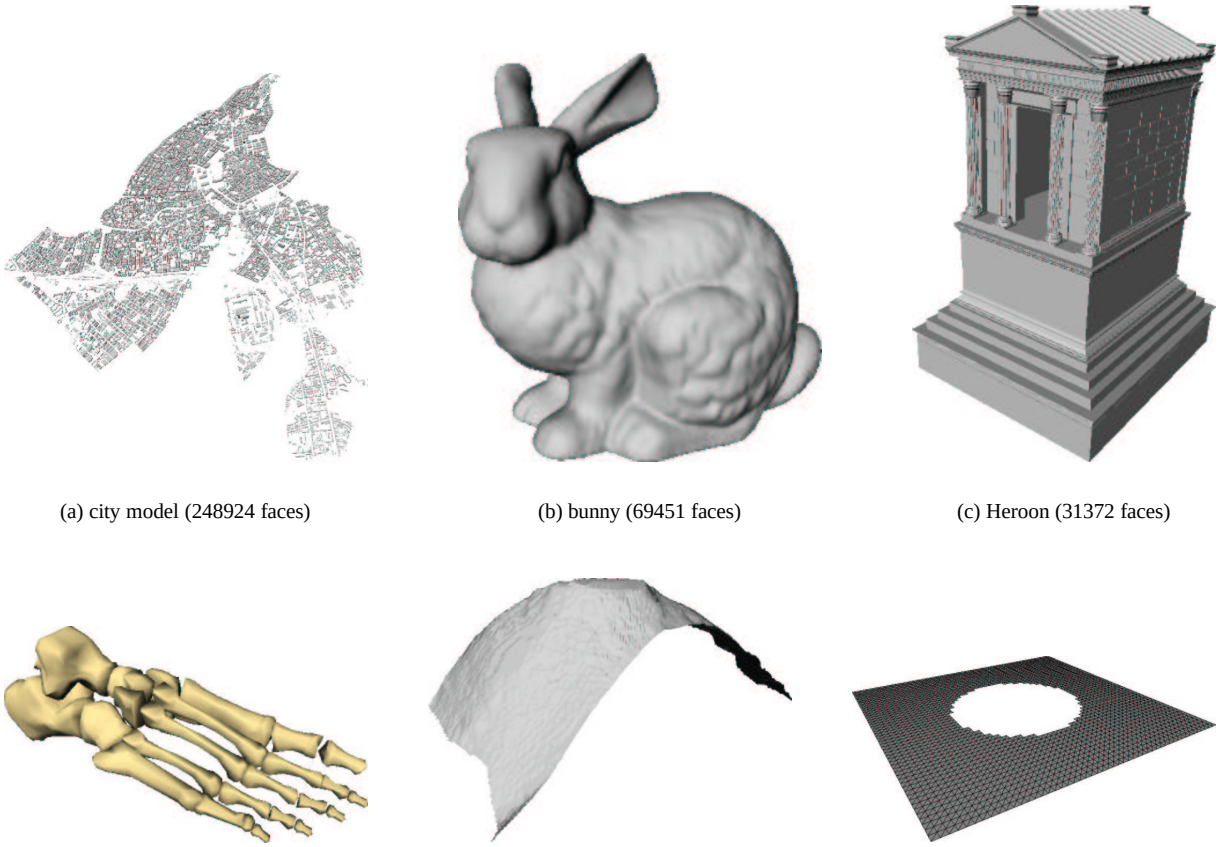
The Figures indicate that the impact of tree balancing on triangle shapes is neglectible for $b \leq 1$. At the same time, $b = 1$ is sufficient to reduce the tree depth close to the limit, as was demonstrated in Figure 7.

4.2.3. Approximation error

In addition to the triangle quality assessment of Section 4.2.2, we calculate the Hausdorff distance for different values of b applied during simplification. We consider a mesh with quite uniform resolution (the bunny) and one with largely varying resolution across its surface (the city model). This property is illustrated in Figure 10 by the initial error metric histograms of these two meshes.

The diagrams in Figure 11 show the Hausdorff distance between the simplified and the original mesh as mesh simplification proceeds[‡]. Note that even for the city model the balanced meshes are very close to the unbalanced one for a face number of 10^3 (0.4% of the original size) or above.

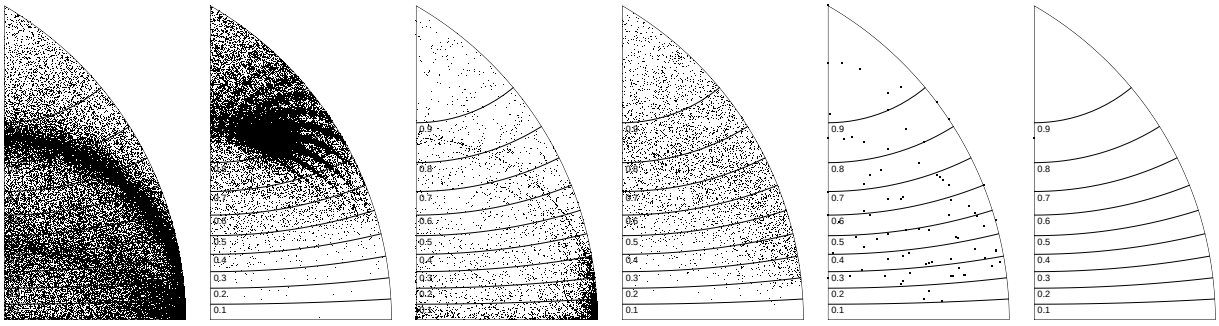
[‡] Note that simplification goes from right to left in this case.



(a) city model (248924 faces) (b) bunny (69451 faces) (c) Heroon (31372 faces)

(d) bones (4204 faces) (e) terrain (23955 faces) (f) flat surface (4022 faces)

Figure 5: *Models used in this paper*

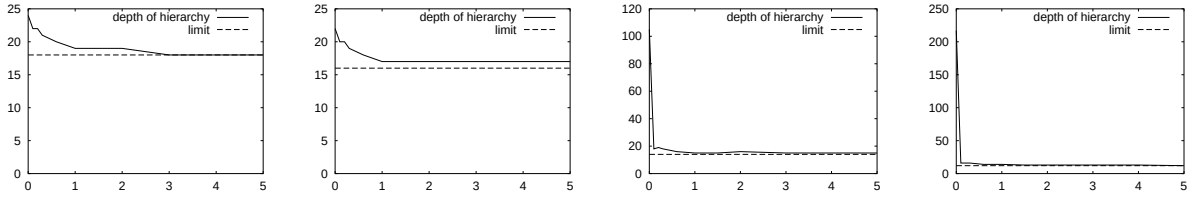


(a) city model (b) bunny (c) Heroon (d) bones (e) terrain (f) flat surface

Figure 6: *Triangle shape plots of the models in Figure 5*

model	number of vertices	depth of hierarchy			tree fill ratio			minimal depth
		unbalanced	$b=1$	$b=3$	unbalanced	$b=1$	$b=3$	
bones	2154	16	13	12	0.03	0.26	0.53	12
flat surface	2161	217	14	13	10^{-62}	0.13	0.26	12
terrain	12246	105	15	15	$3 \cdot 10^{-28}$	0.37	0.37	14
Heroon	15667	27	19	17	$1.1 \cdot 10^{-4}$	0.03	0.12	14
bunny	34834	22	17	17	$8.3 \cdot 10^{-3}$	0.27	0.27	16
city	140245	24	19	18	$8.4 \cdot 10^{-3}$	0.27	0.53	18

Table 1: *Depth of hierarchy and tree fill ratio for the models in Figure 5*

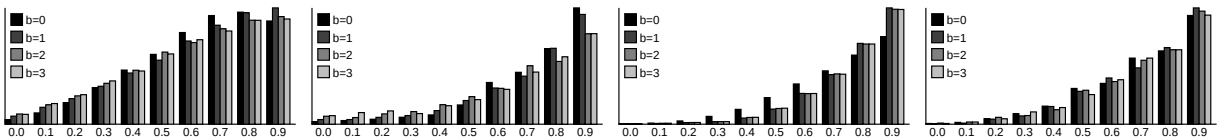


(a) city model (b) bunny (c) terrain (d) flat surface

Figure 7: *Depth of hierarchy for different models and varying balancing factor b*

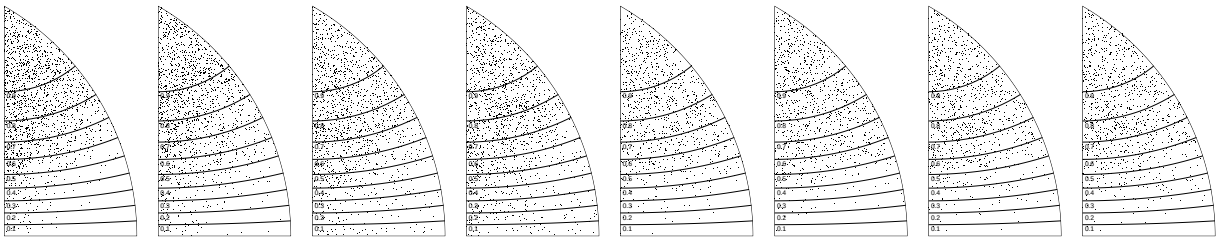
simplification algorithm	depth of hierarchy	tree fill ratio
Dynamic view-dependent simplification[13]	64	$1.9 \cdot 10^{-15}$
View-dependent progressive meshes[7]	23	0.0041
our method ($b=1$)	17	0.27

Table 2: *Comparison of the depth of hierarchy for the bunny model simplified by different algorithms*



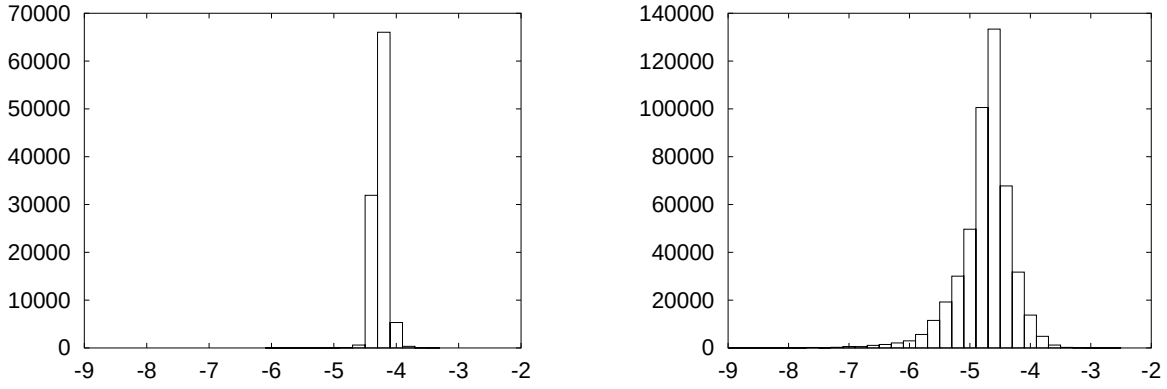
(a) city model (simplified to 30000 faces) (b) city model (simplified to 3000 faces) (c) bunny model (simplified to 10000 faces) (d) bunny model (simplified to 1000 faces)

Figure 8: *Mesh quality (compactness value histograms)*



(a) $b = 0$ (b) $b = 1$ (c) $b = 2$ (d) $b = 3$ (e) $b = 0$ (f) $b = 1$ (g) $b = 2$ (h) $b = 3$

Figure 9: *Mesh quality (triangle shape plots)*



(a) bunny (b) city model

Figure 10: *Initial error metric histograms (x-axis is decadic logarithm of approximation error as defined by Garland and Heckbert*[2]*, y-axis is number of edges in the original mesh within the corresponding approximation error interval)*

Figure 11 also confirms the observation of Section 4.2.2 that $b = 1$ is a good choice to keep the impact of mesh balancing on geometry low.

5. Conclusions and future work

A modified error metric has been presented that can be used as a replacement of common distance-based metrics in traditional priority queue guided mesh simplification algorithms. The effects of the new metric have been studied with respect to hierarchy depth and mesh quality. It has been shown that tree degeneration, which occurred with unbalanced methods, can be avoided. Moreover, the hierarchies created by our method are close to the optimal solution.

Mesh quality is evaluated by Gueziec's compactness value, a normalized shape plot, and by Hausdorff distance. Our experiments show that the quality of the produced meshes is not significantly degraded, although the modified error metric can override geometric decisions during simplification.

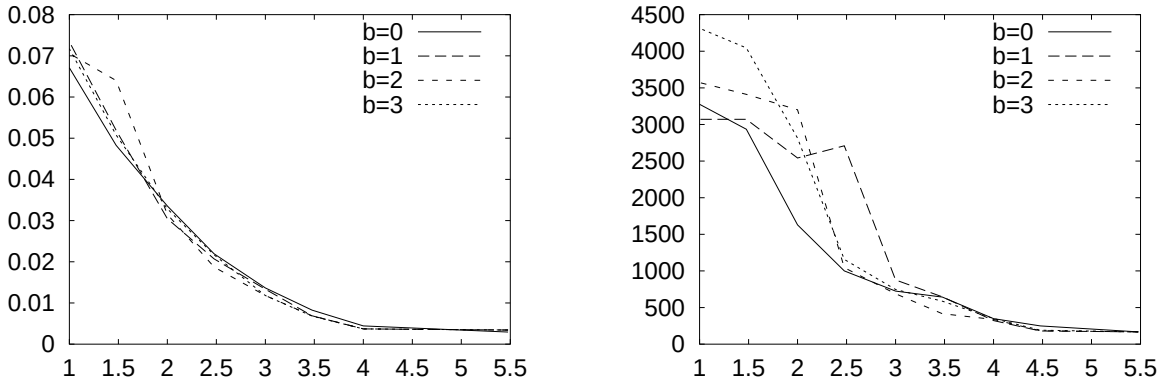
The tree balancing method has been evaluated only during creation of the multiresolution data structure. It would also be interesting to examine its consequences in the much more complex case of adaptive (view-dependent) simplification.

6. Acknowledgments

Many thanks to Andrej Ferko for providing input on mesh quality evaluation. This work has in part been funded by the European Union under contract no. IST-1999-20273.

References

1. Jihad El-Sana and Amitabh Varshney. Generalized view-dependent simplification. In Pere Brunet and Roberto Scopigno, editors, *Proceedings Eurographics*,



(a) bunny

(b) city model

Figure 11: *Hausdorff distance between original and simplified mesh (x-axis is decadic logarithm of number of faces during simplification, y-axis is Hausdorff-distance between the simplified and the original mesh)*

volume 18 of *Computer Graphics Forum*, pages 83–94. Blackwell Publishers, September 1999. ISSN 1067-7055.

2. Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In Whitted [12], pages 209–216. ISBN 0-89791-896-7.
3. Markus Grabner. Consistency of the VDPM framework. In Bianca Falcidieno, editor, *Proceedings of SCCG 2000*, pages 147–155. Comenius University, Bratislava, May 2000. ISBN 80-223-1486-2.
4. Markus Grabner. Compressed adaptive multiresolution encoding. *Journal of WSCG*, 10(1):195–202, February 2002. ISSN 1213-6972.
5. André Guéziec. Surface simplification with variable tolerance. In *Second Annual International Symposium on Medical Robotics and Computer Assisted Surgery (MRCAS '95)*, pages 132–139, November 1995.
6. Hugues Hoppe. Progressive meshes. In Rushmeier [11], pages 99–108. ISSN 0097-8930.
7. Hugues Hoppe. View-dependent refinement of progressive meshes. In Whitted [12], pages 189–198. ISBN 0-89791-896-7.
8. Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 19–26, August 1993. ISSN 0097-8930.
9. Peter Lindstrom, David Koller, William Ribarsky, Larry F. Hughes, Nick Faust, and Gregory Turner. Real-Time, continuous level of detail rendering of height fields. In Rushmeier [11], pages 109–118. ISSN 0097-8930.
10. Ludovít Niepel, Andrej Ferko, and Pavol Cibulka. Computing minimum weight triangulation for smaller point sets. In *60th Anniversary Jubilee Conference*, pages 95–102, Bratislava: Stavebná fakulta STU, 1998. ISBN 80-227-1135-7.
11. Holly Rushmeier, editor. *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series. ACM SIGGRAPH, Addison Wesley, August 1996. ISSN 0097-8930.
12. Turner Whitted, editor. *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series. ACM SIGGRAPH, Addison Wesley, August 1997. ISBN 0-89791-896-7.
13. Julie C. Xia and Amitabh Varshney. Dynamic view-dependent simplification for polygonal models. In *IEEE Visualization '96*. IEEE, October 1996. ISBN 0-89791-864-9.