Rate-Distortion Optimized Compression and View-Dependent Transmission of 3-D Normal Meshes

Jae-Young Sim, *Student Member, IEEE*, Chang-Su Kim, *Member, IEEE*, C.-C. Jay Kuo, *Fellow, IEEE*, and Sang-Uk Lee, *Senior Member, IEEE*

***Abstract*—A unified approach to rate-distortion (R-D) optimized compression and view-dependent transmission of three-dimensional (3-D) normal meshes is investigated in this work. A normal mesh is partitioned into several segments, which are then encoded independently. The bitstream of each segment is truncated optimally using a geometry distortion model based on the subdivision hierarchy. It is shown that the proposed compression algorithm yields a higher coding gain than the conventional algorithm. Moreover, to facilitate interactive transmission of 3-D data according to a client's viewing position, the server can allocate an adaptive bitrate to each segment based on its visibility priority. Simulation results demonstrate that the view-dependent transmission technique can reduce the bandwidth requirement considerably, while maintaining a good visual quality.**

***Index Terms*—Independent partitioning, normal meshes, rate-distortion optimization, view-dependent transmission, 3–D mesh compression.**

I. Introduction

THREE-dimensional (3-D) data compression is an essential technology to facilitate the transmission of 3-D models over the wired or wireless Internet. Especially, progressive compression techniques are useful when transmitting a huge amount of 3-D data interactively, since they enable successive reconstruction of the data from low to high resolutions. There has been a large amount of work on the progressive coding of triangular meshes [1]–[4], since this mesh representation is prevalent as a modeling tool for 3-D objects. A triangular mesh consists of topology as well as geometry data. The topology data characterize the connectivity information among vertices, while the geometry data describe vertex positions. In progressive mesh coding [2]–[4], a triangular mesh is converted to a set of simplified meshes with various level-of-details (LODs). Then, topology and geometry data are compressed in the increasing order of LODs.

Manuscript received February 12, 2003; revised July 15, 2004. This paper was recommended by Associate Editor H. Shum.

J.-Y. Sim and S.-U. Lee are with the School of Electrical Engineering and Computer Science, Seoul National University, Seoul 151–742, Korea (e-mail: jaeyoung@ieee.org; sanguk@ipl.snu.ac.kr).

C.-S. Kim is with the Department of Information Engineering, the Chinese University of Hong Kong, Shatin, Hong Kong (e-mail: cskim@ieee.org).

C.-C. J. Kuo is with the Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90089-2564 USA (e-mail: cckuo@sipi.usc.edu).

Digital Object Identifier 10.1109/TCSVT.2005.848349

A triangular mesh generally has an irregular topology structure, which demands a higher bitrate for coding. To overcome this drawback, efficient progressive mesh compression algorithms have been proposed in [5]–[7] based on remeshing techniques. Remeshing is a process of converting an irregular mesh into a semiregular one, in which the valence of most vertices is equal to six. In other words, most vertices are connected to six adjacent vertices. The topology data of the resulting semiregular mesh can be described very compactly. The geometry data can also be effectively compressed by exploiting the regular structure of vertices. Specifically, the geometry data can be regarded as wavelet coefficients, which can be compressed with a zerotree coding algorithm [8].

Although wavelet-based algorithms [5]–[7] achieve a high coding gain, the local characteristics of wavelet coefficients can be exploited to improve the coding gain further. In this paper, we propose an algorithm that first divides an input mesh into several segments and then encodes each segment independently according to its local characteristics. An optimal rate is assigned to each segment to minimize the geometry distortion subject to the constraint on a total rate with the Lagrangian multiplier method. Simulation results demonstrate that the proposed algorithm yields a higher coding gain than the conventional wavelet algorithm described in [7].

Furthermore, since many Internet-based 3-D applications require interactive communications between a server and multiple clients, we propose a view-dependent streaming algorithm to facilitate the server-client interactions. The proposed algorithm encodes each segment independently so that the compressed bitstream for each segment can be truncated at an arbitrary point. We employ a segment visibility measure to find the optimal set of truncation points, which maximizes the quality of reconstructed 3-D mesh from a client's viewing position. It is shown that the transmission bandwidth can be reduced significantly by allocating a major portion of bitrates to visible segments.

This paper is organized as follows. Related previous work is reviewed in Section II. The remeshing algorithm, which converts an irregular mesh into a normal one, is described in Section III. The rate-distortion (R-D) optimized compression algorithm and the view-dependent transmission algorithm are detailed in Sections IV and V, respectively. Simulation results are presented in Section VI. Finally, concluding remarks are given in Section VII.

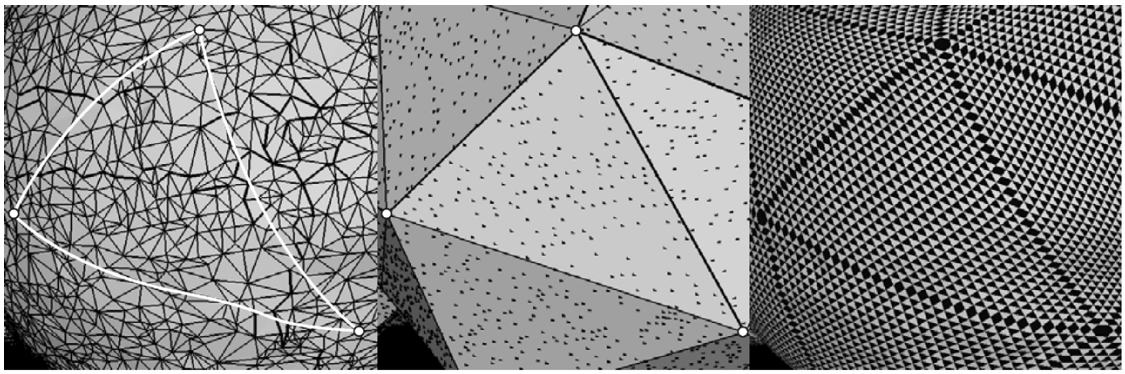


Fig. 1. Remeshing example, where the triangles of the semiregular mesh are illustrated with a normal flipping pattern to show the semiregular connectivity. (a) Irregular mesh. (b) Base mesh. (c) Semiregular mesh.

II. Review of Previous Work

A. Remeshing and Wavelets

Remeshing is a technique of converting an irregular mesh into a semiregular one [9]–[11]. A remeshing example is given in Fig. 1. An irregular mesh is simplified to a base mesh, and then the base mesh is refined to a semiregular mesh by adding new vertices systematically. Note that vertices in the semiregular mesh are positioned more regularly than those in the irregular mesh, and most vertices in the semiregular mesh have a valence 6.

The remeshing process consists of two phases: parameterization and refinement. In the parameterization phase, a mapping from a two-dimensional (2-D) domain, which is homomorphic to a topological disk, to a 3-D surface is defined. Parameterization is usually performed in the mesh simplification step, where each base triangle is employed as the desired 2-D domain. To explain this idea clearly, let us consider an example depicted in Fig. 1. In this figure, vertices within the region bounded by white curves in Fig. 1(a) are projected onto the corresponding base triangle. These projected vertices are depicted by black dots in Fig. 1(b). Each projected vertex onto the base triangle contains the information of the original vertex position. By interpolating these original vertex positions, any point on the base triangle can be mapped approximately to a point (not necessarily a vertex) in the original mesh. Eck *et al.* [9] proposed an algorithm to construct a base mesh using the Voronoi partitioning. It computes a piecewise linear harmonic map that minimizes the parameterization distortion using the conjugate gradient solver. Duchamp *et al.* [12] developed a more efficient computation method for harmonic maps based on the hierarchical mesh reduction. Lee *et al.* [10] proposed an algorithm to find a smoother parameterization, which employs vertex removal operations for mesh simplification.

Based on the parameterization information, the base mesh can be refined into a semiregular mesh with subdivision connectivity. Fig. 2 illustrates the 1–4 subdivision from the lth level to the $(l+1)$th level. Let us use v_e^l, v_f^l, and v_g^l to represent three vertices at the lth level, and v_e^{l+1}, v_f^{l+1}, and v_g^{l+1} to represent their corresponding vertices at the $(l+1)$th level. The triangle (v_e^l, v_f^l, v_g^l) is divided into the four sub-triangles, $(v_e^{l+1}, \hat{v}_t^{l+1}, \hat{v}_s^{l+1})$, $(\hat{v}_t^{l+1}, v_f^{l+1}, \hat{v}_r^{l+1})$, $(\hat{v}_s^{l+1}, \hat{v}_r^{l+1}, v_g^{l+1})$, and

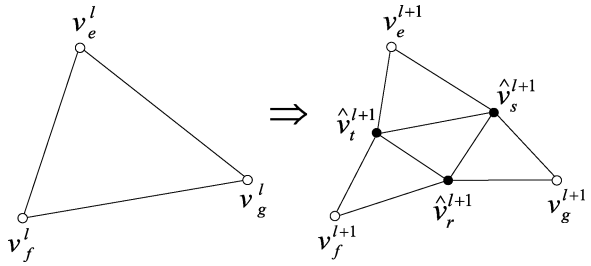


Fig. 2. The one-to-four subdivision.

$(\hat{v}_r^{l+1}, \hat{v}_s^{l+1}, \hat{v}_t^{l+1})$ by adding three new vertices $\hat{v}_r^{l+1}$, $\hat{v}_s^{l+1}$, and $\hat{v}_t^{l+1}$. There are several subdivision methods to determine the positions of vertices at the $(l+1)$th level from those of the lth level [13]. They can be classified into two types: the interpolating and the approximating approaches. In the interpolating approach, a vertex at a coarser level is also a vertex at a finer level, i.e., $v_e^l = v_e^{l+1}$, $v_f^l = v_f^{l+1}$, $v_g^l = v_g^{l+1}$. In contrast, vertex positions may not be preserved as the level increases in the approximating approach. Fig. 3 represents the butterfly subdivision which is an interpolating scheme [14]. Fig. 3(a) shows the weights used to predict a newly added vertex v. Thus, in Fig. 3(b), the position $\hat{\varphi}^{l+1}$ of a new vertex $\hat{v}^{l+1}$ is obtained via

$$\hat{\varphi}^{l+1} = \frac{1}{2}\sum_{j=1}^{2}\varphi_{a_j}^l + \frac{1}{8}\sum_{j=1}^{2}\varphi_{b_j}^l - \frac{1}{16}\sum_{j=1}^{4}\varphi_{c_j}^l \qquad (1)$$

where $\varphi_{a_j}^l$, $\varphi_{b_j}^l$, and $\varphi_{c_j}^l$ denote the positions of $v_{a_j}^l$, $v_{b_j}^l$, and $v_{c_j}^l$, respectively.

The newly added vertex $\hat{v}^{l+1}$ is predicted from the lower level vertices so that it does not lie on the original mesh surface in general. The actual remeshing point v^{l+1} is found on the original surface which corresponds to the center of edge $(v_{a_1}^l, v_{a_2}^l)$ on the parameter domain. Then the difference $v^{l+1} - \hat{v}^{l+1}$ between a remeshing point and its prediction is called a refinement vector or a wavelet coefficient. The semiregular mesh geometry can hence be described by the base mesh geometry and the corresponding hierarchical wavelet coefficients [15].

In addition, each refinement vector can be constrained to have the direction of the surface normal, which is called the normal mesh representation [11]. Then, each wavelet coefficient is compactly represented by a scalar instead of a 3-D

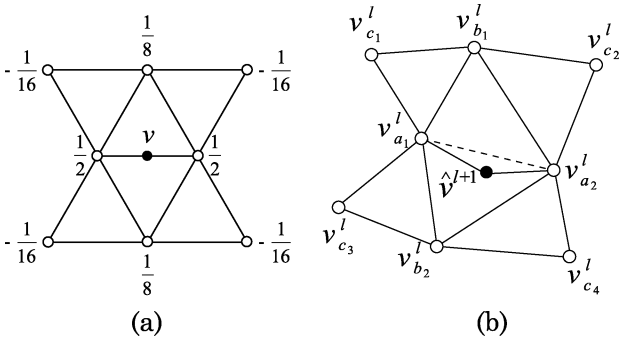


Fig. 3. Butterfly subdivision. (a) Butterfly structure and weights. (b) Position of newly added vertex $\hat{v}^{l+1}$ is predicted by those of the neighboring vertices on the butterfly structure.

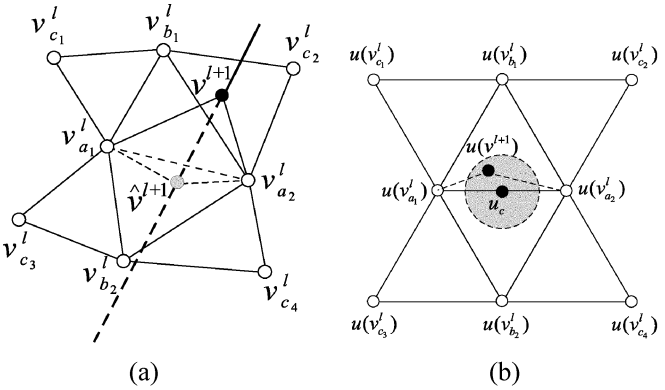


Fig. 4. Normal remeshing. (a) Refinement step and (b) its configuration on the parameter domain, where $u(v)$ denotes the parameter coordinates of v. The validity of the piercing point is tested by checking whether it lies within the shaded circle on the parameter domain or not.

vector, yielding further coding gain [7]. Fig. 4(a) illustrates the normal remeshing procedure. The normal line is first evaluated on $\hat{v}^{l+1}$, whose direction is determined by those of the neighboring vertices at the lth level. Then, the piercing point v^{l+1}, on which the normal line passes through the original mesh surface, is employed as a candidate for the remeshing point. Fig. 4(b) shows the parameter domain configuration, where $u(v)$ denotes the parameter coordinates of v. Let $u_c = \{u(v^l_{a_1}) + u(v^l_{a_2})\}/2$. The piercing point can degrade the shape of the original mesh when $u(v^{l+1})$ lies far from u_c or there can be no piercing point at all. In such a case, the remeshing point is determined from u_c, using the parameterization mapping, without the constraint of the normal direction. It is called a nonnormal vertex and its wavelet coefficient is described by a 3-D vector. In this work, the piercing point is accepted only if

$$\left|u(v^{l+1}) - u_c\right| \leq \tau \cdot \left|u\left(v^l_{a_1}\right) - u_c\right|.$$

The constant τ is set to 0.2 for the first and the second refinement levels, and 0.6 for the finer refinement levels.

B. Coding

Schröder and Sweldens [16], [17] developed a scheme for the wavelet representation of functions on a sphere. Based on this scheme, Kolarov and Lynch [18] combined the wavelet transform and the SPIHT zerotree coding algorithm to compress a function on 2-manifolds. They introduced the notion of G-tree as a tree structure for general 2-manifolds. In a G-tree, the base manifold is the root node, and subdivided triangles and higher level vertices become offsprings of a lower level triangle. Morán and García [5] extended this approach to compress meshes with subdivision connectivity progressively. Moreover, Khodakovsky *et al.* [6], [7] proposed a progressive geometry compression algorithm for semiregular meshes and normal meshes based on the edge-based tree structure and the zerotree coder.

Fig. 5(a) shows the parent-offspring relation between the vertices at the lth and the $(l+1)$th levels based on the edge-based tree. v^l_i is a parent node and $v^{l+1}_{i,j}$'s $(1 \leq j \leq 4)$ are four offsprings of v^l_i. Each $v^{l+1}_{i,j}$ at the $(l+1)$th level has its own four offsprings at the $(l+2)$th level, and so on. The vertex on each base edge becomes the root node of an edge-based tree. In Fig. 5(b), (v^0_e, v^0_f, v^0_g) is a base triangle, and three vertices v^1_r, v^1_s, and v^1_t are root nodes. The vertices at the second level are classified into three sets $v^2_{r,j}$'s, $v^2_{s,j}$'s, and $v^2_{t,j}$'s, which are the offsprings of v^1_r, v^1_s, and v^1_t, respectively. In this way, all vertices in a semiregular mesh can be covered, without overlapping, by base vertices and a number of edge-based trees.

C. View-Dependent Transmission

It requires a huge amount of computations to render detailed 3-D models. Many attempts have been made to develop fast and efficient rendering techniques. An approach is the view-dependent rendering, which assigns different LODs according to the visibility of the models. Several algorithms have been proposed for the view-dependent simplification and refinement of height field surfaces [19], irregular meshes [20]–[22], and meshes with subdivision connectivity [23], [24].

Some effort has been also made on the view-dependent compression and transmission of 3-D models to facilitate interactive communication of 3-D data in server-client environments. In [25], [26], vertex split operations are transmitted to selectively refine a 3-D model based on the vertex hierarchies. Yang *et al.* [27] proposed a view-dependent transmission algorithm for compressed bitstreams of irregular meshes. Grabner and Zach [28] employed an adaptive quantization scheme to provide locally adaptive resolutions to geometry data. Gioia *et al.* [29] proposed a rendering and transmission algorithm for semiregular meshes, which can add or suppress wavelet coefficients to perform view-dependent reconstruction of large meshes at the decoder side. Also, texture data can be transmitted in a view-dependent way [30].

III. Preprocessing

In this work, an input irregular mesh $\mathcal{M}$ is first simplified to a base mesh $\mathcal{M}^0$. Then, $\mathcal{M}^0$ is refined L times

$$\mathcal{M}^0 \rightarrow \mathcal{M}^1 \rightarrow \cdots \mathcal{M}^L$$

to obtain a normal mesh $\mathcal{M}^L$.

In the simplification process, we use half-edge collapse operations with quadric error metrics [31] and the polar map [10] to obtain the initial base mesh and its parameterization. The simplification scheme in [31] provides base mesh $\mathcal{M}^0$ with an acceptable visual quality. However, $\mathcal{M}^0$ can be preprocessed for

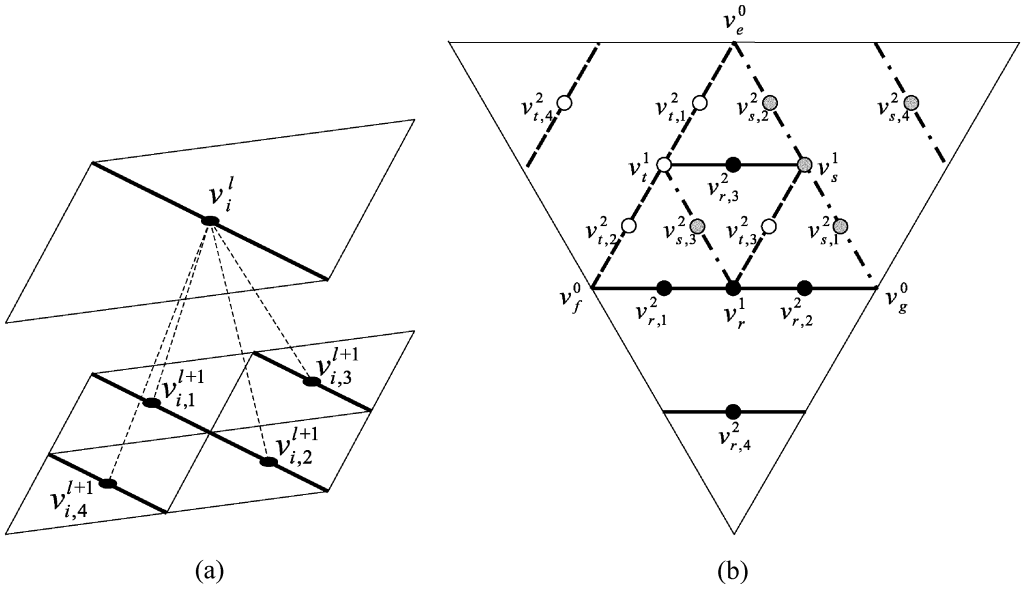


Fig. 5. Edge-based tree. (a) Parent-offspring relation, where v_i^l is a parent node and $v_{i,j}^{l+1}$'s $(1 \leq j \leq 4)$ are four offsprings of v_i^l. (b) Covering of vertices. There are three root vertices v_r^1, v_s^1, and v_t^1.

better geometric fidelity to provide a robust remeshing result. In [11], a base vertex is repositioned to be closer, in the parameter domain, to the center of the adjacent base triangles. In this work, we reposition base vertices not only to balance them in the parameter domain but also to reduce the squared sum of wavelet coefficients to improve the coding gain.

Let ψ_i^l represent the wavelet coefficient for v_i^l. Wavelet coefficients ψ_i^1's for the roots of edge-based trees are included in an energy function, and base vertices are translated to minimize the energy function. Fig. 6(a) shows how base vertex v_i^0 is used in the prediction of associated vertices at the first level. Note that $v_{\alpha_j}^1$'s, $v_{\beta_j}^1$'s, and $v_{\gamma_j}^1$'s have v_i^0 as a butterfly neighbor. Based on the structure in Fig. 3, they are predicted from v_i^0 with butterfly weights 1/2, 1/8, and $-1/16$, respectively. Thus, when v_i^0 is repositioned, wavelet coefficients or prediction errors of $v_{\alpha_j}^1$'s, $v_{\beta_j}^1$'s, and $v_{\gamma_j}^1$'s change accordingly. The energy function $\mathcal{E}$ for v_i^0 is defined as

$$\mathcal{E}\left(v_i^0\right) = \sum \left(\psi_{\alpha_j}^1\right)^2 + \sum \left(\psi_{\beta_j}^1\right)^2 + \sum \left(\psi_{\gamma_j}^1\right)^2 \qquad (2)$$

which is the squared sum of affected wavelet coefficients.

Fig. 6(b) represents the parameter domain corresponding to the base triangles adjacent to v_i^0. The small dots depict the original vertices, which are mapped to this region by the polar map [10]. As shown in Fig. 6(b), v_i^0 is translated to one of those dots, $\bar{v}_i^0$, to minimize the cost function in (2). This procedure is iteratively applied to all base vertices until the decrease in the total energy becomes negligible. Fig. 7 shows the energy minimization results for the "Teeth" model. It is observed that the base mesh yields a more uniform triangles as the iteration continues.

In the energy minimization procedure, the initial parameterization is modified following the change of the base mesh. Then, with the modified parameterization, the energy-minimized base mesh $\mathcal{M}^0$ is repeatedly refined to obtain a normal mesh $\mathcal{M}^L$. The butterfly subdivision is adopted in the refinement.

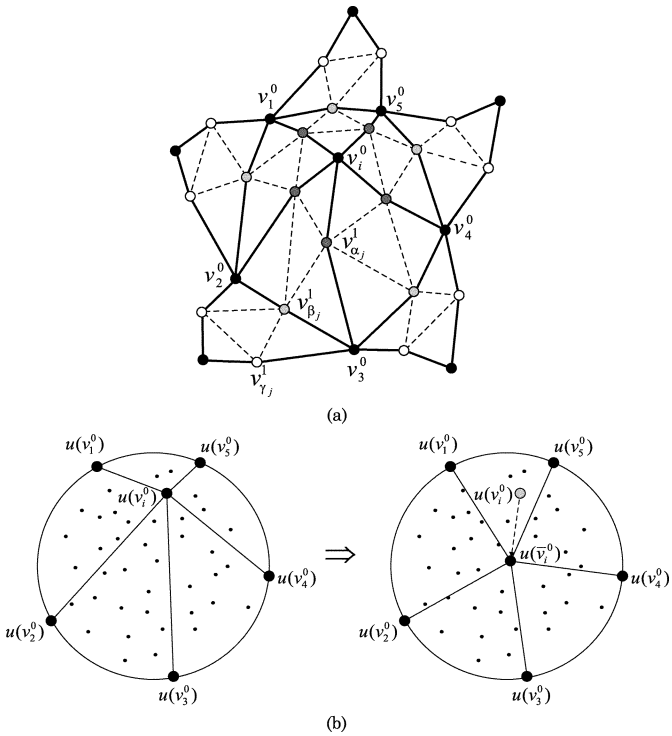


Fig. 6. Energy minimization procedure. (a) Vertices at the first level which are predicted from the base vertex v_i^0 based on the butterfly structure. (b) In the parameter domain, $u(v_i^0)$ is moved to $u(\bar{v}_i^0)$ to minimize the energy function.

Consequently, $\mathcal{M}^L$ can be represented by the base mesh and its geometrical refinements for higher level vertices, i.e., their wavelet coefficients. The base mesh can be encoded by any static mesh compression algorithm, e.g., the algorithm in [32] or [33]. The wavelet coefficients can be compressed effectively by exploiting the hierarchical structure. An R-D optimized wavelet coefficient compression algorithm is proposed in the next section.

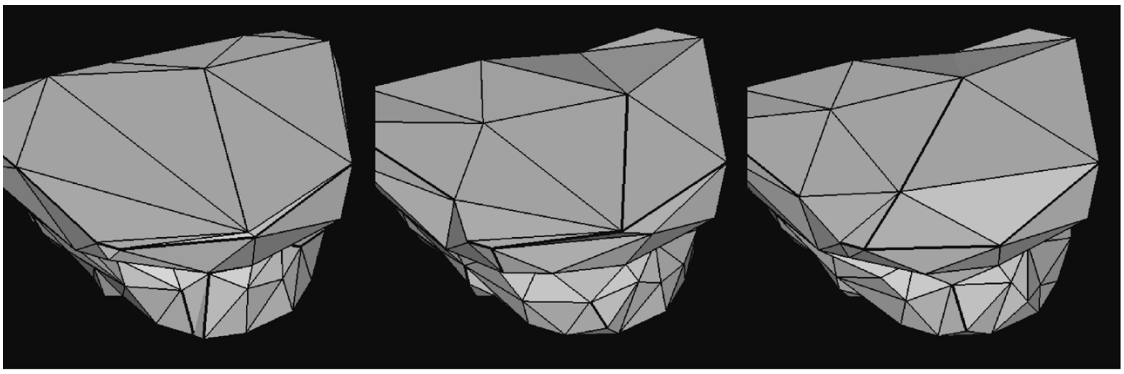


Fig. 7. Results of energy minimization applied to the "Teeth" model.

IV. R-D Optimized Wavelet Coefficient Compression

While conventional algorithms proposed in [5]–[7] encode all wavelet coefficients as a whole, our algorithm partitions them into disjoint segments and encodes each segment adaptively to maximize the R-D performance.

A. Mesh Partitioning

There are several partitioning algorithms for irregular meshes. For example, Mangan and Whitaker [34] used the watershed scheme to partition a 3-D mesh, and Zuckerberger *et al.* [35] partitioned a polyhedral surface based on the flooding convex decomposition and the watershed segmentation. Compared with irregular mesh partitioning, semiregular meshes can be partitioned in a relatively simple way. As described in Section II-B, edge-based trees can provide a disjoint partitioning of vertices in a semiregular mesh [6]. Alternatively, a segment can be defined for each base triangle, and the vertices can be partitioned into appropriate segments [5].

We adopt an edge-based tree in Fig. 5 as one segment for a normal mesh in this work. However, as shown in Fig. 5(b), three tree segments, which grow from v_r^1, v_s^1, and v_t^1, intersect one another within the base triangle (v_e^0, v_f^0, v_g^0). Thus, the conventional edge-based tree structure cannot yield a compact localization of a segment. To address this problem, we propose two more edge-based tree structures called the "scissor" tree and "arrow" tree in this work. Fig. 8(a) shows a scissor tree and Fig. 8(b) illustrates the covering of vertices using scissor trees. All four offsprings $v_{i,j}^{l+1}$'s $(1 \le j \le 4)$ are reachable from the parent v_i^l via a single edge at the $(l+1)$th level. In contrast, in Fig. 5(a), $v_{i,3}^{l+1}$ or $v_{i,4}^{l+1}$ is reachable from v_i^l via at least two edges. Thus, the scissor tree structure alleviates the intersection problem and provides a more compact localization of a segment than the conventional tree structure.

Similarly, as shown in Fig. 8(c) and (d), the arrow tree structure also localizes a segment compactly. However, two arrow trees can conflict with each other at a base edge, which is incident with an odd valence vertex. Fig. 9 shows an exceptional patching example, where dotted arrows represent the directions of arrow trees. Since the valence of v_i^0 is 5, the orientations of two base triangles (v_i^0, v_e^0, v_f^0) and (v_f^0, v_g^0, v_i^0) conflict with each other at $v_{j_5}^1$. In other words, edge (v_i^0, v_f^0) cannot be associated with an arrow tree. In this case, a scissor tree is used instead to reconcile the opposite directions and provide a disjoint covering of all vertices.

All three types of edge-based trees can be employed in the following R-D optimized compression and view-dependent transmission. Especially, in view-dependent transmission, the scissor and arrow structures provide better visual quality than the conventional structure, since they result in the compact localization of a segment.

B. Optimal Truncation of Bitstreams

Each segment (i.e., the edge-based tree) in a normal mesh is encoded by the SPIHT algorithm [8]. As mentioned in Section II-A, there are two types of vertices in a normal mesh: normal vertices and nonnormal vertices. We use one bit per vertex to specify whether it is a normal vertex or not. The SPIHT algorithm can be directly applied to encode scalar wavelet coefficients for normal vertices. On the other hand, to encode a vector wavelet coefficient $(\Delta x, \Delta y, \Delta z)$ for a nonnormal vertex, $\mathrm{sgn}(\Delta x) \cdot \sqrt{(\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2}$ is first encoded as a scalar, where $\mathrm{sgn}(\Delta x)$ denotes the sign of Δx. After the scalar is encoded as significant, the encoding of Δy and Δz follows.

In the SPIHT algorithm, the compressed bitstream consists of significant bits, sign bits, and magnitude refinement bits. In this work, truncation points are defined as last bits of significance encoding passes and magnitude refinement passes for each vertex, and a packet is defined as a set of bits between two consecutive truncation points. Let $\mathcal{C}_k$ denote the kth segment, and $\mathcal{P}_k^m$ denote the mth packet for $\mathcal{C}_k$. Also, let R_k^m denote the cumulative number of bits in the first m packets $\mathcal{P}_k^1, \mathcal{P}_k^2, \ldots, \mathcal{P}_k^m$, and D_k^m denote the distortion of the segment $\mathcal{C}_k$ reconstructed using the first m packets. Then, the packet $\mathcal{P}_k^m$ is associated with the incremental rate and distortion

$$(\Delta R_k^m, \Delta D_k^m) = \left(R_k^m - R_k^{m-1}, D_k^m - D_k^{m-1}\right).$$

It is expected that the R-D curve is convex so that the slope $S_k^m = -\Delta D_k^m / \Delta R_k^m$ is a monotonic decreasing function of m. However, it is possible for some m that S_k^m is larger than S_k^{m-1}. This abnormality is eliminated by merging $\mathcal{P}_k^{m-1}$ and $\mathcal{P}_k^m$ into one packet [36].

Let R_{total} be the total bit budget for all segments $\mathcal{C}_k$ with $k = 1, 2, \ldots, N$, where N is the number of segments. Then, the

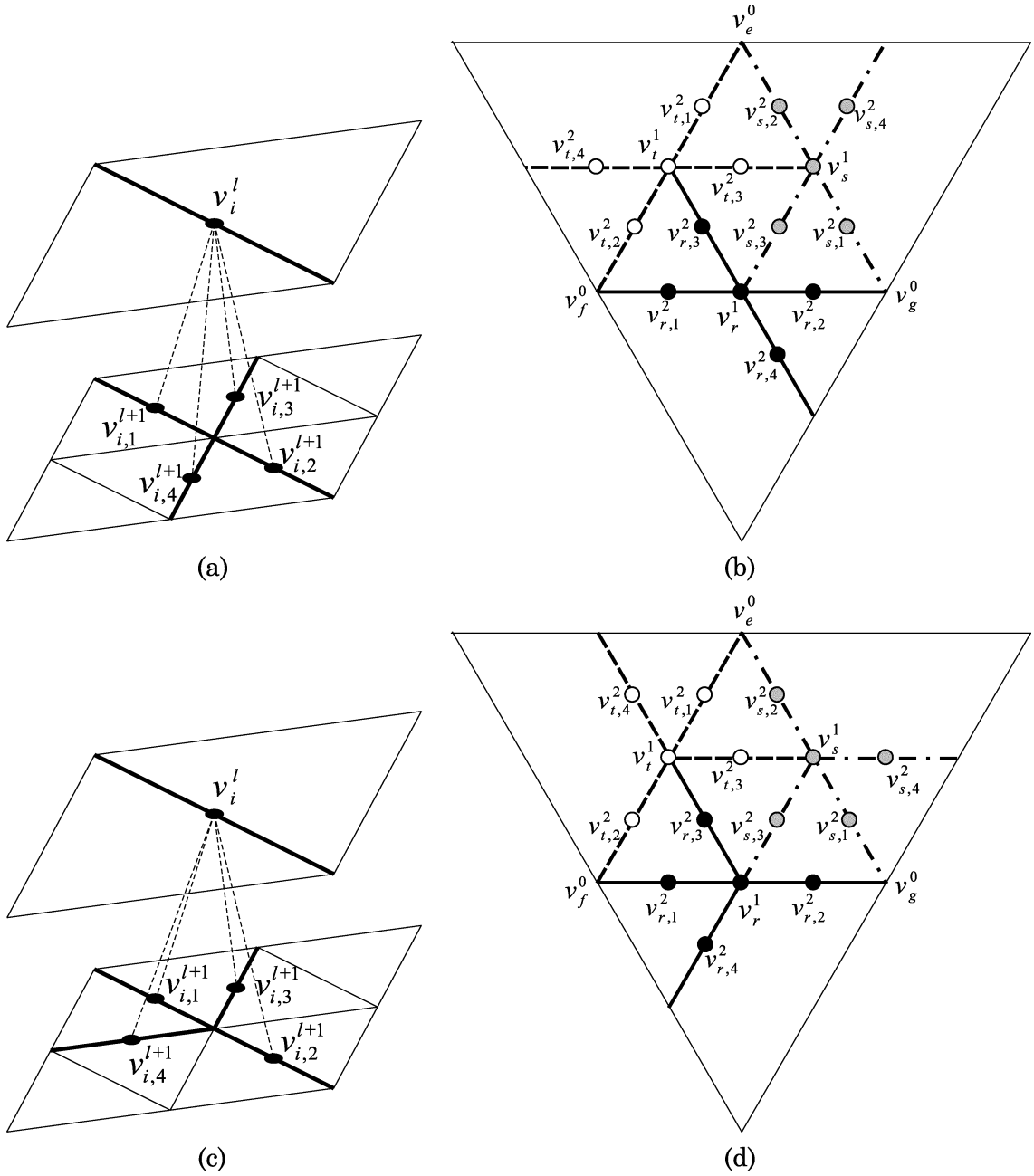


Fig. 8. Two modified edge-based trees. (a) Scissor tree. (b) Covering of vertices using scissor trees. (c) Arrow tree. (d) Covering of vertices using arrow trees.

optimization problem is to find the set of $m(k)$'s that minimizes $\sum_{k=1}^{N} D_k^{m(k)}$ subject to the constraint $\sum_{k=1}^{N} R_k^{m(k)} \leq R_{\text{total}}$. To find the set, we minimize the following cost function:

$$\sum_{k=1}^{N} \left(D_k^{m(k)} + \lambda R_k^{m(k)} \right) \tag{3}$$

where λ is the Lagrangian multiplier. By varying λ, the optimal solution satisfying the rate constraint can be found. In practice, all packets $\mathcal{P}_k^m$ are sorted in the decreasing order of slopes S_k^m. Then, the encoder continues popping the packet with the largest slope from the sorted list and transmitting it to the decoder, until the total bit budget is exhausted.

C. Distortion Model

The geometry distortion $D(\mathcal{M}^L, \tilde{\mathcal{M}}^L)$ between the original normal mesh $\mathcal{M}^L$ and a reconstructed mesh $\tilde{\mathcal{M}}^L$ is defined as

$$D(\mathcal{M}^L, \tilde{\mathcal{M}}^L) = \frac{1}{V} \sum_{v_i \in \mathcal{M}^L} \|\varphi_i - \tilde{\varphi}_i\|^2 \tag{4}$$

where V is the number of vertices in $\mathcal{M}^L$, and φ_i and $\tilde{\varphi}_i$ denote the original and reconstructed positions of v_i, respectively. The remeshing error is not incorporated in the geometry distortion, since remeshing and compression are performed separately. Note that the remeshing error is usually negligible as compared to the quantization distortion in the compression procedure.

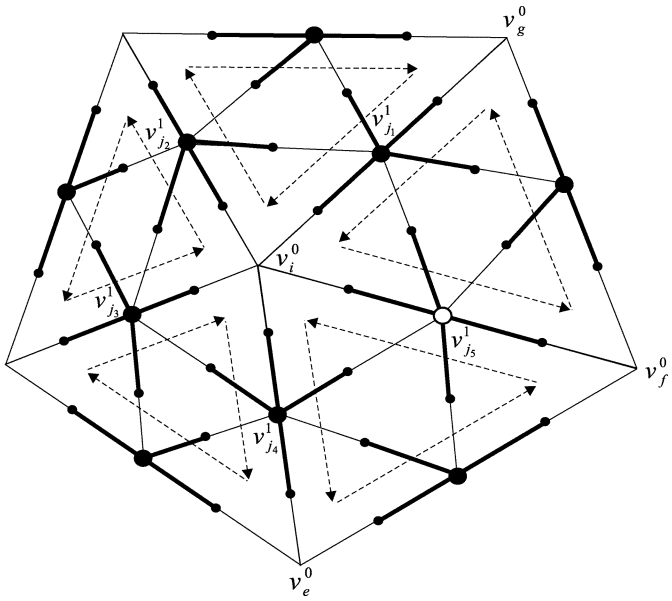


Fig. 9. Exceptional patching example of the arrow trees. Since a base vertex v_i^0 has an odd valence 5, the directions of two triangles (v_i^0, v_e^0, v_f^0) and (v_f^0, v_g^0, v_i^0) conflict each other on edge (v_i^0, v_f^0). We substitute the arrow tree with the scissor tree to achieve a disjoint covering of vertices.

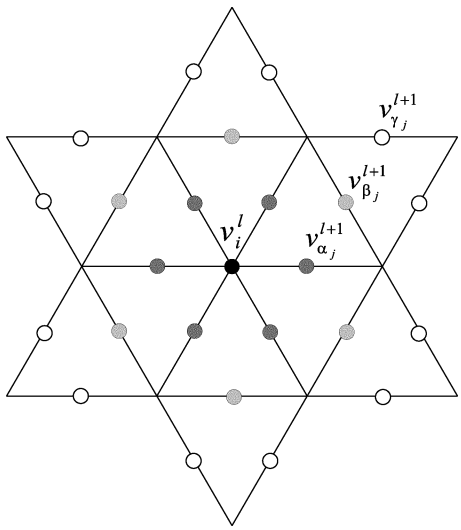


Fig. 10. Locally associated vertices of v_i^l based on the butterfly subdivision. The vertices at level l are classified into $v_{\alpha_j}^{l+1}$'s, $v_{\beta_j}^{l+1}$'s and $v_{\gamma_j}^{l+1}$'s, which are depicted in different gray shades. The vertex v_i^l is a butterfly neighbor of $v_{\alpha_j}^{l+1}$'s, $v_{\beta_j}^{l+1}$'s and $v_{\gamma_j}^{l+1}$'s with weighs 1/2, 1/8, and $-1/16$, respectively.

Vertex positions are related to wavelet coefficients that are prediction errors in the butterfly subdivision. Thus, the geometry distortion is related to the mean squared error of wavelet coefficients. However, since the synthesis filter in the butterfly subdivision is not orthogonal, the geometry distortion is not linearly proportional to the mean squared error of wavelet coefficients, i.e.,

$$\sum \|\varphi_i - \tilde{\varphi}_i\|^2 \neq c \cdot \sum (\psi_i - \tilde{\psi}_i)^2$$

where ψ_i and $\tilde{\psi}_i$ are the original and reconstructed wavelet coefficients of v_i, respectively. Obviously, wavelet coefficients can be inverse-transformed to vertex positions, and the exact geometry distortion can be computed. However, this approach demands too high computational burden in the R-D optimization procedure. In our approach, wavelet coefficient errors are weighted depending on their remeshing levels to approximate the geometry distortion at a modest computational complexity. A similar approach was adopted in image compression [37].

Fig. 3 shows a vertex at the fine level $(l+1)$ and its eight butterfly neighboring vertices at the coarse level l. Conversely, a vertex at the coarse level is used to predict a number of vertices at the fine level. In Fig. 10, vertex v_i^l at level l is used for the prediction of $v_{\alpha_j}^{l+1}$'s, $v_{\beta_j}^{l+1}$'s, and $v_{\gamma_j}^{l+1}$'s at level $(l+1)$ with weights 1/2, 1/8, and $-1/16$, respectively. In a normal mesh, all vertices except a few base vertices have a valence equal to 6. Thus, if $l \geq 1$, the numbers of $v_{\alpha_j}^{l+1}$'s, $v_{\beta_j}^{l+1}$'s, and $v_{\gamma_j}^{l+1}$'s are equal to 6, 6, and 12, respectively. Since v_i^l is used for the prediction of the associated vertices, the position error of v_i^l propagates to $v_{\alpha_j}^{l+1}$'s, $v_{\beta_j}^{l+1}$'s, and $v_{\gamma_j}^{l+1}$'s, and the propagation effect (including the error of v_i^l itself) can be approximated by a weighting parameter

$$w = 1^2 + 6 \times \left(\frac{1}{2}\right)^2 + 6 \times \left(\frac{1}{8}\right)^2 + 12 \times \left(-\frac{1}{16}\right)^2 = 2.640\,625. \quad (5)$$

The vertex position errors at the lth and $(l+1)$th levels also propagate to associated vertices at the $(l+2)$th level. It can be easily shown that the error propagation of v_i^l to the $(l+1)$th and $(l+2)$th level vertices, including its own error, can be approximated by a weighting parameter w^2, assuming that the errors of neighboring vertices are statistically independent. In a similar way, the error propagation of v_i^l to the whole normal mesh $\mathcal{M}^L$ can be approximated by

$$W_l = w^{L-l} \quad (6)$$

where w is given by (5).

Thus, distortion D_k^m in (3) can be obtained by multiplying the squared error of each wavelet coefficient at the lth level with weight W_l and summing up errors from all levels

$$D_k^m = \sum_{l=1}^{L} \sum_{v_i^l \in \mathcal{C}_k} W_l \left(\psi_i^l - \tilde{\psi}_i^l\right)^2 \quad (7)$$

where W_l is given by (6). Intuitively speaking, wavelet coefficients at a lower level represent low frequency components of an input mesh, and are used in the prediction of higher level coefficients. Thus, wavelet coefficients at a lower level are more important and, therefore, multiplied by a larger weighting parameter in our distortion model.

V. View-Dependent Transmission

In 3-D model streaming applications, the client's viewing position can be exploited to maximize visual quality by allocating a major portion of the available bandwidth to visible parts. Fig. 11 illustrates the principle of view-dependent transmission, where the viewing position lies on σ. The bold line from v_i to v_j depicts the visible region of the object. The server can reduce the required bandwidth by transmitting only the data for the visible region. Following discussion in the previous two sections, we propose a more sophisticated algorithm for view-

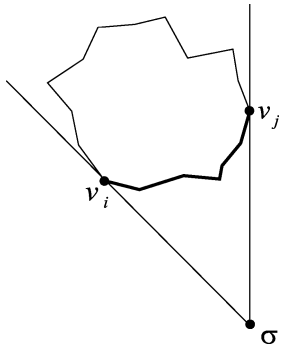


Fig. 11. View-dependent processing with respect to a viewing position.

dependent compression and transmission of normal meshes in this section.

A. Visibility Test and Visibility Priority

The viewing parameter of a client is sent to the server through a feedback channel. With this information, the server performs a visibility test and assigns visibility priorities to vertices. The simplest way to find the visibility is to use the angle between the surface normal vector and the viewing vector. The surface normal test has been used for many graphic processing procedures, such as back-face culling, LOD control, silhouette construction and collision detection [21]–[23], [26], [27]. Note that the surface normal test is valid only if the surface is closed.

In a normal mesh, the visibility of a vertex v can be determined by two vectors, the vertex normal vector $\overrightarrow{N}_v$ and the viewing position vector $\overrightarrow{N}_\sigma$ that begins at the center of object and ends at the viewing position σ. Let $\theta_{v,\sigma}$ denote the angle between $\overrightarrow{N}_v$ and $\overrightarrow{N}_\sigma$, given by

$$\theta_{v,\sigma} = \arccos\left(\frac{\overrightarrow{N}_v \cdot \overrightarrow{N}_\sigma}{\|\overrightarrow{N}_v\| \times \|\overrightarrow{N}_\sigma\|}\right). \tag{8}$$

Vertex v is more visible to the client as $\theta_{v,\sigma}$ approaches 0. It is invisible if $\theta_{v,\sigma}$ is larger than $\pi/2$. In other words, the visibility is proportional to $\cos(\theta_{v,\sigma})$ and is 0 if $\theta_{v,\sigma} > \pi/2$. Since we employ the mean squared error of vertex positions as the geometry distortion measure, the self priority $\mathcal{V}^s(v,\sigma)$ of v from the viewing position σ is defined as

$$\mathcal{V}^s(v,\sigma) = \begin{cases} \cos^2(\theta_{v,\sigma}), & \text{if } 0 \le \theta_{v,\sigma} \le \frac{\pi}{2} \\ 0, & \text{if } \frac{\pi}{2} < \theta_{v,\sigma} \le \pi. \end{cases} \tag{9}$$

It is called the self priority, since it considers only the view-dependent distortion of v itself.

However, as mentioned in Section IV-C, vertices in a normal mesh are related to one another according to the subdivision hierarchy, and the position error of a lower level vertex propagates to higher level vertices. Therefore, the overall visibility priority of a vertex should take into account the error propagation effect as well. With the notation given in Fig. 10, we can obtain the overall visibility priority of each vertex from the highest level to the lowest level recursively via

$$\begin{aligned}\mathcal{V}\left(v_i^l,\sigma\right) = \mathcal{V}^s\left(v_i^l,\sigma\right) &+ \left(\frac{1}{2}\right)^2 \sum_{j=1}^{6} \mathcal{V}\left(v_{\alpha_j}^{l+1},\sigma\right) \\ &+ \left(\frac{1}{8}\right)^2 \sum_{j=1}^{6} \mathcal{V}\left(v_{\beta_j}^{l+1},\sigma\right) \\ &+ \left(-\frac{1}{16}\right)^2 \sum_{j=1}^{12} \mathcal{V}\left(v_{\gamma_j}^{l+1},\sigma\right). \end{aligned} \tag{10}$$

Also, the visibility priority of vertex v_i^L at the highest level is equal to its self priority

$$\mathcal{V}\left(v_i^L,\sigma\right) = \mathcal{V}^s\left(v_i^L,\sigma\right). \tag{11}$$

The visibility priority $\mathcal{V}(v_i^l,\sigma)$ is equal to weighting parameter W_l given by (6), if we assume all vertices are perfectly visible, i.e., $\mathcal{V}^s(v,\sigma) = 1$ for all v's.

It is worthwhile to point out that the view-dependent distortion was defined as the projected geometric error in [19]. In other words, a vertex is assigned a higher priority if the angle between the normal vector and the viewing vector is closer to $\pi/2$. This put more emphasis on faithful reconstruction of the object silhouette. However, in this work, the visibility priorities of vertices are not used for individual vertices but employed to allocate higher bitrates to more visible surface segments. Therefore, instead of using the sine function, we adopt the cosine function in (9).

B. Optimal Truncation of View-Dependent Bitstreams

To support view-dependent transmission effectively, we incorporate the above visibility criterion into the R-D optimized compression system. Let $D_{k,\sigma}^m$ be the view-dependent distortion of the kth reconstructed segment using the first m packets. To compute $D_{k,\sigma}^m$, the visibility priorities of vertices are used instead of the weighting parameters W_l's in (7). That is

$$D_{k,\sigma}^m = \sum_{l=1}^{L} \sum_{v_i^l \in \mathcal{C}_k} \mathcal{V}\left(v_i^l,\sigma\right)\left(\psi_i^l - \tilde{\psi}_i^l\right)^2. \tag{12}$$

The optimal view-dependent transmission is achieved by finding the set of $m(k,\sigma)$'s that minimizes $\sum_{k=1}^{N} D_{k,\sigma}^{m(k,\sigma)}$ subject to the constraint $\sum_{k=1}^{N} R_k^{m(k,\sigma)} \le R_{\text{total}}$. This is basically identical to the R-D optimized compression formulation presented in Section IV-B except that the *view-independent* distortion measure is replaced by the *view-dependent* one. Thus, the optimal set of $m(k,\sigma)$'s can be found using the method described in Section IV-B. By employing the view-dependent distortion measure, the proposed algorithm can allocate the total bit budget more effectively to offer the maximum visual quality to a client.

Since the visibility priorities depend on the viewing position, they should be updated periodically and the R-D optimization algorithm should be performed based on updated visibility priorities. The view-dependent processing for dynamic viewing

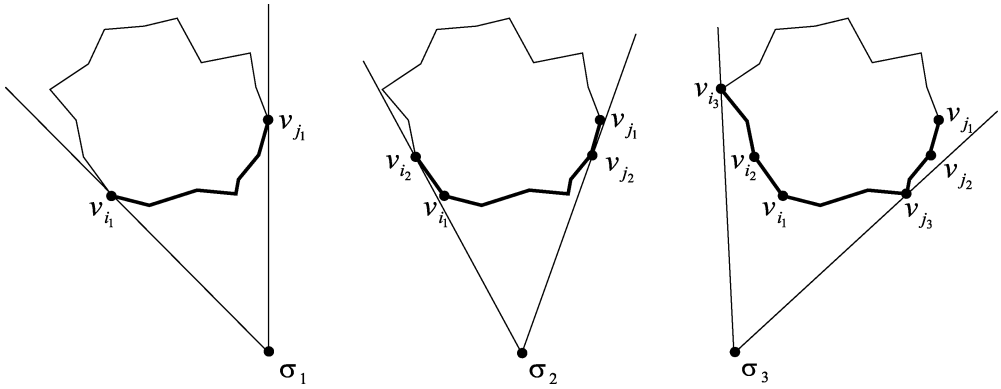


Fig. 12. View-dependent processing with respect to dynamic viewing positions.

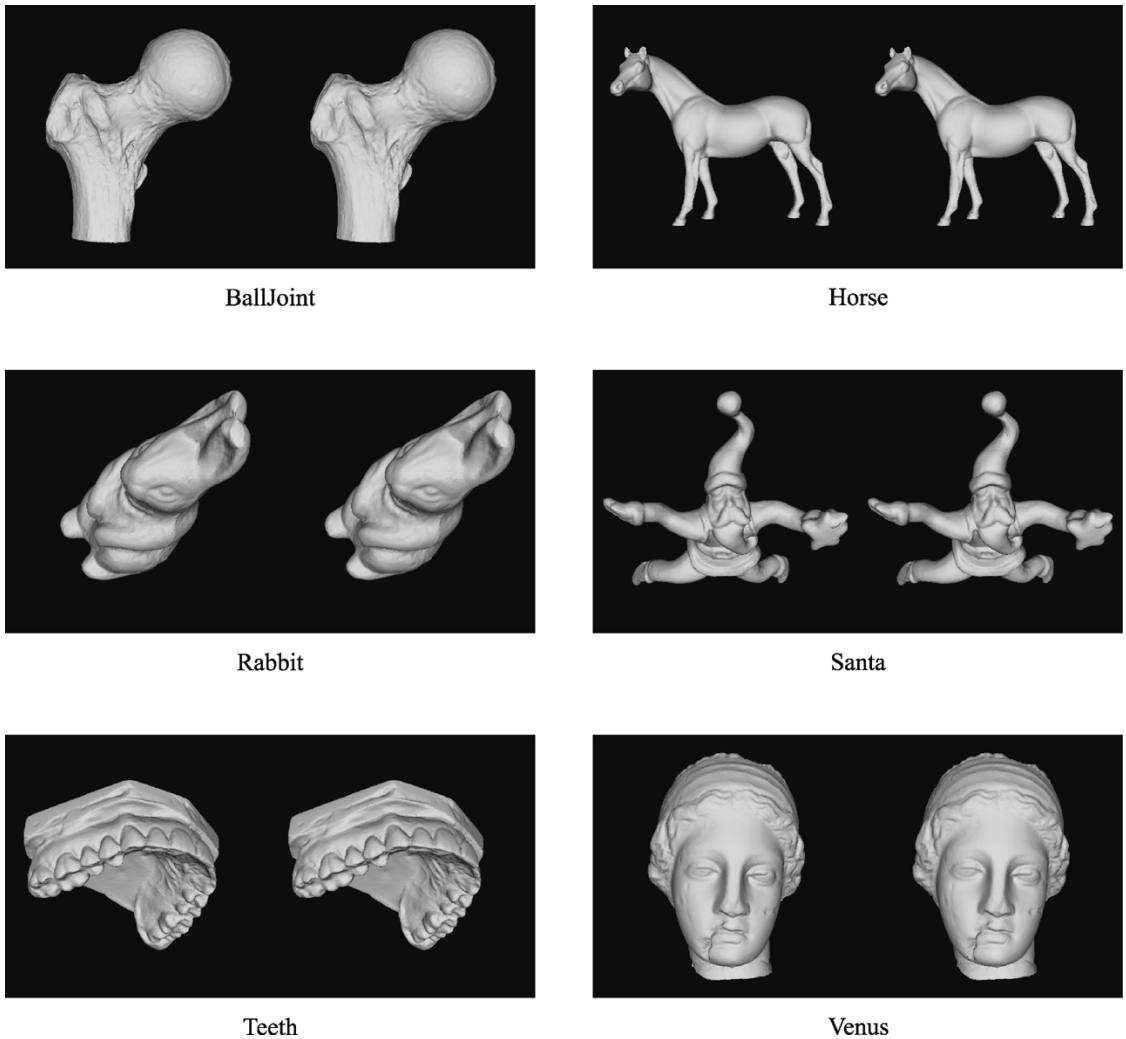


Fig. 13. Comparison of the original irregular mesh (left) and the corresponding normal mesh (right) for six test models with their rendered images.

positions is illustrated in Fig. 12. When an initial viewing position lies on σ_1, the bold curve from v_{i_1} to v_{j_1} is visible. The visibility priorities are initialized to be $\mathcal{V}(v, \sigma_1)$'s, and the R-D optimization is performed to transmit the first $m(k, \sigma_1)$ packets for each segment $\mathcal{C}_k$. When the viewing position approaches toward σ_2, the visible region expands to include v_{i_2}. The server performs the visibility test again, and updates the visibility priorities to $\mathcal{V}(v, \sigma_2)$'s. Then, based on the new information, the R-D optimization algorithm is performed on the remaining packets. Thus, subject to the rate constraint, additional $(m(k, \sigma_2) - m(k, \sigma_1))$ packets are transmitted for each $\mathcal{C}_k$ to maximize the visual quality from σ_2. Similarly, the update of visibility priorities and the R-D optimization are performed based on the next viewing position σ_3.

C. Implementation for Real-Time Applications

The above algorithm computes the visibility priorities of all vertices whenever the viewing position changes. Thus, real-time interaction may be impeded by a high encoder complexity. We develop another implementation for real-time applications, which can reduce the computational complexity significantly at the cost of modest performance degradation.

In this implementation, a visibility priority is assigned to each segment rather than to each vertex. For each segment $\mathcal{C}_k$, a segment normal $\overrightarrow{N}_{\mathcal{C}_k}$ is defined as the sum of vertex normals in the segment and given by

$$\overrightarrow{N}_{\mathcal{C}_k} = \sum_{v_i \in \mathcal{C}_k} \overrightarrow{N}_{v_i}. \tag{13}$$

Then, the visibility priority $\mathcal{V}(\mathcal{C}_k, \sigma)$ of $\mathcal{C}_k$ is defined in a way similar to (9) via

$$\mathcal{V}(\mathcal{C}_k, \sigma) = \begin{cases} \cos^2(\theta_{\mathcal{C}_k,\sigma}), & \text{if } 0 \le \theta_{\mathcal{C}_k,\sigma} \le \frac{\pi}{2} \\ 0, & \text{if } \frac{\pi}{2} < \theta_{\mathcal{C}_k,\sigma} \le \pi \end{cases} \tag{14}$$

where $\theta_{\mathcal{C}_k,\sigma}$ denotes the angle between $\overrightarrow{N}_{\mathcal{C}_k}$ and $\overrightarrow{N}_{\sigma}$.

Then, the view-dependent distortion $D^m_{k,\sigma}$ in (12) is approximated by

$$D^m_{k,\sigma} \simeq \mathcal{V}(\mathcal{C}_k, \sigma) \cdot D^m_k = \mathcal{V}(\mathcal{C}_k, \sigma) \cdot \sum_{l=1}^{L} \sum_{v_i^l \in \mathcal{C}_k} W_l \left(\psi_i^l - \tilde{\psi}_i^l\right)^2 \tag{15}$$

where D^m_k is the view-independent distortion given by (7). In other words, to approximate the view-dependent distortion, the view-independent distortion is weighted by the segment visibility priority. The server can pre-compute the view-independent distortion D^m_k for each pair of k and m. When the viewing position changes, the sever only computes the segment visibility priorities and approximates the view-dependent distortions with (15). Note that this approach requires much less computation, since it does not compute the visibility priorities of all vertices.

VI. Simulation Results

A. Remeshing and R-D Optimized Compression of Normal Meshes

The performance of the proposed algorithm is evaluated with six test models as shown in Fig. 13, where the original irregular mesh and the corresponding normal mesh are presented side-by-side for visual comparison. Furthermore, five levels of the "Venus" normal mesh are shown in Fig. 14. Some properties of these test models are summarized in Table I.

The remeshing distortion in the table is calculated using the METRO algorithm [38]. That is, in order to measure the distortion between two meshes $\mathcal{M}_a$ and $\mathcal{M}_b$, $\mathcal{M}_a$ is scan-converted to yield a sufficient number of sample points. Then, the distance from $\mathcal{M}_a$ to $\mathcal{M}_b$ is given by the root mean of minimum squared distances from the randomly selected sample points on $\mathcal{M}_a$ to $\mathcal{M}_b$. Similarly, the distance from $\mathcal{M}_b$ to $\mathcal{M}_a$ is computed. The distortion is given by the maximum of the two distances, and then normalized by the diagonal of the bounding box for $\mathcal{M}_a$ and $\mathcal{M}_b$.

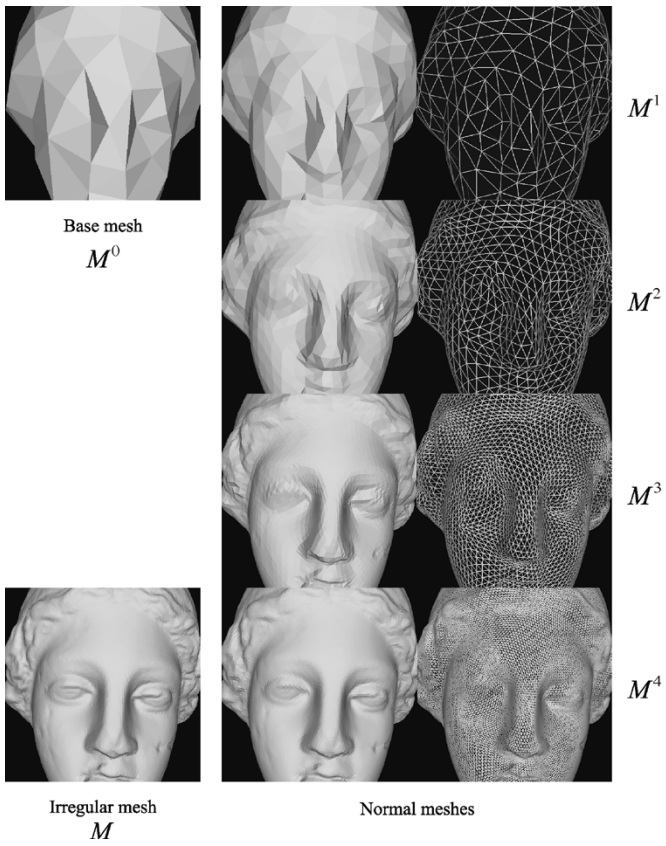


Fig. 14. Representation of the "Venus" model at various levels.

TABLE I
PROPERTIES OF TEST MESH MODELS. # V = NUMBEROF VERTICES IN AN ORIGINAL IRREGULAR MESH $\mathcal{M}$; # T = NUMBER OF TRIANGLES IN $\mathcal{M}$; # BT = NUMBER OF TRIANGLES IN A BASE MESH $\mathcal{M}^0$; # L = MAXIMUM REMESHING LEVEL; # NNV = NUMBER OF NON-NORMAL VERTICES IN A NORMAL MESH $\mathcal{M}^L$

Test model	♯ V	♯ T	♯ BT	♯ L	♯ NNV	Remeshing distortion (10^{-4})
BallJoint	34,267	68,530	140	5	211	1.1095
Horse	48,485	96,966	350	5	1,795	0.7295
Rabbit	67,039	134,074	150	5	84	0.7623
Santa	56,836	113,668	250	5	715	0.7727
Teeth	58,303	116,602	270	5	974	0.9066
Venus	67,173	134,342	300	5	200	0.5120

We compare the performance of the proposed R-D optimized compression algorithm with that of the conventional algorithm [7], which compresses a normal mesh as a whole. For a fair comparison, a base mesh is assumed to be transmitted losslessly in both algorithms. Fig. 15 compares the R-D curves of the proposed algorithm and the conventional algorithm. The unit of the rate is in byte, and the distortion is measured using the METRO algorithm. The conventional algorithm encodes the forest of edge-based trees in an implicit order. Thus, it requires no extra

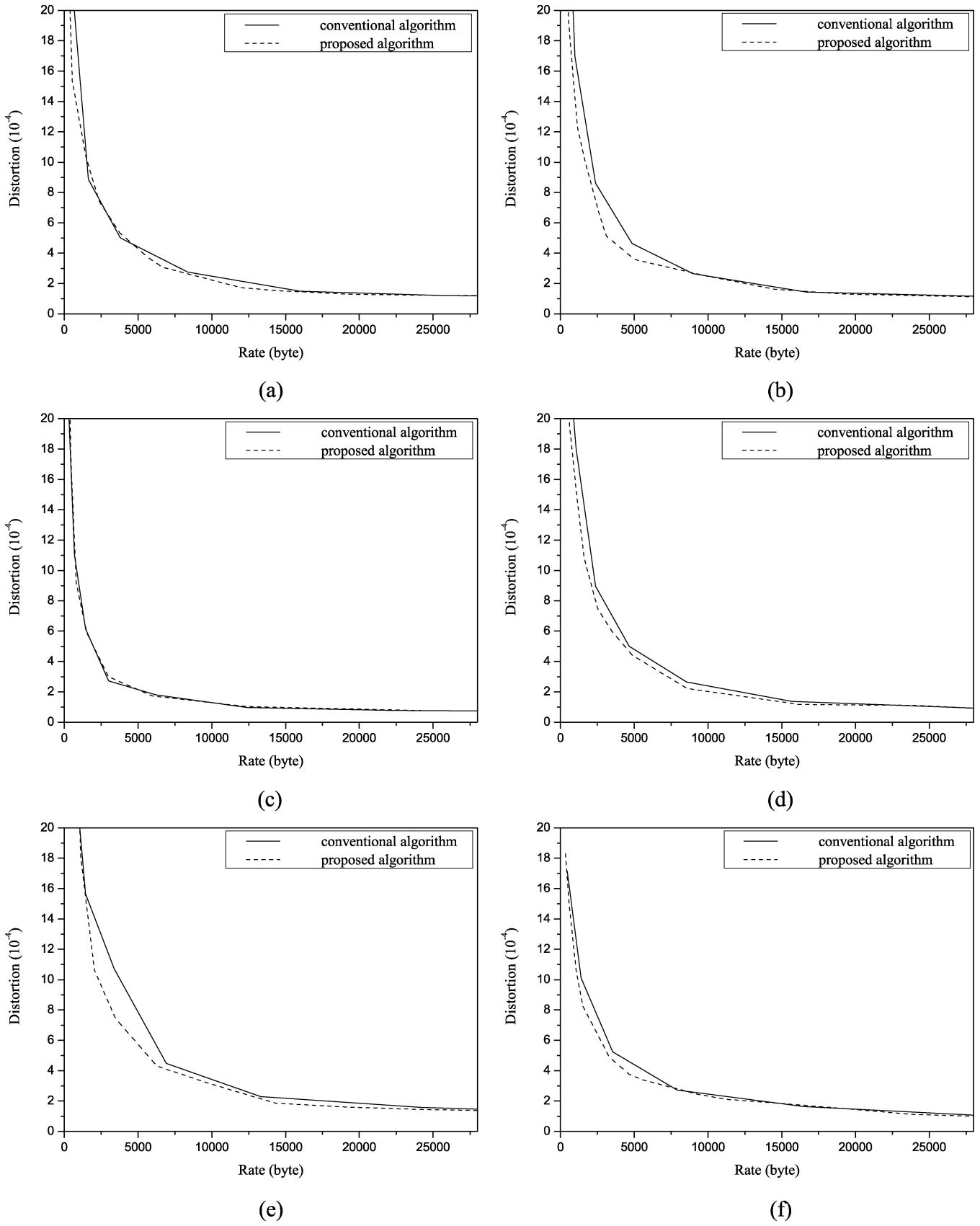


Fig. 15. R-D curves for six test models. (a) BallJoint. (b) Horse. (c) Rabbit. (d) Santa. (e) Teeth. (f) Venus.

information except the SPIHT bitstreams. In contrast, the proposed algorithm partitions the forest into tree segments and assigns a different bitrate to each segment. Therefore, additional data should be transmitted to indicate the bitrate of each segment. These additional data are taken into account in the R-D curves of the proposed algorithm.

As shown in Fig. 15, the proposed algorithm provides a better compression performance than the conventional algorithm for most test models, especially at low bitrates. The "Rabbit" model is relatively smooth, and all of its segments have similar characteristics. Thus, the proposed algorithm does not provide a substantial coding gain over the conventional algorithm. However, for other models, the proposed algorithm allocates a total rate effectively according to the segment characteristics. For example, the conventional algorithm requires 4635 bytes to yield the reconstructed distortion of 5×10^{-4} on the "Horse" model, while the proposed algorithm consumes only 3262 bytes to yield the same distortion.

Notice that even though the proposed algorithm uses the R-D optimization technique, the obtained results are only suboptimal, since the resource is allocated based on the approximate distortion model to alleviate the high computational complexity in Section IV-C.

B. View-Dependent Transmission of Normal Meshes

Next, we investigate the performance of the view-dependent transmission method, when the viewing position is fixed. Fig. 16 shows the rendered images of the reconstructed "Venus" models at various bitrates. The left and right images are obtained using the view-independent and the view-dependent methods, respectively. The two images with similar visual quality are shown at the same row and labeled with the required bitrates. We see that the view-dependent transmission saves more than 50% of the bitrates while providing similar visual quality by allocating a main portion of the transmission bandwidth to visible segments.

In the above test, the conventional tree in Fig. 5 is employed. The modified trees in Fig. 8 provide almost the same R-D performance as the conventional tree. However, in the case of view-dependent transmission, the modified trees reconstruct visible parts more reliably than the conventional one, since they localize segments more compactly. The performance of the three edge-based trees in the view-dependent transmission case is compared in Fig. 17. The upper and lower rows render local parts of the "BallJoint" and "Santa" models, respectively. The bitrates for the conventional, scissor, and arrow trees are 2836, 2844, and 2849 bytes for the "BallJoint" model, and 3770, 3793, and 3814 bytes for the "Santa" model, respectively. It is observed that the original shapes are reconstructed more faithfully using the modified trees.

Finally, we test the performance of the view-dependent transmission algorithm when the viewing position varies dynamically. In this test, the fast algorithm described in Section V-C is used to approximate the view-dependent distortion to alleviate the computational burden of the server, and the conventional edge-based tree is used. Fig. 18 shows the reconstructed models, when the viewing position rotates with an angular speed $\pi/3$ (rad/sec) around the "Venus" model from the right face to the left face. The upper (or lower) row renders a sequence of the front (or rear) part of the model observed. The required bitrates are 3734, 5643, 6601, and 7100 bytes, from left to right, respectively. We see that good visual quality is offered to the client by allocating higher bitrates to the front part, while details of the rear part are reconstructed gradually by including previously visible regions. These simulation results indicate that the proposed algorithm is a promising technique for interactive 3-D data transmission.

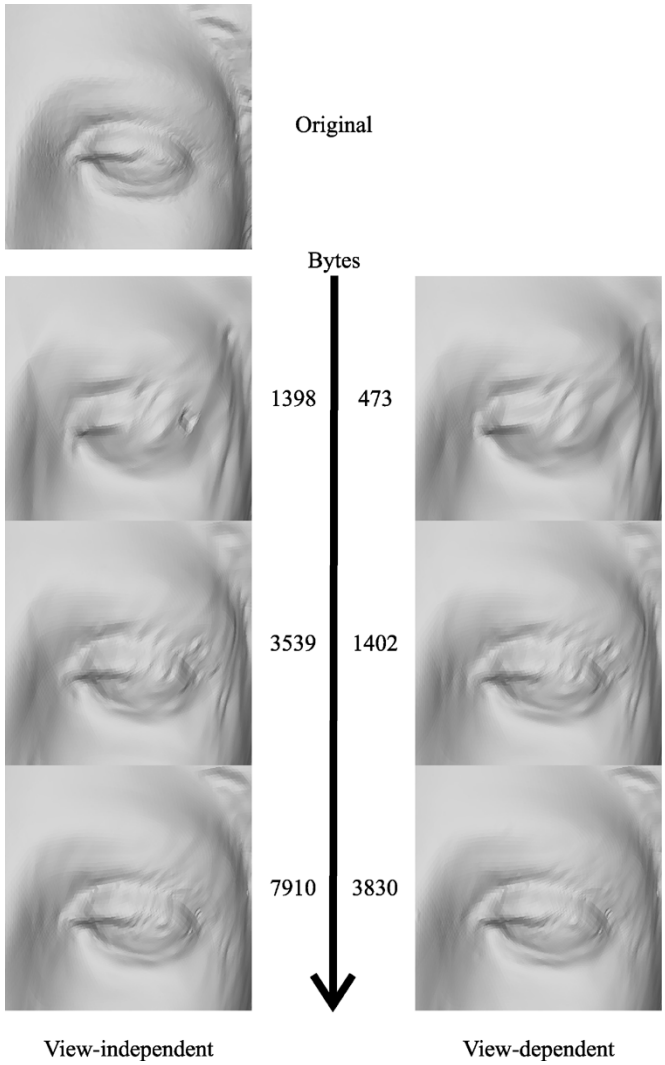


Fig. 16. Comparison of the reconstructed "Venus" models with view-independent and view-dependent methods.

VII. Conclusion and Future Work

A unified framework to achieve the R-D optimized compression and view-dependent transmission of 3-D normal meshes was proposed in this work. The proposed algorithm partitions a normal mesh into disjoint segments, and allocates an optimal bitrate to each segment to minimize the overall distortion. We developed a geometry distortion model to reduce the computational complexity of the optimization process. It was also shown that the visibility priorities of vertices or segments can be incorporated in the distortion model to support view-dependent transmission effectively. Simulation results demonstrated that the proposed algorithm is an efficient technique for 3-D data compression and streaming.

As an extension of the current work, some future research can be performed in the following areas. First, we may consider the region-of-interest coding and transmission. In this work, view-independent and view-dependent distortion models are used in the R-D optimization procedure. More general criteria can be

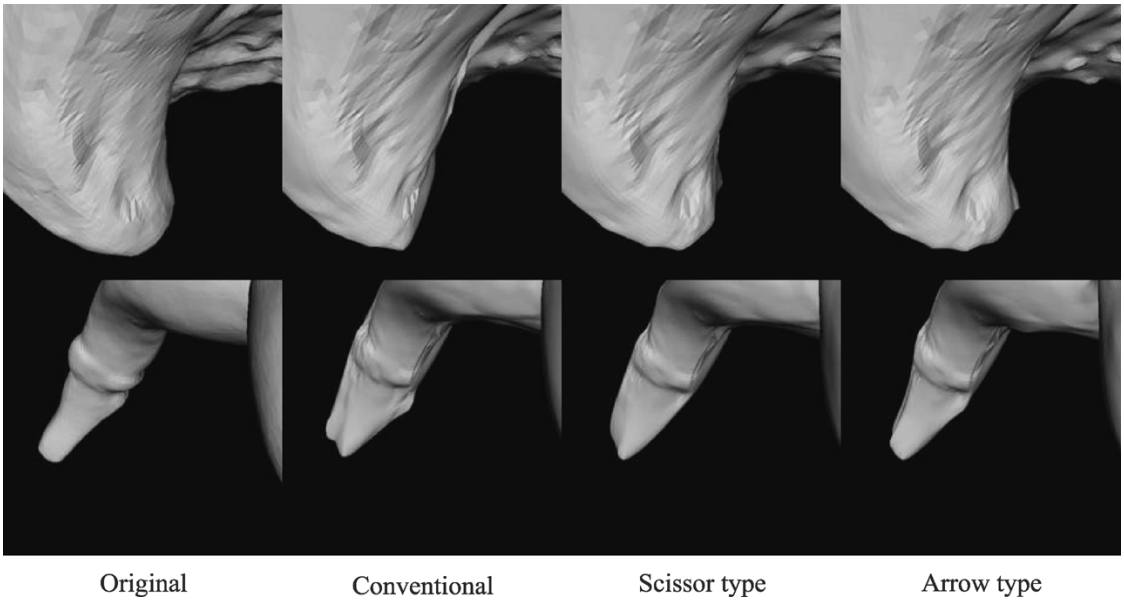


Fig. 17. Comparison of three edge-based trees in the view-dependent transmission, where the upper and the lower rows demonstrate the "BallJoint" and the "Santa" models, respectively.

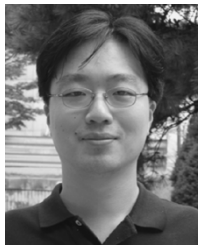
Fig. 18. Reconstructed models with dynamic viewing positions, where the upper and lower rows show the front and rear parts of the "Venus" model, respectively.

used to enable various interactive applications. For example, a client can inform the server of its region-of-interest. Then, the server can find the set of segments which cover the region-of-interest, and then allocate the bit budget only to this set of segments. Second, this framework can be used in the error-resilient coding of 3-D models. To transmit 3-D data over error-prone channels, compressed bitstreams should be resilient to transmission errors. There have been some work on the error-resilient coding of irregular meshes, e.g., [39]. Based on the edge-based tree segmentation, we may modify the proposed algorithm to improve the robustness of the compressed bitstreams for normal meshes.

REFERENCES

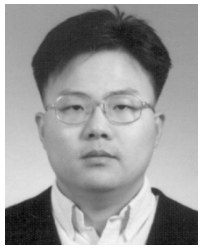
[1] H. Hoppe, "Progressive meshs," in *Proc. SIGGRAPH*, Aug. 1996, pp. 99–108.
[2] J. Li and C.-C. J. Kuo, "Progressive coding of 3-D graphics models," *Proc. IEEE*, vol. 86, no. 6, pp. 1052–1063, Jun. 1998.
[3] D. Cohen-Or, D. Levin, and O. Remez, "Progressive compression of arbitrary triangular meshes," in *Proc. Visualization'99*, 1999, pp. 67–72.
[4] R. Pajarola and J. Rossignac, "Compressed progressive meshs," *IEEE Trans. Vis. Comput. Graphics*, vol. 6, no. 1, pp. 79–93, Jan.–Mar. 2000.
[5] F. Morán and N. García, "Hierarchical coding of 3-D models with subdivision surfaces," in *Proc. ICIP*, vol. 2, 2000, pp. 911–914.
[6] A. Khodakovsky, P. Schröder, and W. Sweldens, "Progressive geometry compression," in *Proc. SIGGRAPH*, 2000, pp. 271–278.
[7] A. Khodakovsky and I. Guskov, "Normal mesh compression," in *Geometric Modeling for Scientific Visualization*. Berlin, Germany: Springer-Verlag, 2002.
[8] A. Said and W. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 243–250, Mar. 1996.
[9] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle, "Multiresolution analysis of arbitrary meshes," in *Proc. SIGGRAPH*, 1995, pp. 173–182.
[10] A. W. F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin, "MAPS: multiresolution adaptive parameterization of surfaces," in *Proc. SIGGRAPH*, 1998, pp. 95–104.
[11] I. Guskov, K. Vidimce, W. Sweldens, and P. Schröder, "Normal meshes," in *Proc. SIGGRAPH*, 2000, pp. 95–102.
[12] T. Duchamp, A. Certain, A. DeRose, and W. Stuetzle, "Hierarchical computation of PL harmonic embeddings," University of Washington, Tech. Rep., Seattle, Jul. 1997.
[13] D. Zorin and P. Schröder, "Subdivision for modeling and animation," presented at the *SIGGRAPH Course Notes*, 1999.

[14] N. Dyn, D. Levin, and J. A. Gregory, "A butterfly subdivision scheme for surface interpolation with tension control," *ACM Trans. Graphics*, vol. 9, no. 2, pp. 160–169, 1990.
[15] M. Lounsbery, T. DeRose, and J. Warren, "Multiresolution analysis for surfaces of arbitrary topological type," *ACM Trans. Graphics*, vol. 16, no. 1, pp. 34–73, Jan. 1997.
[16] P. Schröder and W. Sweldens, "Spherical wavelets: efficiently representing functions on the sphere," in *Proc. SIGGRAPH*, 1995, pp. 161–172.
[17] W. Sweldens and P. Schröder, "Building your own wavelets at home," in *SIGGRAPH Course Notes*, 1996, pp. 15–87.
[18] K. Kolarov and W. Lynch, "Compression of functions defined on surfaces of 3-D objects," in *Proc. Data Compression Conf.*, 1997, pp. 281–291.
[19] P. Lindstrom *et al.*, "Real-time continuous level of detail rendering of height fields," in *Proc. SIGGRAPH*, 1996, pp. 109–118.
[20] J. C. Xia and A. Varshney, "Dynamic view-dependent simplification for polygonal models," in *Proc. Visualization*, 1996, pp. 335–344.
[21] H. Hoppe, "View-dependent refinement of progressive meshes," in *Proc. SIGGRAPH*, 1997, pp. 189–198.
[22] D. Luebke and C. Efikson, "View-dependent simplification of arbitrary polygonal environments," in *Proc. SIGGRAPH*, 1997, pp. 199–208.
[23] D. I. Azuma, B. Curless, T. Duchamp, D. H. Salesin, W. Stuetzle, and D. N. Wood, "View-dependent refinement of multiresolution meshes with subdivision connectivity," Univ. Washington, Seattle, Tech. Rep. UW-CSE-2001-10-02, Oct. 2001.
[24] P. Alliez, N. Laurent, H. Sanson, and F. Schmitt, "Efficient view-dependent refinement of 3-D meshes using $\sqrt{3}$-subdivision," *Vis. Comput.*, vol. 19, no. 4, pp. 205–221, 2003.
[25] D. To, R. Lau, and M. Green, "A method for progressive and selective transmission of multiresolution models," *ACM Virtual Reality Softw. Technol.*, pp. 88–95, 1999.
[26] R. Southern *et al.*, "A stateless client for progressive view-dependent transmission," *ACM Web3-D*, pp. 43–50, 2001.
[27] S. Yang, C.-S. Kim, and C.-C. J. Kuo, "A progressive view-dependent technique for interactive 3-D mesh transmission," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 11, pp. 1249–1264, Nov. 2004.
[28] M. Grabner and C. Zach, "Adaptive quantization with error bounds for compressed view-dependent multiresolution meshes," in *Proc. EUROGRAPHICS*, 2003, pp. 77–82.
[29] P. Gioia, O. Aubault, and C. Bouville, "Real-time reconstruction of wavelet-encoded meshes for view-dependent transmission and visualization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 7, pp. 1009–1020, Jul. 2004.
[30] G. Lafruit *et al.*, "View-dependent scalable texture streaming in 3-D QoS MPEG-4 visual texture coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 7, pp. 1021–1031, Jul. 2004.
[31] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proc. SIGGRAPH*, 1996, pp. 209–216.
[32] G. Taubin and J. Rossignac, "Geometric compression throught topological surgery," *ACM Trans. Graphics*, vol. 17, no. 2, pp. 84–115, Apr. 1998.
[33] C. Touma and C. Gotsman, "Triangle mesh compression," in *Proc. Graphics Interface*, Jun. 1998, pp. 26–34.
[34] A. P. Mangan and R. T. Whitaker, "Partitioning 3-D surface meshes using watershed segmentation," *IEEE Trans. Vis. Comput. Graphics*, vol. 5, no. 4, pp. 308–321, Oct.–Dec. 1999.
[35] E. Zuckerberger, A. Tal, and S. Shlafman, "Polyhedral surface decomposition with applications," *Comput. Graph.*, vol. 26, no. 5, pp. 733–743, Oct. 2002.
[36] *JPEG2000 Verification Model 6.0 (Technical Description)*, ISO/IEC JTC1/SC29/WG1 WG1N1575, Jan. 2000.
[37] G. Davis and A. Nosratinia, "Wavelet-based image coding: an overview," *Appl. Comput. Contr., Signals, Circuits*, vol. 1, no. 1, pp. 436–440, Spring 1998.
[38] P. Cignoni, C. Rocchini, and R. Scopigno, "METRO: measuring error on simplified surfaces," *Comput. Graph. Forum*, vol. 17, no. 2, pp. 167–174, 1998.
[39] Z. Yan, S. Kumar, and C.-C. J. Kuo, "Error-resilient coding of 3-D graphic models via adaptive mesh segmentation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 7, pp. 860–873, Jul. 2001.



Jae-Young Sim (S'02) received the B.S. and M.S. degrees in electrical engineering from Seoul National University, Seoul, Korea, in 1999 and 2001, respectively, where he is currently working toward the Ph.D. degree in electrical engineering and computer science.

His research topics include 3-D multimedia representation, compression, and transmission.



Chang-Su Kim (S'95–M'01) received the B.S. and M.S. degrees in control and instrumentation engineering and the Ph.D. degree in electrical engineering from Seoul National University (SNU), Seoul, Korea, in 1994, 1996, and 2000, respectively.

From 2000 to 2001, he was a Visiting Scholar with the Signal and Image Processing Institute, University of Southern California, Los Angeles, and a Consultant for InterVideo Inc., Los Angeles. From 2001 to 2003, he was a Postdoctoral Researcher with the School of Electrical Engineering, SNU. In August 2003, he joined the Department of Information Engineering, The Chinese University of Hong Kong as an Assistant Professor. His research topics include video and 3-D graphics processing and multimedia communications. He has published more than 70 technical papers in international conferences and journals.

C.-C. Jay Kuo (S'83–M'86–SM'92–F'99) received the B.S. degree in electrical engineering from the National Taiwan University, Taipei, in 1980 and the M.S. and Ph.D. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 1985 and 1987, respectively.

He was Computational and Applied Mathematics (CAM) Research Assistant Professor in the Department of Mathematics, University of California, Los Angeles, from October 1987 to December 1988. Since January 1989, he has been with the Department of Electrical Engineering-Systems and the Signal and Image Processing Institute, University of Southern California, Los Angeles, where he currently has a joint appointment as Professor of Electrical Engineering and Mathematics. He has guided about 60 students to their Ph.D. degrees and supervised 15 Postdoctoral Research Fellows. He is coauthor of seven books and more than 700 technical publications in international conferences and journals. His research interests are in the areas of digital signal and image processing, audio and video coding, multimedia communication technologies and delivery protocols, and embedded system design.

Dr. Kuo is Editor-in-Chief for the *Journal of Visual Communication and Image Representation*, and Editor for the *Journal of Information Science and Engineering* and the EURASIP *Journal of Applied Signal Processing*. He is also on the Editorial Board of the IEEE Signal Processing Magazine. He served as Associate Editorfor IEEE TRANSACTIONS ON IMAGE PROCESSING in 1995–1998, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY in 1995–1997 and IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING in 2001–2003.

Sang-Uk Lee (S'75–M'80–SM'99) received the B.S. degree from Seoul National University, Seoul, Korea, in 1973, the M.S. degree from Iowa State University, Ames, in 1976, and the Ph.D. degree from the University of Southern California, Los Angeles, in 1980, all in electrical engineering.

From 1980 to 1981, he was with General Electric Company, Lynchburg, VA, working on the development of digital mobile radio. From 1981 to 1983, he was a Member of Technical Staff, M/A-COM Research Center, Rockville, MD. In 1983, he joined the Department of Control and Instrumentation Engineering, Seoul National University, as an Assistant Professor, where he is now a Professor of the School of Electrical Engineering. He is also affiliated with the Automation and Systems Research Institute and the Institute of New Media and Communications, both at Seoul National University. His current research interests are in the areas of image and video signal processing, digital communication, and computer vision. He served as an Editor-in-Chief for the *Transactions of the Korean Institute of Communication Science* from 1994 to 1996.

Dr. Lee is currently an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and is on the Editorial Board of the *Journal of Visual Communication and Image Representation* and the EURASIP *Journal of Applied Signal Processing*. He is a Member of Phi Kappa Phi.