Predictive Compression of Geometry, Color and Normal Data of 3-D Mesh Models

Jeong-Hwan Ahn, Chang-Su Kim, and Yo-Sung Ho

***Abstract*—Predictive compression algorithms for geometry, color and normal data of three-dimensional (3-D) mesh models are proposed in this work. In order to eliminate redundancies in geometry data, we predict each vertex position by exploiting the position and angle information in neighboring triangles. To compress color data, we propose a mapping table scheme that compresses frequently recurring colors efficiently. For normal data, we propose an average predictor and a 6-4 subdivision quantizer to improve coding gain. Simulation results demonstrate that the proposed algorithm provides better performance than the MPEG-4 standard for 3-D mesh model coding (3-DMC).**

***Index Terms*—Colors, geometry, MPEG-4, normal vectors, three-dimensional mesh model coding (3-DMC), VRML.**

I. Introduction

In RECENT days, three-dimensional (3-D) models are used in various applications, such as computer graphics and synthetic imaging systems. In particular, polygonal meshes are frequently used to represent 3-D object surfaces. A 3-D mesh model is represented by its connectivity, geometry and attribute data. The connectivity data describe connectivity relations among vertices. The geometry data specify vertex locations. The attribute data generally consist of colors, normal vectors and texture coordinates, which are needed to paint and shade the model. Since transmission bandwidth and storage capacity are limited in most applications, we need to develop an efficient mesh compression scheme. Geometry and attribute data are specified by floating-point numbers, whereas connectivity data are represented by integer indices. Therefore, the compression of geometry and attribute data is more effective in reducing the required bit rates for 3-D models.

Deering [1] proposed a pioneering work on 3-D mesh compression, based on generalized triangle strips. Taubin and Rossignac [2] introduced the topological surgery algorithm. Geometry and color data are predictively encoded along the vertex spanning tree, and normal vectors are quantized in the spherical coordinate system using the 6-4 subdivision. Choi *et al.* [3] proposed a predictive two-stage quantization scheme to encode geometry prediction errors in the topological surgery algorithm more effectively. Ahn and Ho [4] proposed a geometry compression algorithm in the spherical coordinate system. Li *et al.* [5] proposed an embedded quantization scheme for geometry data. Gumhold and Straßer [6] proposed a triangle conquest approach for connectivity coding. Touma and Gotsman [7] proposed an efficient encoding scheme for geometry data based on the parallelogram prediction. Bossen applied this parallelogram prediction rule to the topological surgery algorithm. His approach has been adopted by the MPEG-4 standard for 3-D mesh model coding (3-DMC) [8]. Also, several progressive mesh compression schemes have been recently proposed, *i.e.*, in [9] and [10]. More detailed review of 3-D mesh compression can be found in [11].

In this paper, we propose predictive compression algorithms for geometry, colors and normal data. Although conventional works applied the same predictor and quantizer to geometry, color and normal vectors, these data have different characteristics. Therefore, predictors and quantizers should be differently designed for geometry, color and normal data. For geometry compression, we propose a prediction rule that can exploit all the position and angle information in previously traversed triangles. For color compression, we propose a mapping table scheme that can encode frequently recurring colors efficiently. For normal vector compression, we propose an average predictor and a 6-4 subdivision quantizer to improve prediction accuracy. Simulation results demonstrate the proposed algorithms provide improved coding performance over the MPEG-4 3-DMC standard.

In Section II, we review the background of 3-D mesh compression systems and describe the vertex layer traversal. The geometry, color and normal compression algorithms are proposed in Sections III–V, respectively. Experimental results are discussed in Section VI. Finally, the paper is concluded in Section VII.

II. Vertex Layer Traversal

In this work, we adopt the DPCM structure [12] to compress 3-D meshes. Note that the DPCM structure has been employed in various works [3], [4], [13]–[15]. The encoder predicts the geometry or attribute of the current vertex using the previously reconstructed vertex values. The encoder then quantizes the prediction error and compresses the output index with an entropy coder. The current vertex attribute is then reconstructed by adding the quantized error to the prediction value, and the reconstructed value is employed in the prediction of subsequent vertex values. The prediction plays an important role in DPCM since it reduces data redundancy to achieve coding gain. We propose effective prediction algorithms for geometry, color, and normal data.

Manuscript received October 20, 2003; revised July 5, 2005. This paper was recommended by Associate Editor H. Shum.

J.-H. Ahn is with the Samsung Advanced Institute of Technology, Kyungki 440-600, Korea (e-mail: jeonghwan.ahn@samsung.com).

C.-S. Kim is with the Department of Electronics and Computer Engineering, Korea University, Seoul 136-713, Korea (e-mail: cskim@ieee.org).

Y.-S. Ho is with the Department of Information and Communications, Gwangju Institute of Science and Technology (GIST), Gwangju 500-712, Korea (e-mail: hoyo@gist.ac.kr).

Digital Object Identifier 10.1109/TCSVT.2005.861945

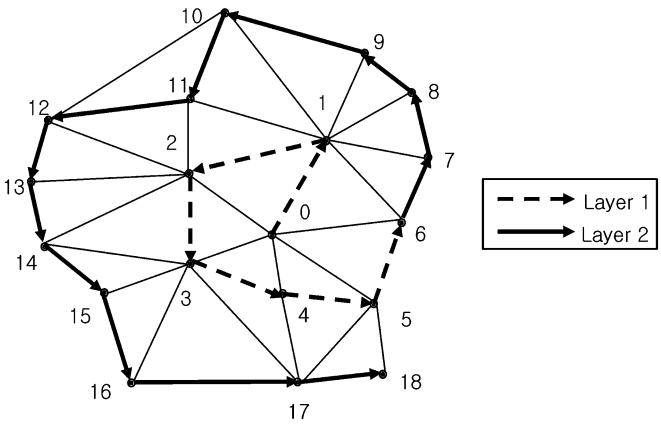


Fig. 1. Vertex layer traversal.

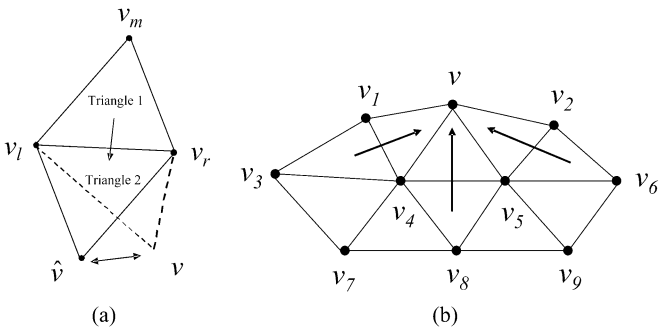


Fig. 2. (a) Parallelogram prediction. (b) Average parallelogram prediction.

The predictive coding of geometry and attribute data requires an ordering of vertices. In this paper, we arrange vertices based on a vertex layer traversal algorithm, and use the ordering in the prediction of geometry, color and normal data. The vertex layer traversal facilitates efficient prediction, while maintaining a simple data structure. It is carried out as follows. We select the root vertex that has the farthest distance from the center of gravity of the input model. Using the root vertex as a pivot point, we visit vertices of topological distance 1 in the counterclockwise direction. Then, we traverse vertices of topological distance 2, and so on. In other words, vertices are traversed in the breadth-first search (BFS) order [16]. During the traversal, a queue Q of vertices is maintained to record the vertex indices, indicating the traversal order. Since the proceeding vertices are pushed into Q before the current vertex, we can identify which vertices are previously traversed or not by checking the vertex indices. Fig. 1 shows an example of the vertex layer traversal.

III. Geometry Compression

The geometry compression has four stages: normalization, prediction, quantization and entropy coding. In the normalization phase, we obtain the tightest bounding box containing the input model and normalize the vertex coordinates so that they lie within the unit cube $[0,1]\times[0,1]\times[0,1]$. The normalization is necessary to limit the dynamic range of prediction errors.

Fig. 2(a) illustrates the parallelogram prediction [7], which predicts the position of a vertex $v=(x,y,z)$. When we pass from triangle 1 to triangle 2, the vertices v_l, v_r and v_m are already encoded. The prediction $\hat{v}$ of v is obtained as $v_l+v_r-v_m$,

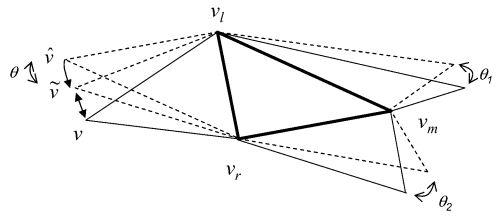


Fig. 3. Dihedral angle estimation.

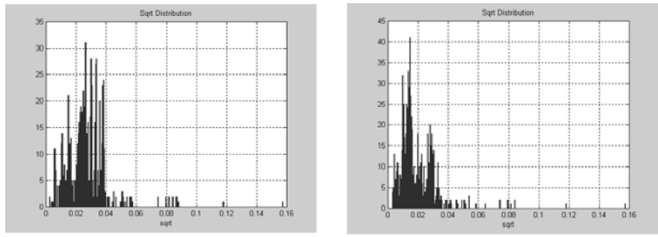


Fig. 4. Comparison of prediction errors for the model EIGHT. (a) Conventional parallelogram prediction. (b) Proposed algorithm.

such that $\hat{v}$ and its three predecessors form a parallelogram and lie on a plane. However, the conventional parallelogram prediction uses only three previously traversed vertices in a single adjacent triangle, and thus the predicted vertex can be located at a biased position. Moreover, vertices on a curved surface may not be predicted effectively, since each parallelogram is assumed to lie on the same plane.

To improve the prediction performance, the proposed algorithm uses all the neighboring triangles that precede the current vertex. Suppose that the current vertex v can be predicted from several neighboring triangles using the parallelogram rule. Then, a prediction is computed for each of these triangles, and the final prediction is the average of all the parallelogram prediction vectors, given by

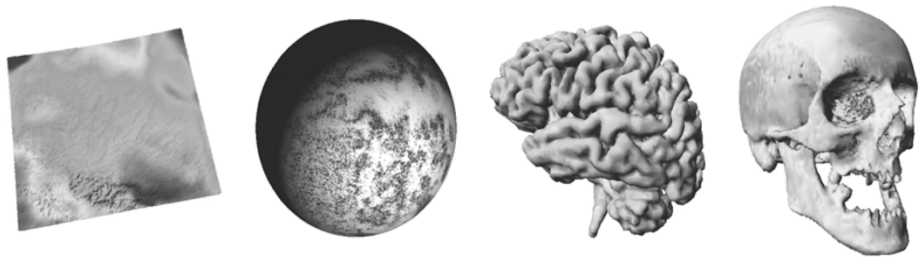
$$\hat{v}=\frac{1}{N}\sum_{k=1}^{N}\hat{v}_k \tag{1}$$

where N is the number of preceding triangles, and $\hat{v}_k$ denotes the parallelogram prediction vector corresponding to the kth preceding triangle. For example, assume that a vertex has three preceding triangles as shown in Fig. 2(b). Then, the predicted vertex $\hat{v}$ is given by

$$\hat{v}=\frac{1}{3}\{(v_1+v_4-v_3)+(v_4+v_5-v_8)+(v_2+v_5-v_6)\}.$$

To enhance the prediction performance further, the proposed algorithm estimates the signed dihedral angle θ between two triangles. Although the concept of the dihedral angle prediction was first mentioned in [7], the proposed algorithm uses all the neighboring dihedral angles in the prediction. As shown in Fig. 3, we estimate the dihedral angle θ using two angles θ_1 and θ_2 via

$$\theta=\begin{cases}\theta_1, & \text{if only } \theta_1 \text{ is available}\\ \theta_2, & \text{if only } \theta_2 \text{ is available}\\ \frac{(\theta_1+\theta_2)}{2}, & \text{if both } \theta_1 \text{ and } \theta_2 \text{ are available.}\end{cases}$$



(a) (b) (c) (d)

Fig. 5. Test models for color compression. Each vertex is associated with a (R, G, B) color vector, and each color component is represented with a 32-bit float number. (a) HUMIDITY. (b) SPHERE. (c) BRAIN. (d) C-SKULL.

Index	1	2	3	4	5	6	7	8	9	10
Color	4	7	6	5	4	6	4	6	8	4

(a) Input sequence

Index	Color Index	Differential Color
1	0	4
2	0	3 $(=7-4)$
3	0	-1 $(=6-7)$
4	0	-1 $(=5-6)$
5	4 $(=5-1)$	-
6	3 $(=6-3)$	-
7	2 $(=7-5)$	-
8	2 $(=8-6)$	-
9	0	2 $(=8-6)$
10	3 $(=10-7)$	-

(b) Coded data

Fig. 6. Example of the mapping table encoding.

Then, we rotate the parallelogram prediction vector $\hat{v}$ to $\tilde{v}$ by the estimated angle θ. If a vertex has more than one preceding triangles, the predicted vertex $\acute{v}$ is given by

$$\acute{v} = \frac{1}{N}\sum_{k=1}^{N} \tilde{v}_k \tag{2}$$

where $\tilde{v}_k$ is the kth rotated parallelogram prediction vector.

Fig. 4 compares the distributions of prediction error Δv for the model EIGHT. The prediction error Δv is defined as $\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}$, where Δx, Δy, and Δz denote the three components of the error vector $v - \acute{v}$. Fig. 4(a) is the result of the conventional parallelogram prediction and Fig. 4(b) is the result of the proposed algorithm. We can observe that the proposed algorithm has more concentrated prediction errors near zero than the conventional prediction. Thus, the empirical entropy of the prediction error is 5.29 bits for the proposed algorithm, while it is 5.64 bits for the conventional prediction. The gain is due to that the proposed algorithm exploits the information in all neighboring triangles that precede the current vertex, while the conventional parallelogram prediction uses only three preceding vertices.

After the prediction, each prediction error is uniformly quantized. The dynamic range of the prediction error is limited to $[-1,1] \times [-1,1] \times [-1,1]$. Thus, we employ the uniform midtread quantizer of M levels with step size $\Delta = 2/(M-1)$, and encode the quantization index with the QM coder [17].

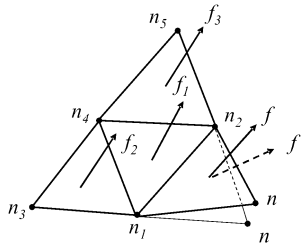


Fig. 7. Average prediction rule for normal vectors.

IV. Color Compression

With geometry data, color data play an important role in human perception of objects. Color information can be attached to vertices, faces or corners in 3-D mesh models. In this work, we propose a mapping table approach to encode color data with per-vertex binding. 3-D mesh models can be obtained by a laser scanning system or generated using an authoring tool. In these cases, some parts are often painted with the same color due to the limited precision of the system. For instance, the HUMIDITY model in Fig. 5 has 39 072 vertices painted with only 401 different colors. The proposed mapping table scheme can provide a high coding gain for such 3-D models that have frequently recurring colors.

In the mapping table, a color value is encoded only once when it appears first during the vertex layer traversal. When the same color recurs on another vertex later, the encoder records the index of the last vertex with the same color, instead of the

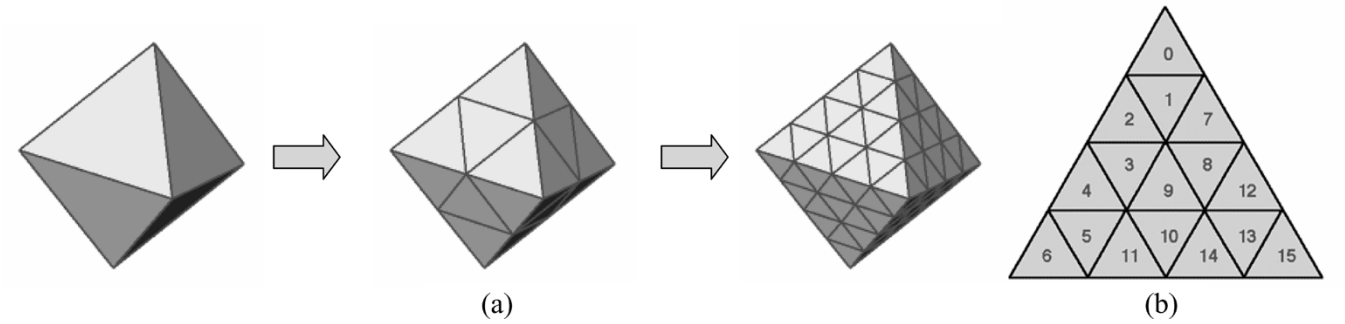


Fig. 8. Normal vector representation in MPEG-4 3-DMC. (a) Three subdivisions of the base octahedron. (b) Index pattern.

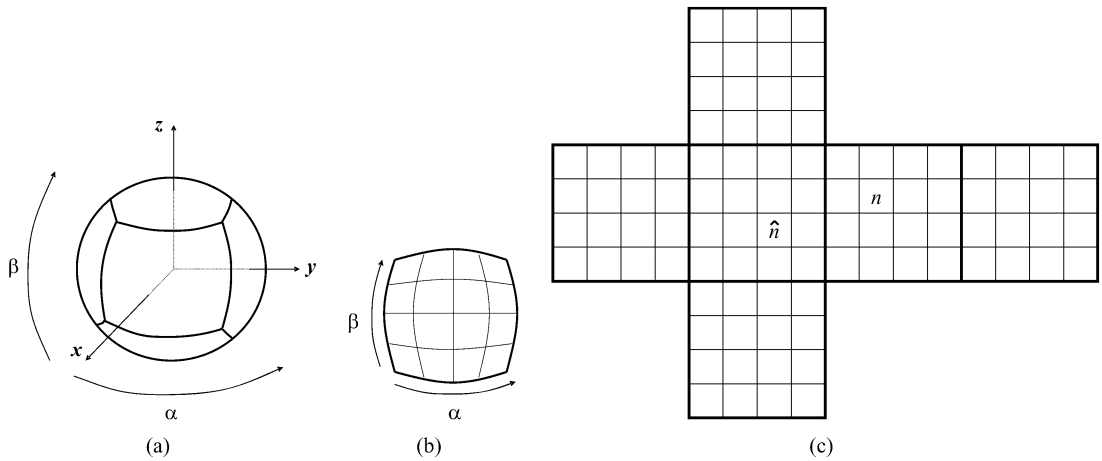


Fig. 9. The 6-4 subdivision. (a) Sextant partitioning. (b) Further subdivision of each sextant. (c) Unfolding of the sextants.

color itself. The mapping table reduces the bit rate for color data significantly since it avoids the duplicated encoding of the same color.

Let us encode the sequence in Fig. 6(a). The coded data for this sequence are tabled in Fig. 6(b). Initially, the mapping table is empty. Since the first vertex has color value 4, the encoder records color index 0, which informs the decoder that color value 4 is first used. Since the second vertex has color value 7, which is not used yet, the encoder records color index 0 and the differential color 3 that is the difference between the current color and the preceding color. The information for the third and fourth vertices is encoded in a similar way. However, since the fifth vertex has color value 4 that is already used in the first vertex, the encoder records color index 4, which is the index difference between the current vertex and the last vertex that has the same color. Note that the encoder need not record the color value, since the decoder can reconstruct the color value using the color index. This encoding operation continues in the vertex order traversal order till the end of the sequence.

The bit rate requirement of the mapping table scheme depends on the input model. However, it was found experimentally that, to achieve an acceptable image quality, the mapping table scheme requires a bit rate higher than 10 bits per color (bpc) in general. Therefore, if the given bit rate is lower than 10 bpc, we employ a DPCM scheme with a large quantizer step size instead.

In the DPCM scheme, we use the first-order predictor with coefficient 1. In other words, each color is simply predicted by only one preceding color. Generally, between two adjacent vertices, the color correlation is higher than the position correlation. Therefore, although higher order predictors, such as the parallelogram predictor, are more efficient in removing redundancies in geometry data, the first-order predictor is sufficient for the color prediction. As in the geometry encoder, the prediction error is uniformly quantized and encoded with the QM coder.

V. Normal Vector Compression

We propose a compression algorithm for normal vector data with per-vertex binding. We only consider normal vectors of length 1, and any normal vector on the unit sphere can be represented by two parameters, e.g., the longitude and the latitude in the spherical coordinate system [18]. In order to improve the coding gain, we predict each normal vector using the average prediction scheme, and quantize the prediction error using the 6-4 subdivision scheme.

The average prediction is based on two assumptions: 1) the face normal vector of a triangle is given by the average of the three vertex normal vectors of the triangle and 2) the face normal vector of a triangle can be approximated by the average of the face normal vectors of the neighboring triangles.

Fig. 7 illustrates the average prediction scheme, where f_i denotes a face normal vector and n_i denotes a vertex normal vector. Suppose that vertex normals n_1, n_2, n_3, n_4, and n_5 are already encoded, that vertex normal n is to be encoded and that surface normal f is also unknown. By the second assumption,

TABLE I
TEST MODELS FOR GEOMETRY COMPRESSION. nV = THE NUMBER OF VERTICES. nF = THE NUMBER OF FACES. THE ENCODING AND DECODING TIMES ARE MEASURED USING A COMPUTER WITH PENTIUM-IV 2.80 GHZ PROCESSOR AND 256 MB RAM

Model	nV	nF	Encoding time (sec.)		Decoding time (sec.)	
			MPEG-4	Proposed	MPEG-4	Proposed
HORSE	11,135	22,258	2.815	3.42	1.141	3.344
SKULL	10,952	22,104	2.640	3.359	1.156	3.313
BEETHOVEN	2,845	2,812	0.557	0.454	0.234	0.437
CROCODILE	17,332	21,590	3.836	4.359	1.469	4.219
CAM-SHAFT	54,898	52,500	14.047	16.25	4.188	16.094
57CHEVY	18,472	15,369	3.593	3.687	1.438	3.610

we have $f_1 \simeq ((f+f_2+f_3)/3)$. In other words, we can predict f by

$$\hat{f} = 3f_1 - f_2 - f_3. \tag{3}$$

Let $\hat{n}$ denote the prediction of n. Then, by the first assumption, (3) can be rewritten in terms of the vertex normal vectors

$$\left(\frac{\hat{n}+n_1+n_2}{3}\right) = 3\left(\frac{n_1+n_2+n_4}{3}\right) - \left(\frac{n_1+n_3+n_4}{3}\right) - \left(\frac{n_2+n_4+n_5}{3}\right).$$

Therefore, we have

$$\hat{n} = n_1 + n_2 - n_3 + n_4 - n_5. \tag{4}$$

Most normal vectors can be predicted with this average prediction rule. However, for rare normal vectors, including those at the start of the vertex layer traversal, only parts of the neighboring normal vectors are available. In such cases, we use the following rule:

$$\hat{n} = \begin{cases} n_1 + n_2 - n_4, & \text{if only } n_1, n_2 \text{ and } n_4 \text{ are available} \\ \frac{n_1+n_2}{2}, & \text{if only } n_1 \text{ and } n_2 \text{ are available} \\ n_1, & \text{if only } n_1 \text{ is available} \\ n_2, & \text{if only } n_2 \text{ is available} \\ 0, & \text{if no neighbor is available.} \end{cases} \tag{5}$$

Then, we encode the prediction residual

$$\Delta n = n - \hat{n}. \tag{6}$$

In MPEG-4 3-DMC, the normal vector is pre-quantized using the 8-4 subdivision scheme [2], [8]. The unit sphere is divided into eight octants, and each octant is subdivided recursively as shown in Fig. 8(a). Then, we can represent the normal vector by its octant number and triangle index, using the index pattern in Fig. 8(b). However, this approach has shortcomings. Since the unit sphere is approximated by an octahedron, quantized normal vectors are not ideally distributed. In other words, quantized normal vectors are uniformly distributed over the octahedron, instead of over the unit sphere. In addition, the index pattern in Fig. 8(b) is not suitable for differential coding. Since the triangles in each octant are represented by one-dimensional indices, neighboring triangles can have dissimilar indices. For example, in Fig. 8(b), although triangles 3 and 9 are adjacent, their index difference is as large as 6. Furthermore, there are index discontinuities along the boundary between two octants.

In order to overcome these shortcomings, we adopt the 6-4 subdivision scheme [19] to quantize the normal vectors. As shown in Fig. 9(a), the surface of the unit sphere is divided into six disjoint regions, called sextants, of the identical shape. The centers of these sextants are at (1, 0, 0), (−1, 0, 0), (0, 1, 0), (0, −1, 0), (0, 0, 1) and (0, 0, −1). For example, the sextant with the center at (1, 0, 0) is given by

$$\{(x,y,z) : x^2+y^2+z^2 = 1, |x| \geq |y| \text{ and } |x| \geq |z|\}$$

or it is described by two angle constraints

$$-\frac{\pi}{4} \leq \alpha \leq \frac{\pi}{4} \text{ and } -\frac{\pi}{4} \leq \beta \leq \frac{\pi}{4}$$

where α is the angle around the z-axis from the xz-plane, and β is the angle around the y-axis from the xy-plane.[1] Then, as shown in Fig. 9(b), we divide the sextant into l^2 cells by uniformly quantizing α and β into l levels, respectively.

The other sextants are divided similarly. To encode the prediction residual Δn in (6), we unfold the unit sphere around the sextant containing the prediction vector $\hat{n}$, as illustrated in Fig. 9(c), where each sextant is depicted as a square for simplicity. Then, the index differences between n and $\hat{n}$ are encoded by the QM coder. For example, in Fig. 9(c), the index differences are 3 in the horizontal direction and 1 in the vertical direction. In this way, we can prevent the discontinuity problem in MPEG-4 3-DMC and exploit the redundancy in normal vectors more effectively.

VI. EXPERIMENTAL RESULTS

A. Error Measures

In two-dimensional (2-D) image coding, root mean squared error (RMSE) or peak-signal-to-noise ratio (PSNR) is widely employed as an objective distortion measure, since they are easily computable and facilitate error analysis. However, these error measures are not always consistent with perceived signal quality.

Measuring the geometry distortion of a 3-D mesh model is even more difficult. Since excessively large quantization errors in just a few vertices may lead to catastrophic deformation of

[1]Note that these two angles are different from the longitude and the latitude in the spherical coordinate system.

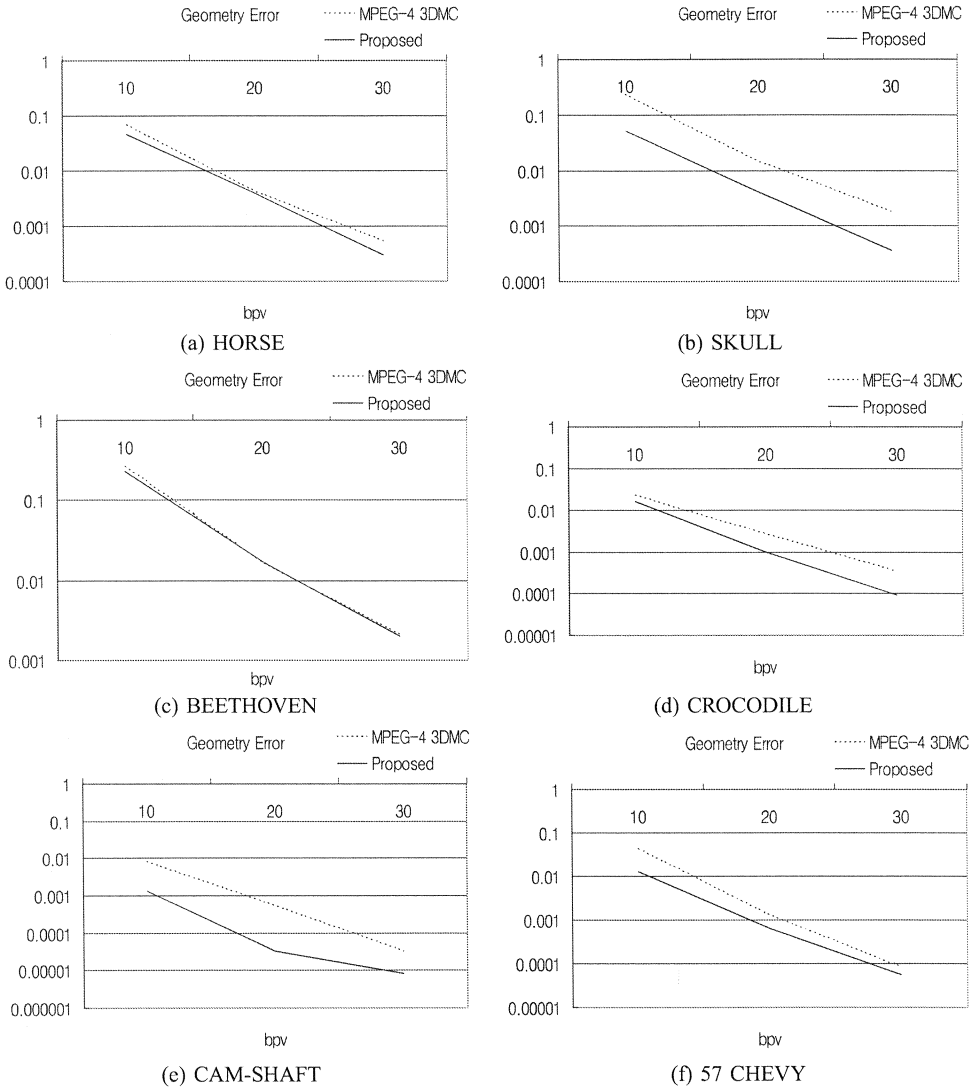


Fig. 10. Geometry compression results.

the 3-D model, RMSE or PSNR cannot represent the geometry distortion appropriately. The Hausdorff distance, which is a max-min distance between two sets of points, is more meaningful in 3-D geometry compression [20]. Therefore, we adopt the Hausdorff distance as the geometry error measure. Let $A = \{a_1, a_2, \ldots, a_k\}$ denote the set of original vertex positions, and $B = \{b_1, b_2, \ldots, b_k\}$ the set of reconstructed positions. Then, the Hausdorff distance between A and B is defined as

$$d_{\text{geometry}} = \max\{h(A,B), h(B,A)\} \tag{7}$$

where

$$h(A,B) = \max_{a_i \in A}\left\{\min_{b_j \in B} \|a_i - b_j\|\right\}.$$

In some sense, color data of 3-D meshes are similar to 2-D images. Hence, we employ RMSE as the color error measure, given by

$$d_{\text{color}} = \frac{1}{k}\sqrt{\sum_{i=1}^{k} \|c_i - \tilde{c}_i\|^2} \tag{8}$$

where c_i and $\tilde{c}_i$ denote the original and reconstructed colors of ith vertex, respectively.

When we render the 3-D model, the angle between a normal vector and a light direction is used. Therefore, the error measure for normal vectors should consider the angle between an original normal vector and its reconstruction. We define the normal error measure as

$$d_{\text{normal}} = \frac{1}{k}\sum_{i=1}^{k} \cos^{-1}(n_i \cdot \tilde{n}_i) \tag{9}$$

where n_i and $\tilde{n}_i$ denote the original and reconstructed normal vectors of ith vertex, respectively.

B. Geometry Compression

Table I summarizes the properties of test models for geometry compression. We measure the encoding and decoding times using a computer with Pentium-IV 2.80 GHz processor and 256 MB RAM. The proposed algorithm and the MPEG-4 3-DMC algorithm [8] require comparable encoding times. However, the proposed algorithm should perform the vertex

TABLE II
COLOR COMPRESSION RESULTS. nC = THE NUMBER OF LISTED COLORS TO DESCRIBE THE MODEL. nCS = THE NUMBER OF DIFFERENT COLORS USED TO PAINT THE MODEL. THE ENCODING AND DECODING TIMES ARE MEASURED USING A COMPUTER WITH PENTIUM-IV 2.80 GHZ PROCESSOR AND 256 MB RAM

Model	nC	nCS	Bit rate (bpc)	RMSE errors		Encoding time (sec.)		Decoding time (sec.)	
				MPEG-4	Proposed	MPEG-4	Proposed	MPEG-4	Proposed
HUMIDITY	39,072	401	5	0.0042	0.0041	20.29	21.11	4.14	4.28
			8	0.0005	0.0003	20.37	21.47	4.19	4.33
			10	0.0001	0	20.46	25.34	4.26	4.34
SPHERE	41,369	337	5	0.3793	0.1820	20.14	21.18	4.25	4.35
			7	0.1435	0.0931	20.27	21.36	4.29	4.42
			10	0.0641	0.0004	20.42	27.30	4.34	4.56
BRAIN	34,278	9,419	6	0.1441	0.1135	17.63	18.08	4.30	4.46
			9	0.0530	0.0492	17.78	18.63	4.36	4.53
			13	0.0112	0.0009	17.92	20.81	4.42	4.72
C-SKULL	84,635	16,775	4	0.2162	0.1704	50.91	52.57	10.02	10.54
			8	0.0539	0.0524	51.11	52.85	10.27	10.61
			10	0.0332	0.0012	51.25	70.67	10.45	11.69

layer traversal and find adjacent vertex lists in the decoder as well as in the encoder. Thus, the proposed algorithm takes about 2–4 times longer decoding times than the MPEG-4 algorithm.

In Fig. 10, we compare the rate-distortion performances of the proposed algorithm with those of the MPEG-4 algorithm. The horizontal axis represents bits per vertex (bpv), and the vertical axis represents geometry errors in the logarithm scale. The proposed scheme outperforms the MPEG-4 algorithm considerably for these test models. This is because the proposed algorithm yields higher prediction accuracy than the MPEG-4 algorithm, by exploiting the geometrical correlation of vertices and the curvature.

C. Color Compression

Fig. 5 shows 3-D test models for color compression and Table II summarizes their properties and compression results. In Table II, nC is the number of listed colors to describe the model, and nCS is the number of different colors used to paint the model. Note that nCS is much smaller than nC, which implies that the same color recurs frequently to paint many vertices in these models.

As mentioned in Section IV, we employ the mapping table scheme if the given bit rate is higher than or equal to 10 bpc, and the DPCM scheme otherwise. The proposed DPCM scheme provides slightly lower distortions than the MPEG-4 3-DMC algorithm. Moreover, note that at higher bitrates the proposed algorithm yields significantly lower distortions by employing the mapping table scheme. For example, the proposed algorithm provides about 160 times lower distortion than the MPEG-4 3-DMC algorithm, when the SPHERE model is encoded at 10 bpc.

For the color compression, the encoding and decoding times of the proposed algorithm are comparable with those of the MPEG-4 algorithm, respectively.

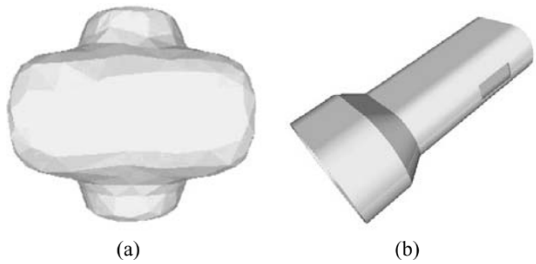


Fig. 11. Test models for normal vector compression. (a) TWO-CYLINDER. (b) LEGS.

D. Normal Vector Compression

The proposed normal compression algorithm is tested on TWO-CYLINDER and LEGS models in Fig. 11 and BRAIN and C-SKULL models in Fig. 5. Table III summarizes the properties of these test models. While LEGS consists of flat regions, and TWO-CYLINDER has curved surfaces. BRAIN and C-SKULL contain larger numbers of vertices and more complex normal vectors. As in the geometry compression, the proposed algorithm takes similar encoding times as the MPEG-4 3-DMC algorithm, but longer decoding times.

Fig. 12 compares the normal compression results, where the horizontal axis represents bits per normal (bpn) and the vertical axis represents normal errors. The proposed algorithm uses the 6-4 subdivision scheme and the average prediction scheme to improve the performance of the MPEG-4 3-DMC algorithm. The curves, labeled as "6-4 subdivision" in Fig. 12, show the performances when we employ only the 6-4 subdivision without using the average prediction. For example, for the BRAIN model at 5 bpn, MPEG-4 3-DMC yields the normal error 0.26. This is reduced to 0.25 when we use the 6-4 subdivision instead of the conventional 8-4 subdivision in MPEG-4 3-DMC. The proposed algorithm further reduces it to 0.23 by using the average prediction as well as the 6-4 subdivision. The proposed algorithm provides higher coding gains

TABLE III
TEST MODELS FOR NORMAL VECTOR COMPRESSION. nN = THE NUMBER OF NORMAL VECTORS. nF = THE NUMBER OF FACES. THE ENCODING AND DECODING TIMES ARE MEASURED USING A COMPUTER WITH PENTIUM-IV 2.80 GHZ PROCESSOR AND 256 MB RAM

Model	nN	nF	Encoding time (sec.)		Decoding time (sec.)	
			MPEG-4	Proposed	MPEG-4	Proposed
TWO-CYLINDER	410	816	0.313	0.406	0.156	0.397
LEGS	724	1,200	0.463	0.614	0.258	0.593
BRAIN	34,278	69,180	13.12	15.75	5.2	15.704
C-SKULL	84,635	172,002	42.50	45.782	13.4	45.016

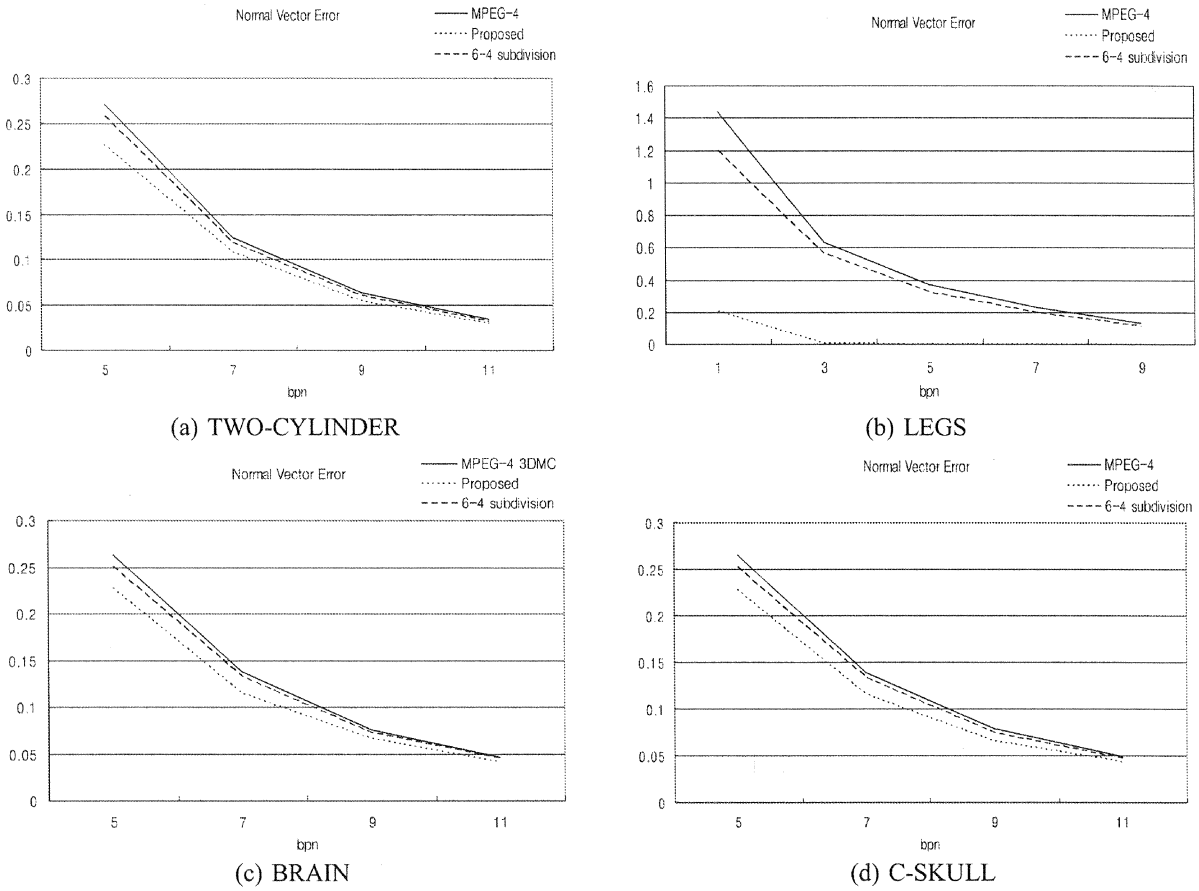


Fig. 12. Normal vector compression results. The graphs "6-4 subdivision" show the performances when we employ only the 6-4 subdivision without using the average prediction.

than the MPEG-4 3-DMC algorithm on the other test models also. Especially, for the LEGS model, a very high coding gain is achieved. This is because the normal vectors in LEGS are highly correlated, and thus can be effectively predicted by the proposed algorithm.

These simulation results indicate that the proposed algorithms for mesh geometry and attribute data compression are quite promising.

VII. CONCLUSION

In this work, we proposed the compression algorithms for geometry, colors and normal vectors of 3-D mesh models. For geometry data, we proposed a prediction rule which exploits all the position and angle information in the neighboring triangles. For color data, we proposed a mapping table approach to encode frequently recurring patterns effectively. For normal vectors, we developed the average predictor and the 6-4 subdivision quantizer. Simulation results demonstrated that the proposed coding schemes outperform MPEG-4 3-DMC on various test models. As future work, we need to investigate an efficient traversing scheme to achieve further coding gain, since the coding performance heavily depends on the input vertex traversal ordering.

REFERENCES

[1] M. Deering, "Geometry compression," *Proc. ACM Comput. Graph.*, pp. 13–20, Aug. 1995.
[2] G. Taubin and J. Rossignac, "Geometry compression through topological surgery," *ACM Trans. Graph.*, pp. 84–115, Apr. 1998.
[3] J. S. Choi, Y. H. Kim, H. J. Lee, I. S. Park, M. H. Lee, and C. T. Ahn, "Geometry compression of 3-D mesh models using predictive two-stage quantization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 12, pp. 312–322, Dec. 2000.

[4] J. H. Ahn and Y. S. Ho, "An efficient geometry compression method for 3-D objects in the spherical coordinate system," in *Proc. IEEE Int. Conf. Image Process.*, vol. 2, Oct. 1999, pp. 482–486.
[5] J. Li and C.-C. Kuo, "Progressive coding of 3-D graphics models," *Proc. IEEE*, vol. 86, no. 6, pp. 1228–1243, Jun. 1998.
[6] S. Gumhold and W. Straßer, "Real time compression of triangle mesh connectivitity," in *Proc. SIGGRAPH '98*, Jul. 1998, pp. 133–140.
[7] C. Touma and C. Gotsman, "Triangle mesh compression," in *Proc. Graph. Interface*, 1998, pp. 26–34.
[8] *Description of Core Experiments on 3-D Model Coding*, ISO/IEC JTC1/SC29/WG11 MPEG/M4566, Mar. 1999.
[9] A. Khodakovsky, P. Schroder, and W. Sweldens, "Progressive geometry compression," in *Proc. SIGGRAPH '00*, Aug. 2000, pp. 271–278.
[10] P. Alliez and M. Desbrun, "Progressive compression for lossless transmisison of triangle meshes," in *Proc. SIGGRAPH '01*, Aug. 2001, pp. 198–205.
[11] J. Peng, C.-S. Kim, and C.-C. J. Kuo, "Technologies for 3-D triangular mesh compression: A survey," J. Vis. Commun. Image Represent., vol. 16, no. 6, pp. 688–733, Feb. 2005, to be published.
[12] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compresson*. Norwell, MA: Kluwer, 1992.
[13] F. Ng, B.-L. Yeo, and M. Yeung, "Improving 3-D model geometry coding using DPCM techniques,", ISO/IEC JTC1/SC29/WG11 MPEG/M4719, 1999.
[14] J. H. Ahn and Y. S. Ho, "Geometry compression of 3-D meshes using optimal quantization for prediction errors,", ISO/IEC JTC1/SC29/WG11 MPEG/M3751, 1998.
[15] ——, "Geometry compression of 3-D models using adaptive quantization for prediction errors," in *Picture Coding Symp.*, Apr. 1999, pp. 193–197.
[16] M. A. Weiss, *Data Structures & Problem Solving Using Java*. New York: Addison Wesley, 2002.
[17] W. Pennebaker and J. Mitchell, *JPEG Still Image Compression Standard*. New York: Van Nostrand Reinhold, 1993.
[18] S. J. Colley, *Vector Calculus*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2002.
[19] J. Li, C.-C. Kuo, and H. Chen, "Embedded coding of mesh geometry,", ISO/IEC JTC1/SC29/WG11 MPEG/M3325, 1998.
[20] P. Cignoni, C. Rocchini, and R. Scopigno, "Metro: Measuring error on simplified surfaces," *Comput. Graph. Forum*, vol. 17, no. 2, pp. 167–174, Jun. 1998.