TRIANGLE MESH COMPRESSION USING GEOMETRIC CONSTRAINTS

Jae Young Sim, Chang-Su Kim, and Sang Uk Lee

Signal Processing Lab., School of Electrical Eng., Seoul National University
San 56-1 Shillim-dong, Kwanak-gu, Seoul 151-742, KOREA
Tel: +82-2-880-8428, Fax: +82-2-880-8220
E-Mail: jyoung@ipl.snu.ac.kr

ABSTRACT

It is important to compress three dimensional (3D) data efficiently, since 3D data are too large to store or transmit in general. In this paper, we propose a lossless compression algorithm of the 3D mesh connectivity, based on the vertex degree. Most techniques for the 3D mesh compression treat the connectivity and the geometry separately, but our approach attempts to exploit the geometric information for compressing the connectivity information. We use the geometric angle constraint of the vertex fanout pattern to predict the vertex degree, so the proposed algorithm yields higher compression efficiency than the conventional algorithms.

1. INTRODUCTION

In recent years, 3D images or data have drawn much attention as the multimedia type of next generation, according to the progress of many 3D modeling technologies and the expansion of various 3D applications. But, the 3D data is too large to store or transmit through constrained bandwidth channels, so it is essential to compress them efficiently. The 3D model in computer graphics is mainly represented by triangular or polygonal meshes, yielding the geometry information (vertex positions) and the connectivity information for the vertices. However, the random structure of the polygonal meshes makes it difficult to use the conventional compression techniques for 2D image or video data to compress the 3D data directly. Thus, the need for efficient compression schemes of the 3D mesh data is increasing dramatically.

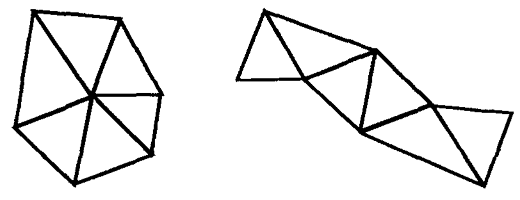
In computer vision area, the notion of level of detail (LOD) has been introduced as the compressing method for the 3D mesh model, but it transforms the original topology of the object and cannot achieve the lossless compression [4]. Until now, several techniques have been proposed to losslessly compress the connectivity information of the 3D triangular mesh model. Deering proposed the generalized mesh, composed of the vertex-buffer and the triangle strip [1]. Taubin *et al.* proposed the vertex and triangle spanning trees, and conjectured that the optimization problem of the spanning trees is NP-complete. In their algorithm, the connectivity information is encoded using only two bits per triangle on average [2]. Gumhold *el al.* presented the fast compressing and decompressing algorithm for the triangle mesh connectivity, by arranging arbitrary triangular mesh into one triangular strip using the breadth first traversal [3]. Tauma *et al.* represented the mesh connectivity as a list of the vertex degrees in a special order, then assigned one of the 'add', 'split', 'merge', and 'add dummy' codes to each vertex degree. It was shown that the connectivity information is encoded with 1.5 bits per vertex on average [5].

In this paper, we follow the vertex degree representation in [5], and propose a more optimized and efficient mesh coding algorithm, called the vertex fanout pattern coding (VFPC). We reduce the entropy of the connectivity information (vertex degree), by *predicting* each vertex degree with the geometrically discovered fanout shape of the vertex. In other words, the proposed algorithm exploits the geometry information for encoding the connectivity information, while the conventional algorithms consider the mesh connectivity is independent of the mesh geometry [1, 2, 5]. Thus, the proposed algorithm can yield better coding performance for the connectivity information than the conventional algorithms.

This paper is organized as follows. In Section 2, we describe the topological patterns of the vertex fanout and the vertex fanout shapes, and propose the VFPC algorithm. In Section 3, we compare the performance of the proposed algorithm with those of the conventional algorithms. Finally, we conclude this paper in Section 4.

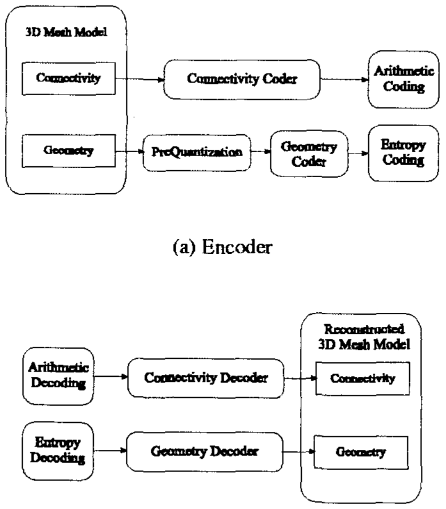
2. PROPOSED ALGORITHM

As mentioned previously, the conventional algorithms treat the mesh connectivity as separated from the mesh geometry [1, 2, 5]. Fig. 1 shows two types of the connected set of triangles, fanout and strip. Most compression algorithms for the mesh connectivity are related to making the optimized triangle strip [1, 2], while [5] uses the topological fanout. In this paper, we compress the mesh connectivity classified by four topological fanout patterns which are explained later, using the combined information of the geometrical vertex fanout shapes. Fig. 2 shows the block diagram for the encoder and decoder of the proposed VFPC algorithm. The geometry information (vertex positions) is represented by assigning three floating point numbers (x, y, z) to each vertex. We pre-quantize these floating point numbers with various resolutions, and use the parallelogram method [6] to compress the quantized vertex position. In case of the connectivity information, we first find a seed vertex, then put it in the active



(a) Fanout (b) Strip

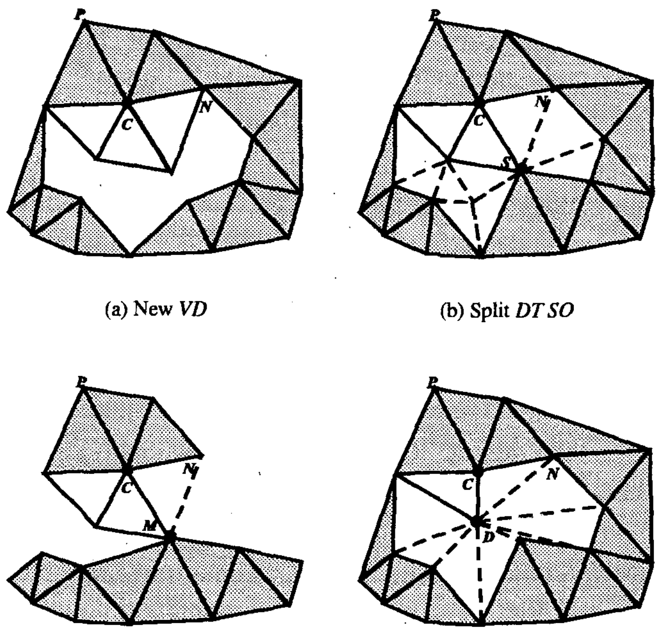
Fig. 1. Two types of the triangle set.



(a) Encoder

(b) Decoder

Fig. 2. The block diagram of the proposed algorithm.



(a) New *VD* (b) Split *DT SO*

(c) Merge *DT MO*1 *MO*2 (d) Bound *DT*

Fig. 3. Four cases of the topological pattern of the vertex fanout.

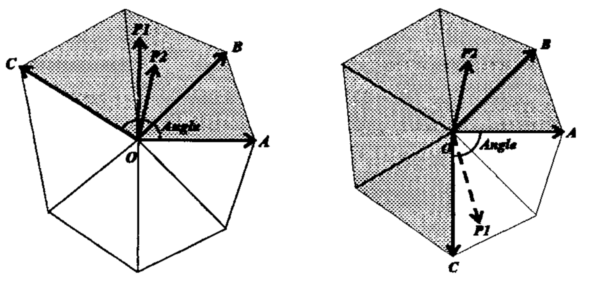
vertex list, and send its fanout degree. Then, we send all the positions of vertices that are connected to the seed vertex, in order to discover all the triangles composing the fanout pattern of the seed vertex. Meanwhile, we insert new vertices in the active vertex list, and remove the seed vertex from it. This procedure is repeated until all the triangles abut on the vertices in the active vertex list are discovered. In this mesh expansion process, there are four cases of the topological patterns of the vertex fanout, and two cases of the fanout geometry shapes. We employ the fanout geometry shapes to reduce the entropy of the code distribution of the vertex fanout degrees. The decoding process is simply the inverse of the encoding process.

2.1. Topological Vertex Fanout Patterns

Fig. 3 shows the topological vertex fanout patterns, which are classified into 4 cases, in which *P*, *C*, and *N* denote previous, current, and next vertices in the active vertex list, and *S*, *M*, and *D* denote splitting, merging, and dummy vertices, respectively. The shaded region of the mesh represents discovered (coded) triangles. Solid and dashed lines represent triangles at the current vertex, which are currently found and not yet encoded, respectively. Fig. 3(a) is the 'new' case. At the current vertex, the vertex degree, *VD*, is determined by counting the triangles abut on the current vertex. In this case *VD* is 6. Fig. 3(b) is the 'split' case, which occurs when one of the vertices organizing newly discovered triangles is already a vertex in the active vertex list. In this case we should transmit two parameters, the number of newly discovered triangles (*DT*) until the splitting vertex *S* is appeared, and the clockwise offset (*SO*) of *S* from the current vertex *C* in the active vertex list. In this Figure, *DT* is 2 and *SO* is 4. Fig. 3(c) is the 'merge' case, which only takes a place on the torus type mesh. The only difference from the 'split' case is that the merging vertex *M* exists in the other list in the super-list of the active vertex lists. Thus, another extra parameter *MO*2, which indicates the index of the list in the super-list, is required with the merge offset *MO*1. Fig. 3(d) is the 'boundary' case, which occurs when the mesh has boundaries. If the mesh contains a boundary, we insert the dummy vertex *D*, generating dummy triangles, as shown in this figure. Triangles having dashed lines in Fig. 3(d) are dummy triangles at the boundary. So we code this case with a parameter *DT*, which is the number of the discovered triangles at the current vertex until the dummy vertex is discovered.

2.2. Geometric Constraint

While encoding the topological patterns of the vertex fanout as mentioned previously, we exploit the geometric constraint for achieving the better compression efficiency. In [5], at the time of adding a new vertex into the active vertex list, the vertex degree and the position of the vertex are encoded by the 'add' code. Thus, at the time the decoder receives



(a) Small (b) Big

Fig. 4. Two geometric shapes of the vertex fanout.

the 'add' code, only one triangle abut on the vertex is reconstructed geometrically. On the contrary, we simultaneously send the degree code (topological fanout patterns) of the current vertex and the position information of all the vertices connected with the current vertex. Therefore, we can use the partially discovered angle of the vertex fanout to predict the vertex degree. Fig. 4 represents two cases of deciding the angle of the undiscovered vertex fanout part. Shaded and white regions of the vertex fanout represent discovered and undiscovered triangles, respectively, and *O* is the current vertex in the active vertex list.

The prediction consists of three steps. First, we find the angle which is smaller than π, from vectors ($\mathbf{OA}$, $\mathbf{OC}$) connected to the current vertex, by

$$Angle = \arccos\left(\frac{\mathbf{OA} \bullet \mathbf{OC}}{|\mathbf{OA}|\,|\mathbf{OC}|}\right). \tag{1}$$

Second, we should determine which part of the fanout is undiscovered to find the undiscovered angle of the fanout shape. This routine uses one additional discovered vector abut on the current vertex, *i.e.*, $\mathbf{OB}$. In Fig. 4, $\mathbf{OP1}$ is the cross product of $\mathbf{OA}$ and $\mathbf{OC}$, and $\mathbf{OP2}$ is the cross product of $\mathbf{OA}$ and $\mathbf{OB}$, given by

$$\begin{aligned}\mathbf{OP1} &= \mathbf{OA} \times \mathbf{OC},\\ \mathbf{OP2} &= \mathbf{OA} \times \mathbf{OB}.\end{aligned}$$

If the discovered angle of the geometric fanout shape is smaller than π, then the inner product of two cross products, $\mathbf{OP1}$ and $\mathbf{OP2}$, yields a positive value. On the contrary, if the discovered angle is larger than π, the inner product of $\mathbf{OP1}$ and $\mathbf{OP2}$ yields a negative value. By denoting the undiscovered angle by *UA*, the pseudo code for deciding the geometric shape of the vertex fanout is given as follows.

```
if ($\mathbf{OP1} \bullet \mathbf{OP2} < 0$)
    $UA = Angle$
else if ($\mathbf{OP1} \bullet \mathbf{OP2} > 0$)
    $UA = 2\pi - Angle$
```

Third, we predict the vertex degree of the current vertex using the angle (*UA*), and assign a code of the vertex degree error $\mathcal{E}$, which is the difference between the real vertex degree and the predicted vertex degree, given by

$$\mathcal{E} = (UT + DT) - RD, \tag{2}$$

$$\text{where } NF = \frac{\pi}{6}, \text{ and } UT = [\frac{UA}{NF}].$$

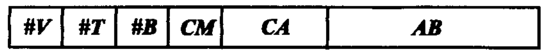
Our approximation of the normalization factor NF as $\frac{\pi}{6}$ is found to be reasonable, since in typical meshes, the most frequent vertex degree is 6, and the geometric shape of this vertex fanout is often composed of 6 triangles similar to equilateral triangles. UT is the number of undiscovered triangles, which is the positive integer closest to $\frac{UA}{NF}$. DT is the number of discovered triangles, and RD is the real vertex degree. With this geometric constraint (GC), the vertex degree error codes are more compactly distributed than the original codes of the vertex degrees without the geometric constraint.

Table 1. The comparison of 'add' codes and $\mathcal{E}$ codes distribution of the model "bunny".

'add' Codes (Without GC)	Frequency	$\mathcal{E}$ Codes (With GC)	Frequency
3	11	-3	5
4	390	-2	56
5	3971	-1	3430
6	26242	0	28220
7	3893	1	1709
8	300	2	53
9	23	3	10
10	3	4	18
11	1	5	3
12	1	exception	3
Total Freq.	34835	Total Freq.	33507

2.3. Entropy Coding Scheme

Table 1 is the comparison of the code distribution between 'add' codes without using GC (conventional algorithm, [5]) and $\mathcal{E}$ codes using GC (proposed algorithm) for the 3D mesh model, "Bunny", when each vertex is pre-quantized with 12bit resolution. It can be seen that $\mathcal{E}$ codes are more compactly distributed than the 'add' codes. In conventional algorithm, 'add' codes in [5] occur *NV* times, where *NV* is the number of vertices. But, in the proposed algorithm, $\mathcal{E}$ codes occur less than *NV* times, since we need not transmit the vertex degree information of the remained vertices, if the number of the vertices in the active vertex list is less than 3. In such case, note that all the triangles abut on the remained vertices are discovered already. Therefore, the proposed algorithm can yield better compression performance for the connectivity information than [5]. We have also found that the vertex degree error codes $\mathcal{E}$ and a few 'split' codes occur frequently, while most 'split' cases, all the 'merge' cases, and all the 'boundary' cases are rare. Thus, we employ a two level coding method using an arithmetic coding and a fixed length coding. First, we define *N* frequent codes with



#V : Number of vertices
#T : Number of triangles
#B : Number of boundaries
CM : Context mode bit
CA : Context bit array
AB : Arithmetic bit stream

Fig. 5. The bitstream of the proposed coding scheme.

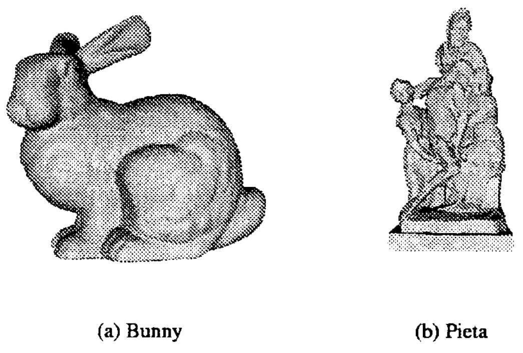


Fig. 6. 3D mesh models.

N symbols, and four extra symbols for the rarely occurring codes, which are 'new exception', 'split exception', 'merge case', and 'boundary case' codes. So the total number of symbols for the arithmetic code is *N*+4. Second, for the extra symbols at the arithmetic coding level we encode some parameters related these rare cases using a fixed length coding method. Fig. 5 shows the arithmetic coding bitstream with the header information.

3. EXPERIMENTAL RESULTS

Experiments have been performed on several MPEG4-SNHC 3D mesh data, which are described with the VRML language. Fig. 6 shows the examples of the 3D mesh models. "Bunny" is very regular model, which has 34835 vertices, 69473 triangles, and 4 boundaries, and "Pieta" is very irregular model of torus type, having 3476 vertices, 6976 triangles. The pre-quantized data is indistinguishable from the original model, if bits per coordinate is more than 12. Table 2 compares the performances of the proposed algorithm and the conventional algorithms ([2], [5]) at the various bit resolutions of pre-quantization. Conventional connectivity compression algorithms are not affected by the resolution of the geometric pre-quantization, since they separate the connectivity from the geometry. But, since the proposed algorithm employs the geometry information to compress the connectivity information, the performance depends on the geometric pre-quantization resolution. It can be seen that, if the resolution of the pre-quantization step is low, the performance of the proposed algorithm is comparable to the conventional algorithm in [5], since the 3D mesh model is too coarse to exploit the geometric constraint information. However,

Table 2. The experimental results of the proposed algorithm compared to previous methods for various pre-quantization resolutions.

3D mesh model	Ver-tices	[2] (bpv)	[5] (bpv)	Proposed (bpv)		
				8bit	10bit	12bit
bunny	34835	2.84	1.56	1.61	1.34	1.26
pieta	3476	5.83	5.29	4.82	4.68	4.66

for the "Bunny" model, if the pre-quantization resolution is more than 12 bits per coordinate, the average bit per vertex of the connectivity is lower than [2] by about 1.6 bit per vertex (bpv) and than [5] by 0.3 bpv, respectively. And for the "Pieta" model, the proposed algorithm requires about 1.2 and 0.6 less bpv than [2] and [5], respectively.

4. CONCLUSIONS

For the compression of the connectivity information of 3D triangle mesh data, we proposed an efficient and optimized coding algorithm using the geometric constraint. First, we have classified four cases of the topological vertex fanout patterns. Second, we have described the geometric constraint method, which is used for the prediction of the vertex degree. The experimental results on several 3D triangle mesh data demonstrated that the proposed algorithm yields the better compression performance than the conventional approaches [2, 5].

REFERENCES

[1] M. Deering, "Geometry compression," in *Proc. of SIGGRAPH*, pp. 13-20, ACM Press, 1995.

[2] G. Taubin and J, Rossignac, "Geometry compression through topological surgery," *Research Report* RC-20340, IBM Research Division, 1996.

[3] S. Gumhold and W. Straßer, "Real time compression of triangle mesh connectivity," in *Proc. of SIGGRAPH*, pp. 133-140, ACM Press, 1998.

[4] H. Hoppe, "Progressive meshes," in *Proc. of SIGGRAPH*, pp. 99-108, 1996.

[5] C. Tauma and C. Gotsman, "Triangle mesh compression," in *Proc. Graphics Interface'98*, pp. 26-34, 1998.

[6] "Description of core experiments on 3D model coding," ISO/IEC JTC1/SC29/WG11 MPEG98/N2302, July 1998.

[7] T.M. Cover and J.A. Thomas, *Elements of Information Theory*, John Wiley & Sons, 1991.