Fast 3D Mesh Compression Using Shared Vertex Analysis

Euee Seon Jang, Seungwook Lee, Bonki Koo, Daiyong Kim, and Kyoungsoo Son

A trend in 3D mesh compression is codec design with low computational complexity which preserves the input vertex and face order. However, this added information increases the complexity. We present a fast 3D mesh compression method that compresses the redundant shared vertex information between neighboring faces using simple first-order differential coding followed by fast entropy coding with a fixed length prefix. Our algorithm is feasible for low complexity designs and maintains the order, which is now part of the MPEG-4 scalable complexity 3D mesh compression standard. The proposed algorithm is 30 times faster than MPEG-4 3D mesh coding extension.

Keywords: 3D mesh compression, prefix code, real-time encoding and decoding.

I. Introduction

With the plethora of 3D mesh-based models in many graphics applications such as games, the need for 3D mesh compression for transmission and storage purposes is ever increasing. Many proposed technologies, including the latest MPEG-4 3D mesh coding (3DMC) standard, achieve good compression efficiency [1], [2]. However, few of these tools are being used in the graphics industry, perhaps because of the complexity during the encoding and decoding processes in most applications where the computational complexity should be minimized to support real-time rendering—one of the most important requirements for interactive 3D graphics.

3D mesh compression has been extensively researched in many studies with a focus on compression efficiency. The compression of both static and dynamic meshes over time has been investigated [3]. Peng and others provided a good survey on various 3D mesh compression technologies [2]. However, even the most efficient 3D mesh compression methods, such as transformation, prediction, quantization, and entropy coding, that are used to compress various attributes in a 3D mesh model (connectivity, geometry, and other attributes) do not take into account the order of faces and/or vertices in a given 3D mesh representation, despite its importance for animation and other interactive rendering processes.

In the MPEG-4 3DMC extension (3DMCe) [1], additional fields were introduced to transmit the original ordering information whenever needed. Considering the overhead incurred by transmitting the ordering information, the potential coding gain is lost, and the computational complexity of decoding and reordering increases greatly [4].

In this letter, we propose a fast 3DMC method that preserves the vertex/face ordering information with reasonable compression efficiency but improved computational complexity. The proposed method has the following features: fast connectivity coding by exploiting shared vertex properties, circular differential coding (CDC), and fast entropy coding using binary shift coding (BSC) and exponential Golomb coding (EGC) [5].

II. Proposed Codec

As shown in Fig. 1, the proposed codec consists of four processes: quantization (for geometry and attributes), shared

---

Manuscript received Aug. 25, 2009; revised Oct. 21, 2009; accepted Nov. 19, 2009.

This work was supported by the Strategic Technology Development program of MCST/MKE/KEIT, Rep. of Korea [2008-F-030-01, Development of Full 3D Reconstruction Technology for Broadcasting Communication Fusion].

Euee Seon Jang (phone: +82 2 2220 1086, email: esjang@hanyang.ac.kr), Daiyong Kim (corresponding author, email: dykim@dmlab.hanyang.ac.kr), and Kyoungsoo Son (email: bonokensin@dmlab.hanyang.ac.kr) are with the Department of Electronics, Computer, and Communication Engineering, Hanyang University, Seoul, Rep. of Korea.

Seoungwook Lee (email: tajinet@etri.re.kr) and Bonki Koo (email: bkkoo@etri.re.kr) are with the Contents Research Division, ETRI, Daejeon, Rep. of Korea.

doi:10.4218/etrij.10.0209.0357

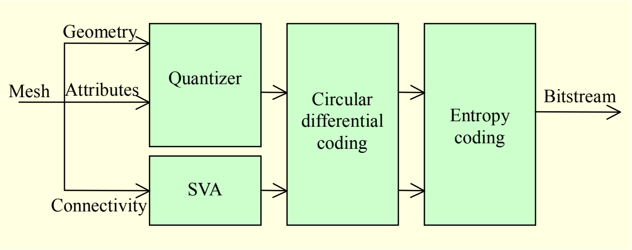

Fig. 1. Encoding diagram of the proposed method.

vertex analysis (SVA) (for connectivity), CDC, and entropy coding. The quantization process is the same as that of MPEG-4 3DMC. In the following subsections, we explain SVA, CDC, and entropy coding in detail.

1. Connectivity Coding: SVA

As shown in Table 1, each face contains the list of vertices that form the faces. A vertex may be used by several faces; therefore, it is very likely that the vertices used in the previous face will also be used in the current face. Through SVA, the connectivity between the current face and the previously encoded face is checked by counting the number of shared vertices. If an input 3D mesh is triangular, the number of shared vertices varies from zero to three. Therefore, we define four different modes to represent vertices for the current face as shown in Fig. 2. For a triangular face, the vertices in the previous face may not be reused in the current face (mode 0), reused by one vertex (mode 1), reused by two vertices (mode 2), or all reused (mode 3). Indices of new vertices in modes 0, 1, and 2 are subject to differential encoding by computing the difference in the vertex index (DVI), which is calculated by comparing the current vertex index and the previous vertex index. Depending on the number of unshared vertexes, additional information may be sent. For example, the location of the shared vertex (*shared position*) is necessary in mode 1, while the location of the unshared vertex (*unshared position*) is necessary in mode 2. The location field can be represented by two bits. For modes 2 and 3, the 1-bit face direction flag is included to indicate whether the order of the vertices in the current face is the same as that of the previous face.

In Table 1, the encoding mode of the first face is mode 0 because it does not have any previous face. We assume that all three vertex indices of the previous face are zeros; therefore, the DVIs for the first face are the same as the original vertex indices.

In the decoding process, the original order of the vertex index of a face may or may not be preserved. For example, a face of the vertex indices (0, 1, 2) may be decoded as (2, 0, 1) or (1, 2, 0). This is not a problem because the triangle is represented by three vertices. A face of (0, 1, 2) and a face of (0, 2, 1), however, are different in that the normal vectors of the two faces are in exactly the opposite directions. This is why we transmit the face direction flag whenever needed.

Table 1. Example of SVA and CDC ($M_d = 8$).

Face index	Face vertex index			Mode	Position	Face direction	DVI 1	DVI 2	DVI 3
	1st	2nd	3rd						
1	0	1	2	0	-	-	0	1	2
2	2	3	0	2	1	1	-	2	-
3	4	5	0	1	2	-	2	2	-
4	0	3	4	2	1	1	-	-2	-
5	5	6	1	0	-	-	5	3	-3

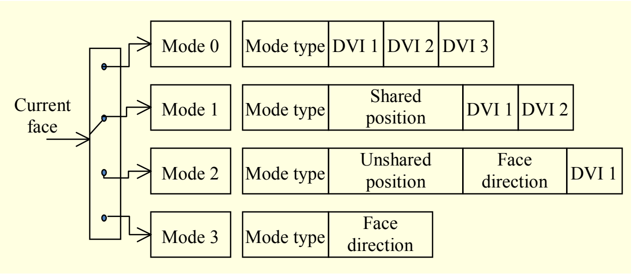

Fig. 2. SVA mode.

2. Circular Differential Coding

A DVI value from SVA may range from $-(M_d-1)$ to (M_d-1), where M_d is the total number of vertices. We can reduce the differential error range by half by calculating the circular differential as

$$dvi = vi_{\mathrm{c}} - vi_{\mathrm{p}}$$

$$dvi = \begin{cases} dvi - M_d, & \text{if } dvi > \left\lfloor \dfrac{M_d}{2} \right\rfloor; \\ dvi, & \text{if } -\left\lfloor \dfrac{M_d}{2} \right\rfloor \le dvi \le \left\lfloor \dfrac{M_d}{2} \right\rfloor; \\ dvi + M_d, & \text{if } dvi < -\left\lfloor \dfrac{M_d}{2} \right\rfloor, \end{cases}$$

where vi_{c} and vi_{p} represent the current and previous vertex indices. For example, the first DVI of face 5 in Table 1 is changed from 5 to –2. The decoding process of the CDC is carried out as follows:

$$vi_{\mathrm{c}} = vi_{\mathrm{p}} + dvi,$$

$$vi_{\mathrm{c}} = \begin{cases} dvi - M_d, & \text{if } dvi \ge M_d; \\ vi_{\mathrm{c}}, & \text{if } 0 \le dvi < M_d; \\ dvi + M_d, & \text{if } dvi < 0. \end{cases}$$

3. Entropy Coding: BSC and EGC

The main idea of the BSC method, called the "Huffman shift," is to decide the bit length of the symbols. The size of a symbol in a BSC representation is a multiple of the length of the bits. A symbol in the BSC representation can be determined by a modulus operation with each quotient and remainder pair as shown in [5]. The probability density function of the CDC output is an exponential-like function which has a zero-mean and $M_d/2$ as the maximum value. The fastest and best coding algorithm for this distribution is EGC [6].

We employed BSC and EGC because they produce good compression efficiency with very low computational complexity. In our previous research, we found that application of BSC to image and video coding is quite effective in that BSC is several times faster than conventional variable length decoding or arithmetic coding. Further details on BSC can be found in [7].

III. Experimental Results

We evaluated the proposed algorithm with two other methods: the MPEG-4 3DMCe reference codec and the quantization-based compact representation (QBCR), which represent the two extreme points in compression efficiency and computational complexity. MPEG-4 3DMCe is an anchor point to evaluate the compression performance of the proposed method, and QBCR is an anchor point to evaluate the computational complexity. QBCR is the simplest design to build a low-complexity 3D mesh compression as it only uses quantization. It is also a fast design in that it requires only inverse quantization at the decoder. Each field value (vertex or face) can go through the conventional graphics pipeline with additional quantization, which increases the complexity only negligibly.

For evaluation, we used the 729 3D mesh models that are used in the MPEG-4 3DGC website [8]. We measured the compression efficiency by averaging the compression performance of the 729 models for each method as shown in Fig. 3. From the figure, it is quite clear that the performance of the proposed method is between that of 3DMCe and that of QBCR. The proposed method is compared with 3DMCe and QBCR in terms of computational complexity by measuring the encoding and decoding times. In Fig. 4, it is quite apparent that the proposed method and QBCR are much faster than 3DMCe. In summary, the proposed methods were 35 times faster on average than the 3DMCe encoder and decoder.

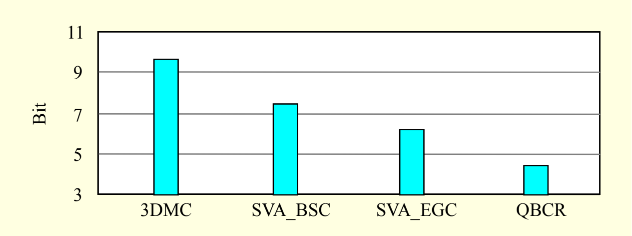

Fig. 3. Average compression ratio.

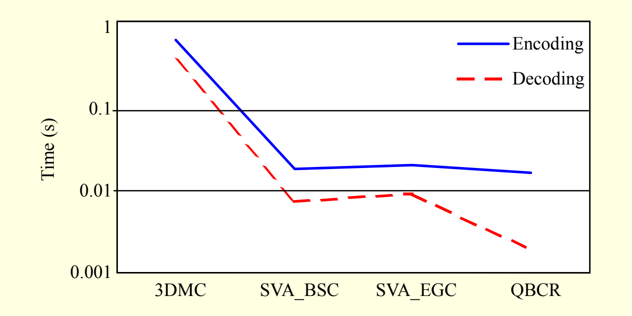

Fig. 4. Average encoding and decoding times of 4 methods.

IV. Conclusion

In this letter, we presented a fast 3D mesh coding method using a shared vertex analysis. The experimental results showed that the proposed method achieved very low computational complexity with a reasonable coding gain. The major advantages of the proposed algorithm are the simple and efficient design of connectivity coding, the use of CDC for efficient differential coding, and the adoption of low-complexity entropy coding. Extending the proposed method to encoding 3D models with other attributes, such as colors, texture coordinates, normals, and so on, would be a promising research topic in the near future.

References

[1] ISO/IEC 14496-16:2006/Amd.1:2007 Part 16: Animation Framework eXtension (AFX) AMENDMENT 1: Geometry and Shadow, 2007.

[2] J. Peng, C. Kim, and C.-C.J. Kuo, "Technologies for 3D Triangular Mesh Compression: A Survey," *J. Visual Commun. Image Represent.*, vol.16, no. 6, Dec. 2005, pp. 688-733.

[3] S. Ramanathan et al., "Impact of Vertex Clustering on Registration-Based 3D Dynamic Mesh Coding," *Image and Vision Computing*, vol. 26, no. 7, July 2008, pp. 1012-1026.

[4] E. Chang et al., "Vertex and Face Permutation Order Compression for Efficient Animation Support," *Proc. Electronic Imaging*, 2006.

[5] G. Son et al., "Simple and Fast Compression of 3D Meshes," *Int. Conf. Convergence Inf. Technol.*, 2007, pp. 2175-2180.

[6] I.E.G. Richardson, *H.264 and MPEG-4 Video Compression*, John Wiley & Sons Ltd., 2003.

[7] D. Salomon, *Data Compression,* 3rd Ed., Springer, 2004.

[8] 3D Mesh Test Models in the MPEG-4 3D Graphics Group, http://www.gti.ssr.upm.es/~mpeg/3dgc/3Dmodels.