New Quadric Metric for Simplifying Meshes with Appearance Attributes

Hugues Hoppe
Microsoft Research

ABSTRACT

Complex triangle meshes arise naturally in many areas of computer graphics and visualization. Previous work has shown that a quadric error metric allows fast and accurate geometric simplification of meshes. This quadric approach was recently generalized to handle meshes with appearance attributes. In this paper we present an improved quadric error metric for simplifying meshes with attributes. The new metric, based on geometric correspondence in 3D, requires less storage, evaluates more quickly, and results in more accurate simplified meshes.

Meshes often have attribute discontinuities, such as surface creases and material boundaries, which require multiple attribute vectors per vertex. We show that a wedge-based mesh data structure captures such discontinuities efficiently and permits simultaneous optimization of these multiple attribute vectors. In addition to the new quadric metric, we experiment with two techniques proposed in geometric simplification, memoryless simplification and volume preservation, and show that both of these are beneficial within the quadric framework. The new scheme is demonstrated on a variety of meshes with colors and normals.

Additional Keywords: level of detail, mesh decimation, multiresolution.

1 INTRODUCTION

Complex triangle meshes occur extensively in computer graphics as the result of geometric modeling operations, global illumination simulations, and reconstructions from 3D scans. Because such meshes are difficult to store, transmit, and render, several techniques have been developed for geometrically simplifying them (e.g. [1, 4, 6, 8, 9, 10, 13, 17]). However, meshes often have associated appearance attributes at their vertices, such as normals, colors, and texture coordinates, and relatively few techniques account for these attributes during simplification [1, 2, 5, 7, 10] as reviewed in Section 2. This paper presents a new technique for efficiently and accurately simplifying meshes with attribute data.

Among mesh simplification metrics, the quadric error metric introduced by Garland and Heckbert [6] holds much promise because it is both fast and reasonably accurate. Their more recent work [7] generalizes this approach to deal with appearance attributes as summarized in Section 3. In this paper, we build upon that work by developing an improved quadric error metric for simplifying meshes with attributes. The new metric offers the following advantages:

- It more intuitively measures error based on geometric correspondence in $\mathbf{R}^3$.
- It requires less storage, since its space complexity is linear on the number of attributes.
- It evaluates more quickly since the quadric matrix is sparse.
- It results in more accurate simplifications, as demonstrated by results.

Web: http://research.microsoft.com/~hoppe/

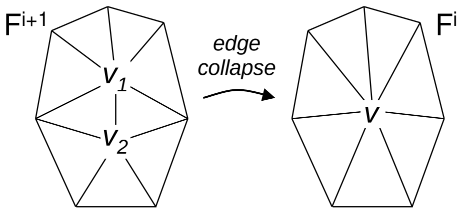


Figure 1: Edge collapse transformation.

Appearance attributes are not always continuous over the surface of the mesh. To deal with attribute discontinuities (such as creases), we extend the quadric scheme to a wedge-based mesh data structure, as described in Section 5.

In Section 6, we present two enhancements to the quadric simplification scheme, inspired by the recent geometric simplification method of Lindstrom and Turk [15]. The enhancements are memoryless simplification and volume preservation. Results of quantitative testing of all the combinations indicate that the new quadric metric, memoryless simplification, and volume preservation all contribute to improved accuracy, and do so in order of decreasing importance. As shown in Section 7, the quality of the results is surprisingly good. The simplified meshes are generally just as accurate, with respect to both geometry and attributes, as similar ones produced by the more expensive optimization in [10].

Notation In this paper, we describe a triangle mesh M by its set of vertices V and its set of faces F. Each vertex $v \in V$ has a geometric position $\mathbf{p}_v \in \mathbf{R}^3$ and a set of m attribute scalars denoted by $\mathbf{s}_v \in \mathbf{R}^m$. These two elements form a column vector $\mathbf{v}_v = \left(\begin{smallmatrix}\mathbf{p}_v \\ \mathbf{s}_v\end{smallmatrix}\right) \in \mathbf{R}^{3+m}$. (We also refer to arbitrary vectors in $\mathbf{R}^{3+m}$ using the notation $\mathbf{v} = \left(\begin{smallmatrix}\mathbf{p} \\ \mathbf{s}\end{smallmatrix}\right) \in \mathbf{R}^{3+m}$.) A mesh with (r, g, b) vertex colors therefore has $m = 3$, and a mesh with both colors and normals has $m = 6$. Each triangle face $f \in F$ is denoted as a vertex triplet (v_1, v_2, v_3).

2 PREVIOUS WORK

Most recent geometric simplification schemes coarsen a mesh through a sequence of edge collapse transformations, shown in Figure 1. These transformations have the advantage that their inverses can be stored concisely to form a progressive mesh representation [10].

In a simplification scheme based on edge collapses, two issues must be resolved: (1) the position and attributes $\mathbf{v}$ to assign to the unified vertex, and (2) the order in which to perform edge collapses. A common approach is to define a single cost metric C to determine both. The unified vertex is assigned the value $\mathbf{v}$ minimizing $C(\mathbf{v})$, and the same cost $C(\mathbf{v})$ is used to order the candidate edge collapses. We now briefly review some previous approaches to defining $C(\mathbf{v})$.

Geometric simplification of meshes Guéziec [8] constrains edge collapses to preserve mesh volume, and bounds the maximum geometric approximation error through a framework of tolerance volumes. Hoppe et al. [10, 12] sample a set of points on the original mesh, and define $C(\mathbf{v})$ as the sum of their squared distances to the approximating mesh; one drawback is that the subset of points that must be reprojected grows as the mesh is simplified. Kobbelt et al. [13] also sample points on the original mesh, but constrain their maximum distance to the approximating mesh, and use a fairness functional to order the edge collapses.

Ronfard and Rossignac [17] associate to each original vertex the set of planes spanned by its adjacent faces, merge these sets of planes after each edge collapse, and define $C(\mathbf{v})$ as the sum of squared distances from $\mathbf{v}$ to its associated planes; again a drawback is that these plane sets grow as the mesh is simplified. Garland and Heckbert [6] show that this same $C(\mathbf{v})$ can be efficiently represented as a compact *quadric error metric* (QEM), as reviewed in Section 3.

Lindstrom and Turk [15] define $C(\mathbf{v})$ as a sum of squared tetrahedral volumes between the two mesh neighborhoods of Figure 1. Specifically, each tetrahedron is formed by the vertex $\mathbf{v}$ and a face $f \in F^{i+1}$. Because each tetrahedral volume is proportional to the distance of $\mathbf{v}$ from the plane spanning f, the metric $C(\mathbf{v})$ can be seen as an instance of a QEM over the neighborhood F^{i+1} where the metric on each face $f \in F^{i+1}$ is weighted by the squared area of f. (This observation does not appear in their paper [15].) A major difference from the earlier scheme [6] is that the error metric is defined over the mesh simplified so far instead of the original mesh. We refer to this as the *memoryless* version of QEM simplification. Lindstrom and Turk also use constraints to preserve volume and boundaries.

Simplification of meshes with appearance attributes Bajaj and Schikore [1] track geometric and attribute errors on faces of the mesh to obtain error-bounded simplifications of meshes with attributes. Hoppe [10] extends the point sampling approach to also include attributes in the cost metric, but decouples geometric optimization from attribute optimization when minimizing $C(\mathbf{v})$. Garland and Heckbert [7] generalize their earlier QEM scheme to deal with surface properties, as summarized in Section 3. In this paper, we describe another, more intuitive generalization that proves to be more accurate and efficient.

Cohen et al. [5] simplify meshes with explicit texture coordinates. By tracking parametric instead of geometric correspondence, their scheme bounds the displacement of a point on the mesh with any given texture coordinate, which is the correct metric for texture-mapped surfaces. Our scheme, like [1, 7, 10], seeks to minimize the attribute deviation at any given point on the surface. This is the correct metric for vertex attributes like colors and normals that do not define a parametrization on the surface.

Image-based approaches When the attribute field on the original mesh has complex detail, it may be difficult to significantly coarsen the mesh without quickly degrading this detail. An alternative approach is to capture attribute fields as sampled texture images on the mesh faces [3, 5, 14, 16, 18]. In our opinion, such an image-based approach will gain wide acceptance as more rasterization operations (normal maps, displacement maps, etc.) are integrated into the hardware. However, there are still many attribute fields that are most concisely represented as piecewise-linear functionals using vertex attributes. The discontinuous normal field of Figure 12 and the radiosity solution of Figure 13 are good examples.

3 PREVIOUS QUADRIC ERROR METRICS

Simplification of geometry The original QEM scheme [6] addresses the case $m = 0$. It defines on each face f of the original mesh a quadric $Q^f(\mathbf{v})$ equal to the squared distance of a point $\mathbf{v} = (\mathbf{p}) \in \mathbf{R}^3$ to the plane containing the face. (The derivation of Q^f will follow shortly.) Each vertex v of the original mesh is assigned the sum of quadrics on its adjacent faces weighted by face area:

$$Q^v(\mathbf{v}) = \sum_{f \ni v} \operatorname{area}(f) \cdot Q^f(\mathbf{v}) \ . \tag{1}$$

After each edge collapse $(v_1, v_2) \rightarrow v$, the new vertex v is assigned the position $\mathbf{v}$ minimizing $Q^v(\mathbf{v}) = Q^{v_1}(\mathbf{v}) + Q^{v_2}(\mathbf{v})$, and the next edge collapse chosen is the one with the lowest such minimum.

Let us now derive $Q^f(\mathbf{v})$ for a given face $f = (v_1, v_2, v_3)$. Recall that $\mathbf{v} = (\mathbf{p})$ when $m = 0$. The signed distance of $\mathbf{p}$ to the plane $P \subset \mathbf{R}^3$ containing f is $\mathbf{n}^T\mathbf{p} + d$, where the face normal $\mathbf{n} = (\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1) / \|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)\|$ and the scalar $d = -\mathbf{n}^T\mathbf{p}_1$. As an aside, a different formulation is to obtain these parameters by solving the linear system

$$\begin{pmatrix} \mathbf{p}_1^T & 1 \\ \mathbf{p}_2^T & 1 \\ \mathbf{p}_3^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{n} \\ d \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

with the additional constraint that $\|\mathbf{n}\| = 1$.

The squared distance between point $\mathbf{p}$ and plane P is therefore

$$Q^f(\mathbf{v} = (\mathbf{p})) \ = \ (\mathbf{n}^T\mathbf{v} + d)^2 \ = \ \mathbf{v}^T(\mathbf{n}\mathbf{n}^T)\mathbf{v} + 2d\mathbf{n}^T\mathbf{v} + d^2 \ ,$$

which can be represented as a quadratic functional $\mathbf{v}^T\mathbf{A}\mathbf{v} + 2\mathbf{b}^T\mathbf{v} + c$ where $\mathbf{A}$ is a symmetric 3×3 matrix, $\mathbf{b}$ is a column vector of size 3, and c is a scalar [6]. Thus,

$$Q^f \ = \ (\mathbf{A}, \mathbf{b}, c) \ = \ \left(\left(\mathbf{n}\mathbf{n}^T \right), \left(d\,\mathbf{n} \right), \ d^2 \right)$$

is stored using $6 + 3 + 1 = 10$ coefficients. The advantage of this representation is that the quadric Q^v of Equation 1 is obtained using simple linear combinations of these coefficient vectors.[1]

After an edge collapse, the vertex position $\mathbf{v}_{min}$ minimizing $Q^v(\mathbf{v})$ is found where the gradient $(\nabla Q^v(\mathbf{v}) = 2\mathbf{A}\mathbf{v} + 2\mathbf{b})$ equals zero, which is obtained by solving the linear system

$$\mathbf{A}\mathbf{v}_{min} = -\mathbf{b} \ . \tag{2}$$

Simplification of geometry and attributes In [7], Garland and Heckbert extend their framework to deal with vertex attributes ($m > 0$). Their approach is to generalize the distance-to-plane metric in $\mathbf{R}^3$ to a distance-to-hyperplane in $\mathbf{R}^{3+m}$. That is, $Q^f(\mathbf{v})$ for $\mathbf{v} = \left(\begin{smallmatrix}\mathbf{p}\\ \mathbf{s}\end{smallmatrix}\right) \in \mathbf{R}^{3+m}$ is defined as the distance in $\mathbf{R}^{3+m}$ from $\mathbf{v}$ to the affine subspace $P' \subset \mathbf{R}^{3+m}$ spanned by the three vertices $(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$.

Let $\mathbf{v}'$ denote the projection of $\mathbf{v}$ onto this affine subspace. The error $Q^f(\mathbf{v}) = \|\mathbf{v} - \mathbf{v}'\|^2$ can be seen as the sum of two terms, the geometric distance error $\|\mathbf{p} - \mathbf{p}'\|^2$ and the attribute deviation error $\|\mathbf{s} - \mathbf{s}'\|^2$. Observe that the point $\mathbf{p}'$ does not correspond to the projection of $\mathbf{p}$ onto the plane $P \subset \mathbf{R}^3$ as it did previously. The effect is that $\mathbf{v}$ is generally not compared to the geometrically closest point, but to some *geometrically farther* point that has a closer attribute value. As a consequence, the metric may underestimate the actual error, as shown in Section 4.

The quadric $Q^f(\mathbf{v})$ consists of a matrix $\mathbf{A}$ of size $(3+m) \times (3+m)$, a column vector $\mathbf{b}$ of size $3+m$, and a scalar c. Because the matrix $\mathbf{A}$ resulting from the above formulation is dense, storage of Q requires a total of $(4+m)(5+m)/2$ coefficients, which is quadratic on m.

To trade off geometric accuracy and attribute accuracy, the user specifies for each attribute $j \in \{1 \ldots m\}$ a relative importance weight λ_j that pre-multiplies the attribute values, effectively scaling some axes in $\mathbf{R}^{3+m}$. For scale-invariance, the mesh is resized to tightly fit in the unit cube.

[1] Such a quadric can also be represented in homogeneous form as a single 4×4 symmetric matrix, but we find the $(\mathbf{A}, \mathbf{b}, c)$ notation more convenient.

4 NEW QUADRIC ERROR METRIC

Our contribution is to introduce a new quadric that defines both geometric error and attribute error based on geometric correspondence in 3D (see Figure 2). Rather than projecting a given point $\mathbf{p}$ onto the mesh face in an abstract higher-dimensional space $\mathbf{R}^{3+m}$, we perform the projection in $\mathbf{R}^3$ and compute geometric and attribute error based on this correspondence.

The error metric for a face f is defined as the sum

$$Q^f(\mathbf{v} = (\begin{smallmatrix}\mathbf{p}\\ \mathbf{s}\end{smallmatrix})) \; = \; Q^f_p(\mathbf{v}) \; + \; \sum_{j=1}^{m} Q^f_{s_j}(\mathbf{v})$$

where the *geometric error* $Q^f_p(\mathbf{v})$ is the squared distance from $\mathbf{p}$ to its projection $\mathbf{p}'$ on the plane $P \subset \mathbf{R}^3$ containing f, and the *attribute error* $Q^f_{s_j}(\mathbf{v})$ is the squared deviation between $\mathbf{s}$ and the value $\mathbf{s}'$ interpolated from face f at that projected point $\mathbf{p}'$. Let us now derive these terms.

The geometric error term is simply a zero-extended version of that in [6]:

$$Q^f_p = (\mathbf{A}, \mathbf{b}, c) = \left(\left(\begin{array}{c|c} \mathbf{n}\mathbf{n}^T & \ddots 0 \ddots \\ \hline \ddots 0 \ddots & \ddots 0 \ddots \end{array} \right), \left(\begin{array}{c} d\,\mathbf{n} \\ \hline 0 \end{array} \right), \; d^2 \right)$$

where the line dividers in $\mathbf{A}$ and $\mathbf{b}$ mark the first 3 rows and 3 columns.

To form the attribute error term $Q^f_{s_j}$, we first define a linear functional

$$\hat{s}_j(\mathbf{p}) = \mathbf{g}_j^T \, \mathbf{p} \; + \; d_j$$

that represents the expected attribute value at all points $\mathbf{p} \in \mathbf{R}^3$. The gradient $\mathbf{g}_j$ and scalar d_j are defined as follows. Naturally, $\hat{s}_j(\mathbf{p})$ should interpolate the face vertices $f = ((\begin{smallmatrix}\mathbf{p}_1\\ \mathbf{s}_1\end{smallmatrix}), (\begin{smallmatrix}\mathbf{p}_2\\ \mathbf{s}_2\end{smallmatrix}), (\begin{smallmatrix}\mathbf{p}_3\\ \mathbf{s}_3\end{smallmatrix}))$ and thus match the linear interpolant over the plane P. In addition, $\hat{s}_j(\mathbf{p})$ for an arbitrary $\mathbf{p} \in \mathbf{R}^3$ should be identical to the value $\hat{s}_j(\mathbf{p}')$ at its geometric projection on P; this is equivalent to setting $\mathbf{n}^T \mathbf{g}_j = 0$. Thus $\mathbf{g}_j$ is simply the gradient of the scalar function over the triangle face. Parameters $(\mathbf{g}_j, d_j)$ are computed by solving the 4×4 linear system

$$\begin{pmatrix} \mathbf{p}_1^T & 1 \\ \mathbf{p}_2^T & 1 \\ \mathbf{p}_3^T & 1 \\ \mathbf{n}^T & 0 \end{pmatrix} \begin{pmatrix} \\ \mathbf{g}_j \\ \\ d_j \end{pmatrix} = \begin{pmatrix} s_{1,j} \\ s_{2,j} \\ s_{3,j} \\ 0 \end{pmatrix}.$$

Since $Q^f_{s_j}(\mathbf{v}) \; = \; (\hat{s}_j(\mathbf{p}) - s_j)^2 \; = \; (\mathbf{g}_j^T \mathbf{p} + d_j - s_j)^2$, through algebraic rearrangement we obtain $Q^f_{s_j} = (\mathbf{A}, \mathbf{b}, c) =$

$$\left(\left(\begin{array}{c|cccc} \mathbf{g}_j \mathbf{g}_j^T & \ddots 0 \ddots & -\mathbf{g}_j & \ddots 0 \ddots \\ \hline \ddots 0 \ddots & \ddots 0 \ddots & 0 & \ddots 0 \ddots \\ -\mathbf{g}_j^T & \cdots 0 \cdots & 1 & \cdots 0 \cdots \\ \ddots 0 \ddots & \ddots 0 \ddots & 0 & \ddots 0 \ddots \end{array} \right), \left(\begin{array}{c} d_j \, \mathbf{g}_j \\ \hline 0 \\ -d_j \\ 0 \end{array} \right), \; d_j^2 \right)$$

where the value 1 appears in $\mathbf{A}_{3+j,3+j}$ and the $-d_j$ appears in $\mathbf{b}_{3+j}$.

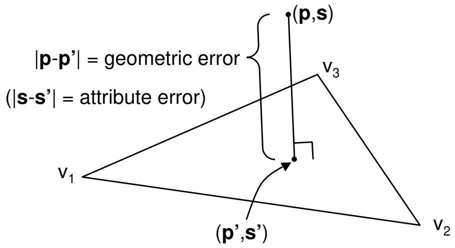


Figure 2: Correspondence between point $\mathbf{p}$ with attribute $\mathbf{s}$ and its projection onto the plane spanning face (v_1, v_2, v_3).

Example	m	Previous Q	New Q
geometry	0	10	10
+ color	3	28	23
+ normals	6	55	35
+ texture coord.	8	78	43
in general	$m > 0$	$(4+m)(5+m)/2$	$11+4m$

Table 1: Number of coefficients necessary to represent Q for various numbers m of scalar attributes, for [7] and for our new scheme.

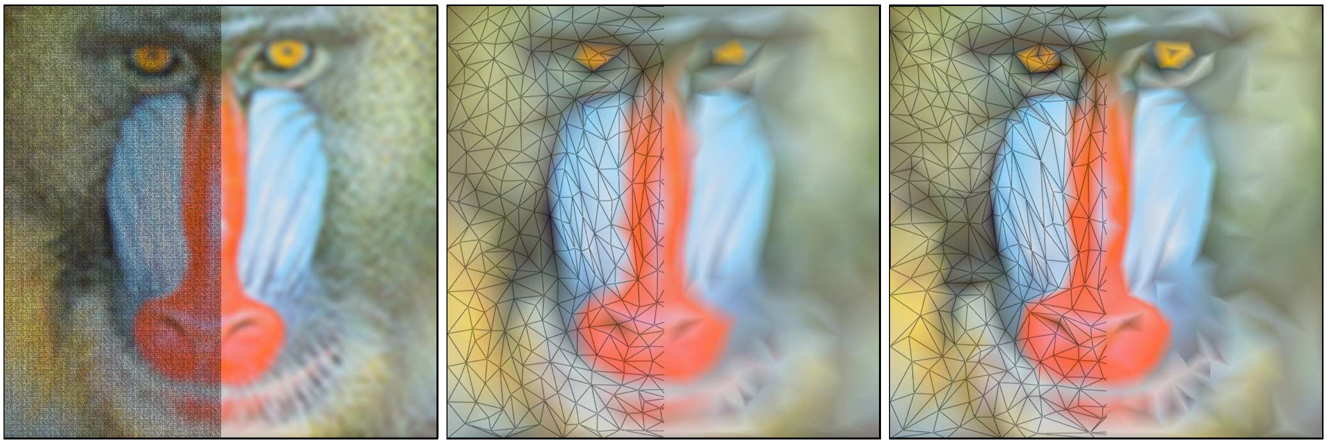
Summing all of these quadrics together yields $Q^f = (\mathbf{A}, \mathbf{b}, c) =$ (

$$\left(\begin{array}{c|c} \mathbf{n}\mathbf{n}^T + \sum_j \mathbf{g}_j \mathbf{g}_j^T & -\mathbf{g}_1 \; \cdots \; -\mathbf{g}_m \\ \hline -\mathbf{g}_1^T & \\ \vdots & I \\ -\mathbf{g}_m^T & \end{array} \right), \left(\begin{array}{c} d\mathbf{n} + \sum_j d_j \mathbf{g}_j \\ \hline -d_1 \\ \vdots \\ -d_m \end{array} \right), \; d^2 + \sum_j d_j^2$$

). Note that the first 3 rows and the first 3 columns of $\mathbf{A}$ are dense, but that the remaining $m \times m$ submatrix is the identity I. Recall that a set of weights λ_j is used to scale attribute errors relative to geometric error. If one were to define $Q = Q_p + \sum_j \lambda_j^2 Q_{s_j}$, the submatrix would have the weights λ_j^2 on its diagonal. We use the simpler approach of pre-scaling the attribute values s_j by λ_j prior to constructing and evaluating Q. In either case, the $m \times m$ submatrix is always a constant matrix times a scalar factor, and thus requires only one coefficient of storage. Overall, Q requires $11+4m$ coefficients, which is now linear on m (see Table 1).

Some attributes such as color channels have bounded extents (e.g. $0 \leq r, g, b \leq 1$). The $\mathbf{v}_{min}$ found in Equation 2 may contain attributes outside these linear inequality constraints. A fall-back strategy when this occurs could be to solve a more expensive "constrained quadratic programming" problem. We have chosen to simply truncate the attributes to their bounds and re-evaluate $Q^v(\mathbf{v})$ there. Similarly, surface normal attributes should remain normalized. However, quadratic minimization subject to quadratic constraints is an even more difficult problem, so again we leave these attributes unconstrained and renormalize the optimized values.

Figure 3 demonstrates the improvement in accuracy provided by the new quadric metric. The original mesh is a regular planar grid whose vertex colors are sampled from the "mandrill" image. The simplified meshes are obtained without the enhancements described later in Section 6. The values $\lambda_1, \lambda_2, \lambda_3$ scaling the (r, g, b) color channels relative to the unit size image are all set to $\lambda = 1$. It should be noted that for planar geometry, the previous QEM [7] obtains progressively better results as λ is decreased, converging to our QEM as λ approaches 0 (but not at $\lambda = 0$). The reason is that the hyperplane $P' \subset \mathbf{R}^{3+m}$ becomes more parallel to the $P \subset \mathbf{R}^3$. Quantitative results for the "mandrill" simplification appear later in Table 2.



(a) Original mesh (79,202 faces) (b) Simplified using Q from [7] (c) Simplified using our new Q

Figure 3: Result of simplifying a vertex-colored 200×200 mesh down to 1,000 faces using the previous QEM [7] and using our new QEM. Mesh edges are rendered on the left half of each image. Weights λ_j relating color to geometric accuracy are set to 1.

5 ATTRIBUTE DISCONTINUITIES

Meshes often have discontinuities in their attribute fields. For instance, a crease is a path of edges on a mesh across which normals are discontinuous. In radiosity solutions, intensities on adjacent patches are generally different if the patches are not parallel. Modeling such discontinuities involves storing multiple sets of attribute values per vertex. For this purpose we have chosen to introduce *wedges* [11] into our data structure (Figure 4).

A vertex is partitioned into $k \geq 1$ wedges, each wedge w_i having its own attribute vector $\mathbf{s}^i$. The corner of each face adjacent to the vertex is assigned to one of the wedges. The quadric $Q^{w_i}(\mathbf{p}, \mathbf{s}^i)$ at wedge w_i is the area-weighted sum of Q^f for its subset of adjacent faces,

$$Q^w(\mathbf{v}) = \sum_{f \ni w} \text{area}(f) \cdot Q^f(\mathbf{v}) \;, \tag{3}$$

and Equation 1 is replaced by

$$Q^v(\mathbf{p}, \mathbf{s}^1, \ldots, \mathbf{s}^k) = \sum_{i=1}^{k} Q^{w_i}(\mathbf{p}, \mathbf{s}^i) \;. \tag{4}$$

The new vertex quadric Q^v has dimension $3 + km$. Note that this variable-sized quadric Q^v need never be stored explicitly in the mesh, as it is straightforward to construct from the Q^{w_i} when an edge collapse is considered. Minimizing this quadric Q^v produces both the vertex position and all of its wedge attributes simultaneously.

For an edge collapse, the earlier strategy of merging vertex quadrics as $Q^v(\mathbf{v}) = Q^{v_1}(\mathbf{v}) + Q^{v_2}(\mathbf{v})$ must be redefined to act on wedge quadrics instead. Referring to Figure 5, we unify the wedges w'_a and w'_b following an edge collapse if both w_a and w_b extended into face f_1 prior to the edge collapse, and similarly on the other side for face f_2. For each pair of unified wedges (0, 1, or 2 pairs), we sum together their wedge quadrics. These rules reproduce the original scheme when both vertices v_1 and v_2 each have a single wedge.

One last detail is that we preserve the geometry of discontinuity curves by associating an additional quadric with every sharp edge (including boundary edges), as described in [7]. In our framework this edge quadric is added to the $Q^w(\mathbf{v})$ on the 4 corners adjacent to the edge (or 2 corners in the case of a boundary edge).

6 SIMPLIFICATION ENHANCEMENTS

Besides defining the new QEM, we also experimented with two techniques, memoryless simplification and volume preservation, and found that they further improve results.

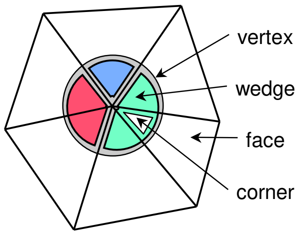


Figure 4: Wedge-based mesh representation.

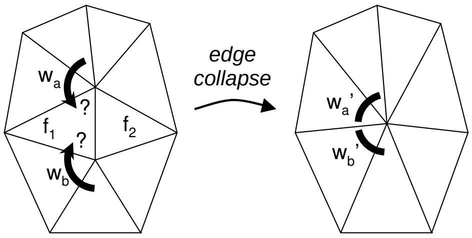


Figure 5: Tests for wedge unification after edge collapse.

6.1 Memoryless simplification

Instead of assigning $Q^w(\mathbf{v})$ to wedges in the original mesh and summing these through each edge collapse as $Q^v(\mathbf{v}) = Q^{v_1}(\mathbf{v}) + Q^{v_2}(\mathbf{v})$, we experimented with the alternative approach of memoryless simplification, in which $Q^w(\mathbf{v})$ is redefined based on the geometry and attributes of the mesh simplified so far. Thus, when evaluating an edge collapse $(v_1, v_2) \rightarrow v$, we compute $Q^v(\mathbf{v})$ using Equations 3 and 4 over the set of faces F^{i+1} in Figure 1. As mentioned earlier, the squared tetrahedral volumes used in [15] give rise to a similar metric except that it weights each Q^f by the square of its face area.

As shown in the results of Section 7, we confirm the surprising finding [15] that memoryless simplification improves the accuracy of results. Although this may seem counter-intuitive at first, it can be explained by the illustration in Figure 6. The ovals denote the freedom of the vertices to move within the surface without significantly increasing $Q^v(\mathbf{v})$. With the standard QEM scheme the Q^v are computed on the original mesh and subsequently summed. As a result, the merging of the non-parallel ovals (corresponding to fine level features) gives rise to tight spherical quadrics that lock vertices and prevent further simplification, even though the resulting mesh is planar.

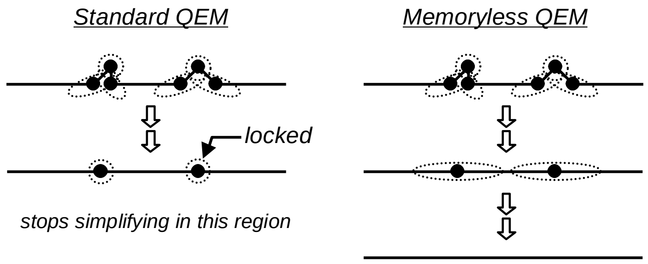


Figure 6: Illustration of standard QEM and memoryless QEM simplification. The dashed ovals symbolize the shapes of the quadric functionals Q_p^v; (a) in the standard scheme they are are computed once in a preprocess and subsequently summed during simplification; (b) in the memoryless scheme they are computed using the mesh simplified so far.

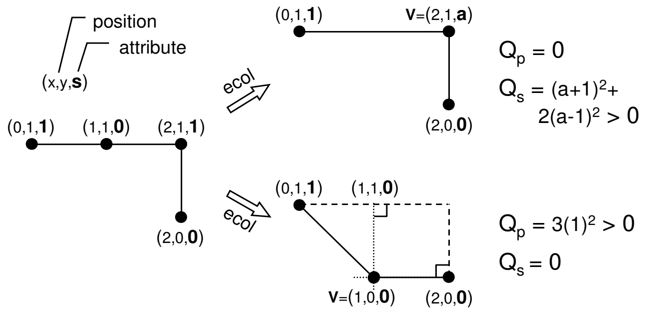


Figure 7: For a polygonal curve in $\mathbf{R}^2$, illustration of trade-off between geometric accuracy ($Q_p = 0$) and attribute accuracy ($Q_s = 0$). Attribute accuracy can cause bias towards the center of curvature when the attribute gradient is high.

In contrast, with memoryless simplification, the Q^v are recomputed at the coarser level based on the geometry simplified so far, in this case the planar surface. This ability to forget finer level detail allows simplification to proceed further when desirable.

Although memoryless simplification makes storing QEM's unnecessary, to speed up the algorithm we have found it useful to cache the values (area(f) $\cdot$ $Q^f(\mathbf{v})$) on the faces of the mesh, updating them appropriately as edges are collapsed. For this reason the compact size of our new QEM is still advantageous.

6.2 Volume preservation

Experiments reveal that the new QEM sometimes shrinks the model geometry in areas of high attribute gradient. That is, the new vertex $\mathbf{v}$ may be pushed towards the center of curvature of the surface at sharp attribute transitions.

The intuition for this effect is illustrated in Figure 7 using simplification of a polygonal curve in $\mathbf{R}^2$. The scalar field defined over the original model transitions from $\mathbf{1}$ to $\mathbf{0}$ to $\mathbf{1}$ to $\mathbf{0}$. Clearly an edge collapse over this neighborhood will not be able to preserve this complicated attribute field. Let us consider the errors measured by the quadric metric.

After the edge collapse on the upper right, the geometry is preserved exactly ($Q_p = 0$), but the attribute error Q_s cannot be made zero. The reason is that the attribute value $\mathbf{a}$ for the vertex $\mathbf{v} = (2, 1, \mathbf{a})$ cannot simultaneously interpolate the attribute gradients on all 3 original segments. In particular, $\mathbf{a}$ would have to be set to $-\mathbf{1}$ to extrapolate the leftmost segment $[(0, 1, \mathbf{1}), (1, 1, \mathbf{0})]$.

On the other hand, the edge collapse on the lower right results in geometric error ($Q_p > 0$), but "achieves" $Q_s = 0$ since the projection of $\mathbf{v} = (1, 0, \mathbf{0})$ onto each of the original 3 line segments exactly interpolates the original attributes. Intuitively, the motion of the vertex towards the center of curvature allows it to project onto the interiors of the original line segments, thus avoiding attribute *extrapolation*.

To counteract this bias towards geometric shrinkage, we introduce a volume preservation constraint. As shown in [8, 15], preserving volume during an edge collapse is equivalent to a linear constraint

$$\mathbf{g}_{VOL}^T\,\mathbf{p} + d_{VOL} = 0$$

on the position $\mathbf{p}$ of the unified vertex $\mathbf{v}$. The volumetric gradient $\mathbf{g}_{VOL}$ is the sum of the face normals of F^{i+1} (Figure 1) weighted by one third their face areas. Minimizing $Q^v(\mathbf{v})$ subject to that linear constraint is easily achieved using a system with one Lagrange multiplier γ:

$$\begin{pmatrix} \mathbf{A} & \mathbf{g}_{VOL} \\ \mathbf{g}_{VOL}^T & 0 \end{pmatrix} \begin{pmatrix} \mathbf{v}_{min} \\ \gamma \end{pmatrix} = \begin{pmatrix} -\mathbf{b} \\ -d_{VOL} \end{pmatrix} .$$

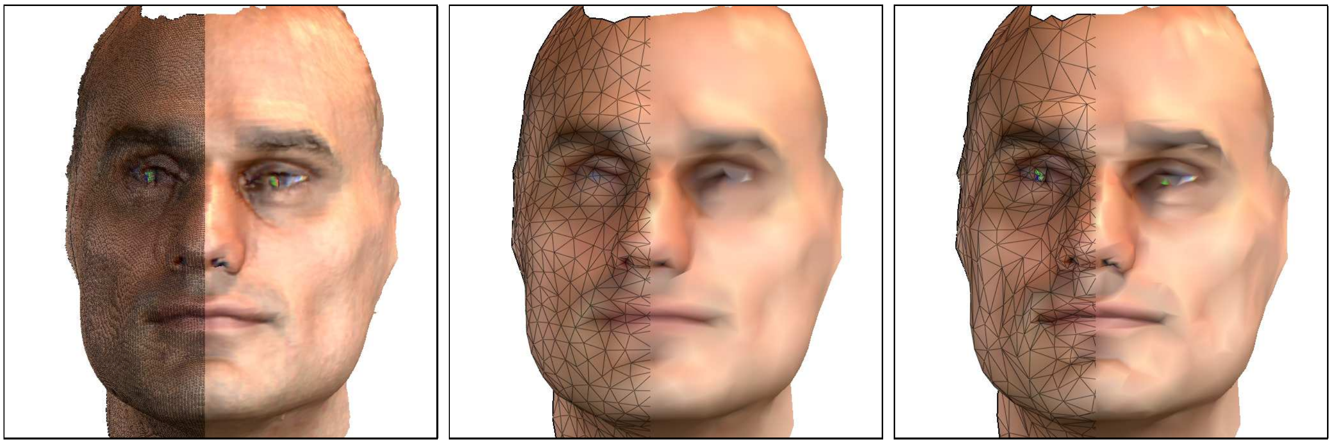
This system (or even that of Equation 2) may be ill-conditioned if the mesh neighborhood has zero Gaussian curvature, i.e. if it is planar or cylindrical. Note that the attributes cannot contribute any zero singular values because the k submatrices of size $m \times m$ on the diagonal of $\mathbf{A}$ are all the identity I. When the system is ill-conditioned, we set $\mathbf{p} = (\mathbf{p}_1 + \mathbf{p}_2)/2$ and solve for $\{\mathbf{s}_1, \ldots, \mathbf{s}_k\}$ in the remaining system.

7 RESULTS

We implemented a simplification testbed to explore various error metrics and simplification options. Because of its generality, our testbed is not designed for speed; it attains simplification rates of only 150–400 faces/second on a 450 MHz Pentium II processor. However, the elements of our scheme have already been shown to be fast [6, 7, 15]. In particular, the new QEM is faster to evaluate than in [7] due to its sparse structure. However, memoryless simplification requires more QEM constructions as in [15]. Based on the results in [7, 15], we expect that an optimized implementation would have simplification rates on the order of 10,000 faces/second.

For quantitatively measuring the accuracy of simplified meshes, we use an approach similar to [6]. The distance between two meshes M^A and M^B is obtained by sampling a collection of points from M^A and measuring their distances to their closest points on M^B. Because this is a non-symmetric process, the same number of points is also sampled from M^B and projected on M^A. Both sets of distances are combined, and statistics are reported using rms (L^2 norm) and maximum (L^∞ norm) values. For meshes with attributes, we also sample the attributes at the same points and measure their deviations from the values linearly interpolated at the closest point on the other mesh.

Table 2 presents quantitative results for the simplification of the planar mandrill model. The column of maximum errors is de-emphasized because these values are noisier (e.g. Figure 9) and seem less correlated with perceived accuracy. The table shows that, with few exceptions, all three simplification options improve results. The number marked with a '†' in Table 2 deserves more explanation. One would not expect it to be so different from the number in the next row since volume preservation should be irrelevant for a planar mesh. The explanation is that numerical noise in the mesh geometry is gradually amplified by the combination of the "shrinkage effect" (Figure 7) and memoryless re-evaluation, until the geometry undulates significantly. Fortunately, volume preservation eliminates this undesirable behavior, as discussed in Section 6.2.



(a) Original mesh (b) Simplified using [7] (c) Simplified using our scheme

Figure 8: Simplification of a vertex-colored mesh of 135,133 faces down to 1,500 faces.

Simplification options			rms	max
New Q	Memoryless	$\Delta Vol = 0$	color error	color error
			0.086	0.57
		√	0.087	0.61
	√		0.082	0.57
	√	√	0.082	0.54
√			0.068	0.48
√		√	0.068	0.76
√	√		† 0.071	0.59
√	√	√	0.054	0.33
(time-intensive scheme [10])			0.056	0.33

Table 2: Quantitative accuracy results for 1000-face "mandrill" meshes as in Figure 3, with and without (1) the new QEM, (2) memoryless simplification, and (3) volume preservation. The top row therefore corresponds to the scheme of [7], and the bottom row to our new scheme. The more expensive method from [10] is also included for comparison.

Simplification options			Rms error	
New Q	$\Delta Vol = 0$	Memoryless	Geometry	Color
			0.00135	0.035
		√	0.00113	0.035
	√		0.00091	0.029
	√	√	0.00089	0.029
√			0.00167	0.035
√		√	0.00127	0.035
√	√		0.00109	0.027
√	√	√	0.00099	0.027
(time-intensive scheme [10])			0.00095	0.027

Table 3: Results for 1500-face head meshes as in Figure 8.

Figure 8 shows the simplification of a more general vertex-colored mesh, and Table 3 shows its associated accuracy numbers. For this mesh, we set $\lambda = 0.1$ for the color attribute channels. Figures 9 and 10 plot error as a function of simplified mesh complexity. The new scheme clearly outperforms the previous QEM scheme [7], often requiring less than half as many faces for the same rms error. It also matches or outperforms the slower scheme of [10] over the most useful range of simplification complexity.

Figure 11 demonstrates the importance of including surface normal attribute values in the simplification metric. Although the mesh in Figure 11b is in fact geometrically more accurate than that in Figure 11c, its fuzzy surface normal field makes it less useful. Generally we set $\lambda = 0.02$ for surface normal attributes. Figures 12 and 13 show simplifications of meshes with attribute discontinuities. The mesh of Figure 12 has only creases, whereas the mesh of Figure 13 has both normal and color discontinuities. Table 4 compares accuracy results for all the figures using the PM scheme [10], the previous QEM [7], and the new QEM.

8 SUMMARY AND FUTURE WORK

We have described a new quadric error metric for simplifying triangle meshes with appearance attributes. The new metric captures both geometric error and attribute error based on closest-point correspondence in 3D, rather than in an abstract higher-dimensional space. We have demonstrated that this new metric produces more accurate simplifications. Moreover, it requires less storage and evaluates more quickly.

Associating the quadrics with a wedge-based data structure permits efficient simplification of models with attribute discontinuities, as was demonstrated on models with creases and illumination discontinuities. The previously introduced techniques of memoryless simplification and volume preservation further improve results. Surprisingly, the accuracy of the new scheme generally matches or exceeds that of a much slower optimization method.

In future work, it would be desirable to measure parametric error (instead of geometric error) for the simplification of texture-coordinate attributes [5]. Perhaps the quadric approach can be extended to provide a useful approximation in this context.

ACKNOWLEDGMENTS

I would like to thank Jed Lengyel, Peter Lindstrom, and John Platt for helpful comments. The 3D models are courtesy of Stanford University, Viewpoint DataLabs, and Rui Bastos at UNC. Special thanks to Steve Marschner for the Cyberware scan of Brian Guenter.

REFERENCES

[1] BAJAJ, C., AND SCHIKORE, D. Error-bounded reduction of triangle meshes with multivariate data. *SPIE 2656* (1996), 34–45.

Fig.	Rms error								
	Geometry			Color			Normals		
	[10]	[7]	New	[10]	[7]	New	[10]	[7]	New
3	10^{-7}	10^{-7}	10^{-7}	.056	.086	**.054**	-	-	-
8	**.00095**	.00135	.00099	.027	.035	**.027**	.11	.13	**.11**
11c	.00074	.00096	**.00070**	-	-	-	.13	.14	**.11**
12	.00092	.00074	**.00057**	-	-	-	**.26**	.32	.30
13	.00005	.00010	**.00002**	.036	.045	**.034**	.12	.17	**.12**

Table 4: Quantitative results for all figures. Boldface indicates the best result.

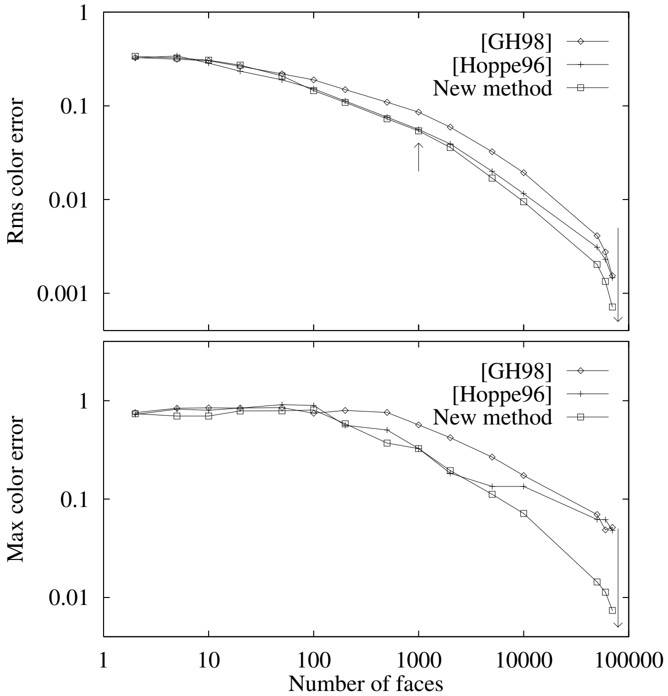


Figure 9: Rms and maximum color error for the “mandrill” model of Figure 3, using [7], [10], and our new scheme.

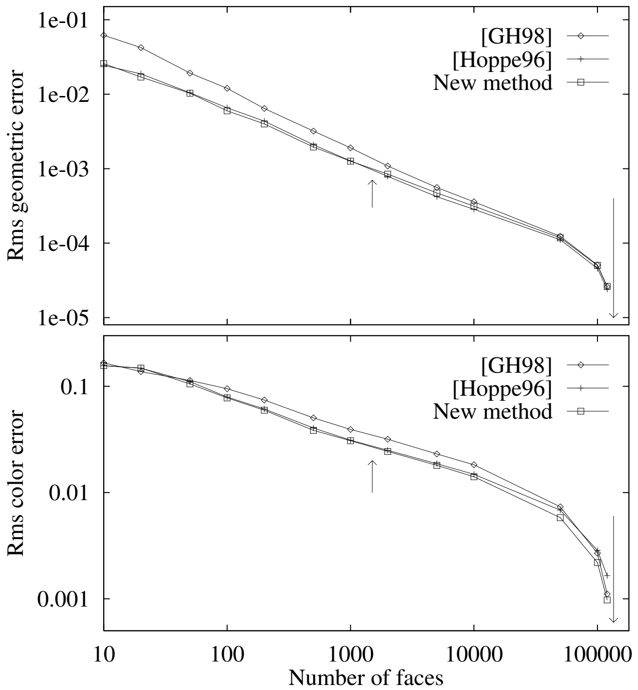


Figure 10: Accuracy results for the head model of Figure 8, using [7], [10], and our new scheme.

[2] Certain, A., Popovic, J., Duchamp, T., Salesin, D., Stuetzle, W., and DeRose, T. Interactive multi-resolution surface viewing. *Computer Graphics (SIGGRAPH '96 Proceedings)* (1996), 91–98.

[3] Cignoni, P., Montani, C., Rocchini, C., and Scopigno, R. A general method for preserving attribute values on simplified meshes. In *Visualization '98 Proceedings* (1998), IEEE, pp. 59–66.

[4] Cohen, J., Manocha, D., and Olano, M. Simplifying polygonal models using successive mappings. In *Visualization '97 Proceedings* (1997), IEEE, pp. 81–88.

[5] Cohen, J., Olano, M., and Manocha, D. Appearance-preserving simplification. *Computer Graphics (SIGGRAPH '98 Proceedings)* (1998), 115–122.

[6] Garland, M., and Heckbert, P. Surface simplification using quadric error metrics. *Computer Graphics (SIGGRAPH '97 Proceedings)* (1997), 209–216.

[7] Garland, M., and Heckbert, P. Simplifying surfaces with color and texture using quadric error metrics. In *Visualization '98 Proceedings* (1998), IEEE, pp. 263–269.

[8] Guéziec, A. Surface simplification with variable tolerance. In *Proceedings of the Second International Symposium on Medical Robotics and Computer Assisted Surgery* (November 1995), pp. 132–139.

[9] Heckbert, P., and Garland, M. Survey of polygonal surface simplification algorithms. In *Multiresolution surface modeling (SIGGRAPH '97 Course notes #25)*. ACM SIGGRAPH, 1997.

[10] Hoppe, H. Progressive meshes. *Computer Graphics (SIGGRAPH '96 Proceedings)* (1996), 99–108.

[11] Hoppe, H. Efficient implementation of progressive meshes. *Computers and Graphics 22*, 1 (1998), 27–36.

[12] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W. Mesh optimization. *Computer Graphics (SIGGRAPH '93 Proceedings)* (1993), 19–26.

[13] Kobbelt, L., Campagna, S., and Seidel, H.-P. A general framework for mesh decimation. In *Proceedings of Graphics Interface '98* (1998), pp. 43–50.

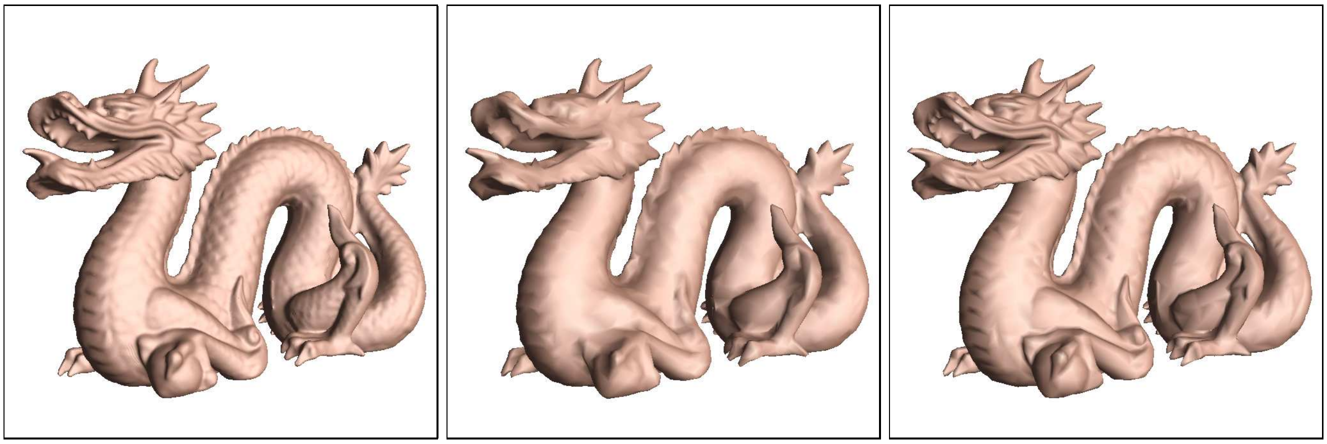
[14] Krishnamurthy, V., and Levoy, M. Fitting smooth surfaces to dense polygon meshes. *Computer Graphics (SIGGRAPH '96 Proceedings)* (1996), 313–324.

[15] Lindstrom, P., and Turk, G. Fast and memory efficient polygonal simplification. In *Visualization '98 Proceedings* (1998), IEEE, pp. 279–286.

[16] Maruya, M. Generating texture map from object-surface texture data. *Computer Graphics Forum (Proceedings of Eurographics '95) 14*, 3 (1995), 397–405.

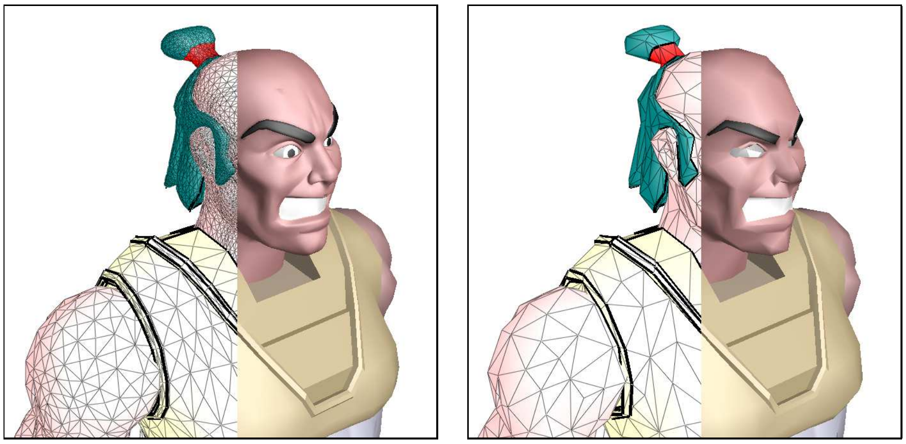
[17] Ronfard, R., and Rossignac, J. Full-range approximation of triangulated polyhedra. *Computer Graphics Forum (Proceedings of Eurographics '96) 15*, 3 (1996), 67–76.

[18] Soucy, M., Godin, G., and Rioux, M. A texture-mapping approach for the compression of colored 3D triangulations. *The Visual Computer 12* (1986), 503–514.



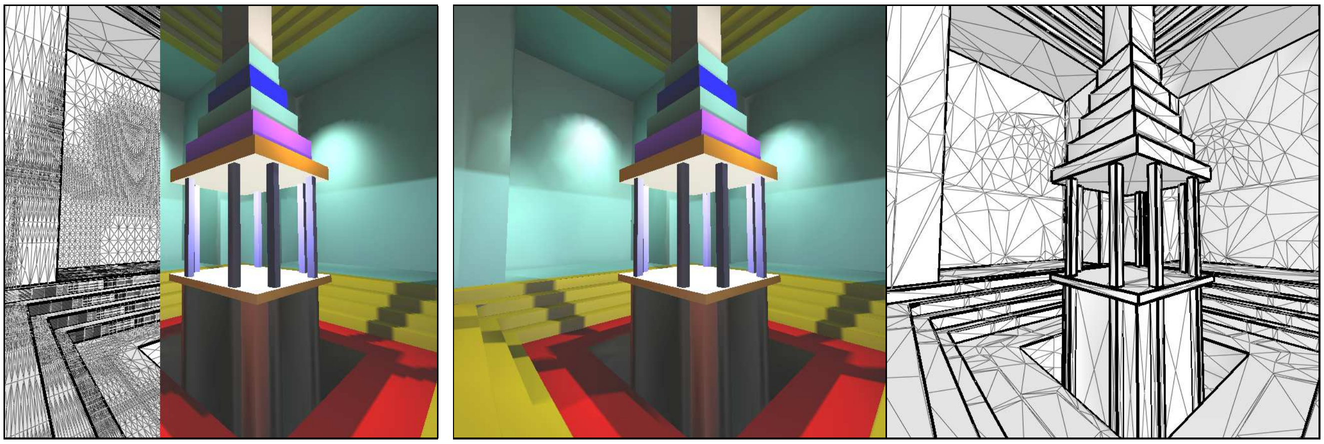
(a) Original mesh (b) Q is just geometric error (c) Q also includes normals

Figure 11: Simplification of a mesh of 920,000 faces down to 10,000 faces. For the geometric simplification in (b), normals are simply carried through. In (c) we optimize both geometry and normals, using $\lambda = 0.02$ for normals.



(a) Original mesh (42,712 faces) (b) Simplified mesh (5,000 faces)

Figure 12: Simplification of a mesh with discontinuities on normal attributes (indicated by the thick lines).



(a) Original mesh (298,468 faces) (b) Simplified mesh (5,000 faces)

Figure 13: Simplification of a mesh with discontinuities of color attributes.