AN IMPROVEMENT OF CONNECTIVITY COMPRESSION FOR TRIANGULAR MESHES

B. S. Jong
Department of Information andComputer Engineering
Chung-Yuan Christian University

T. W. Lin
Department of Computer and Information Science
Soochow University

Abstract

Polygonal meshes have been used as the primary geometric model representation for networked gaming and for complex interactive design in manufacturing. Accurate polygonal mesh approximation of a surface with sharp features requires extremely large number of triangle. The need for succinct representation is not only to reduce storage requirements, but also to consume less network bandwidth and thus reduce the transmission time. In this paper, we propose a new method for compressing the connectivity of triangle meshes homeomorphic to a sphere. In average, it encodes with less than 1.19t bits of any t-triangles mesh. Furthermore, it is suitable for compressing all models and particularly attractive for compressing large object for remote instant access without overhead.

1. Introduction

Compression of 3D objects has becoming important not only for storage but also for interaction. Although complexity of industrial CAD models significantly raises the size of memory for storing these models, new production technology dramatically reduces storage cost such that the storage cost for these models is lower. The performance of graphics display affects user interaction. High-level display adapters is limited by a data transfer bandwidth or insufficient onboard memory to store the whole model. In particular, 3D models is distributed over the internet for collaborative design, gaming, learning, etc. The virtual interaction is seriously limited by the available bandwidth. The 3D object compression is one of possible method for providing feasible human-machine interaction [1-6].

There are many representations for 3D objects. The polyhedral models remain the primary 3D representation used in manufacturing, architectural, GIS, geoscience, and entertainment industries [7-8]. In particular, polyhedral models are effective for hardware-assisted rendering, which is important for video games, virtual reality, fly-through, and electronic mock-up applications involving complex models.

A set of non-overlapping triangles may be rendered efficiently using hardware-assisted rasterizers. Triangle count is a suitable measure of a model's complexity and triangle-meshes are an appropriate target for current efforts on compression [8]. A triangle mesh may be represented by its vertex data and by its connectivity. Vertex data comprises coordinates of all the vertices and optionally the coordinates of the associated normal vectors and textures. Connectivity captures at least the incidence relation between the triangles of the mesh and their bounding vertices. For most meshes in practice, the number of triangles is roughly twice the number of vertices [8]. Consequently, when pointers or integer indices are used as vertex-references and when floating point coordinates are used to encode vertex locations, connectivity data consumes twice as much storage as vertex coordinates. Furthermore, for most applications, vertex data may be compressed down to about a fifth of the uncompressed one, with an average of 12 bits per vertex location and 6 bits per vertex normal.

There are many approaches to compress the connectivity data. Automatically computed triangulation may be used as a first guess for connectivity and then only a little information is necessary to be stored for producing the correct connectivity. Unfortunately, these approaches are not compatible with the schemes for compressing vertex data. They require the connectivity information for predicting the data for each new vertex from previously encoded neighbors. The lack of connectivity information would considerably increase the storage needed to encode vertex data.

Therefore a new compression scheme is necessary to solve the bottleneck. It individually encodes the connectivity information and the vertex data of triangle-meshes. In this scheme, the connectivity information of a triangle can be compressed in which the required memory is between 1.7 and 2 bits. Apply entropy codes to the compressed connectivity information, the required memory could be reduced to 1.3 bits per triangle. There is a major drawback of this method. It requires two passes for decompressing the compressed connectivity information.

In this paper, we propose new idea to trace the connectivity relations between triangle-meshes. Only little information may be kept in the tracing boundary edges. In this manner, our method is effective and efficient not only in compression but also in decompression. Only single pass is performed in them. The required memory for connectivity information is reduced to about 1 bit per triangle in average.

2. Related work

Although many representations have been proposed for 3D models, polyhedra are the de facto standard for exchanging and viewing 3D data sets. This trend is reinforced by the wide spread of 3D graphic libraries and 3D adapters for personal computers. Current works in polyhedra compression may be subdivided into three categories: polyhedral simplification, compression of attributes, and encoding of the connectivity information [1-8].

In polyhedral simplification methods, the number of vertices in the mesh is reduced by changing the model's connectivity. To minimize the error produced by the simplification, the positions of the remaining vertices may be adjusted. The major purpose of these techniques is to accelerate graphics speed and to reduce data volume. Multiple levels of detail of meshes for an object are generated to achieve these aims. Representations of level of detail could be considered for lossy compression. They are inappropriate for applications that require access to the exact connectivity of the model.

Lossy or lossless compression methods are used to reduce the storage necessary for the attributes of an objects such as vertex locations, normals, colors, and texture. Applying general purpose data compression algorithms to the data stream of attributes leads to suboptimal solutions. In the approach of normalizing the geometry into a unit cube, vertex coordinates could be rounded off fixed length integers. The rounding controls the amount of lost information. Within a spatial organization of vertices into a spanning tree and geometric predictors of positions and properties, attributes may be encoded losslessly with fewer bits and further compressed with standard lossless entropy encoding techniques.

Connectivity encoding is the central part of any 3D compression method. It guides the geometry and photometry coding. Single-resolution compression does not change the connectivity in the sense that the decoder can perfectly recover the connectivity. For an input mesh, vertex-layers and triangle-layers are constructed by using a traversal algorithm. The entire connectivity encoding procedure is: (1) encode the total number of layers; (2) encode the information of each vertex-layer; and (3) encode all triangle information in each triangle layer.

3. New compressing method

The compression algorithm is restricted to manifold representations of triangle meshes. In a manifold mesh, each edge is bounding one or two triangles and the star of each vertex v (i.e., its incident triangles and edges) remains connected, when v is removed. Edges that bound two triangles are called interior edges. Edges that bound exactly one triangle are called exterior edges. The union of exterior edges of all triangles is called the boundary of these triangles. Vertices that do not bound any exterior edge are called interior vertices. The other vertices are called exterior vertices.

Our compression algorithm performs a series of steps. Each step removes one triangle from the current mesh. At each stage, the remaining portion of the mesh is a maximally connected component of the interior of the union of the remaining triangles $\{T\}$. Boundary B of such a component is kept. One edge of these exterior edges is referred as the active gate, g.

At each step, we identifies the unique triangle, X, that is part of $\{T\}$ and is incident upon g. Let v be the only vertex of X that does not bound g. We analyzes the relation that v has with respect to B and g. There are three basic relations labeled C, L, and R. The selection of the appropriate relation may be performed by the following sequence of tests:

If $v \notin B$ then relation C
Else If v precedes g then relation R
Else If v follows g then relation L

To clarify some implementation details, we introduce a data structure for storing the connectivity of the mesh. This data structure is based on the concept of a half-edge used in many polyhedral representations. A half-edge h is an edge of a triangle T whose interior is traceing around the border of the triangle in the clockwise direction. With each half-edge h, we associate the following entities:

- $h.s$ is the starting vertex for h.
- $h.e$ is the ending vertex for h.
- $h.v$ is the third vertex of T that does not bound h.
- $h.n$ is the half-edge that follows h in the boundary of T.
- $h.p$ is the half-edge that precedes h in the boundary of T.

We use a doubly linked list for storing the links between the successive boundary edges B. With each boundary edge b, we associate the following entities:

- $b.p$ is the boundary edge that precedes b in B.
- $b.n$ is the boundary edge that follows b in B.
- $b.f$ is a flag associated with b.

It is clear that the active gate g is contained in B. The boundary edge e associated with g is called active edge.

To determine appropriate relation, the condition $v \notin B$ must be examined. A simple examination is by tracing around the boundary edges to determine whether the vertex v is an interior vertex. It consumes much execution time. To eliminate this drawback, a binary flag is associated with each vertex. Initially, the binary flag of each vertex is set to zero. When a vertex is traced, i.e. the vertex becomes a exterior vertex, its binary flag is set to one. Therefore, the condition $v \notin B$ could be checked by verfying its binary flag $v.f$.

Compression algorithm identifies the relation type using:

If not $g.v.f$ then relation C
Else If $g.p = e.p$ then relation L
Else If $g.n = e.n$ then relation R

For each relation, the corresponding changes to the half-edge and boubly linked list data structures must be performed as follows.

a) relation C:

$g.v.f = 1$
new a node b
$b.p = e.p$
$b.n = e$
$b.f = 0$
$e.p = b$
$g = reverse(g.n)$

b) relation L:

b = *e.p*
e.p = *b.p*
destroy node *b*
g = *reverse*(*g.n*)

c) relation R:

b = *e.n*
e.n = *b.n*
destroy node *b*
g = *reverse*(*g.p*)

Only these three basic relations could not compress a triangle meshes with manifold representation. For example, a mesh is shown in Fig. 1. This mesh may represent the final stages of the compression of a large region in the mesh or the full compression of a small mesh with boundary. Starting at gate g, our algorithm removes triangles by following the arrows shown in Fig. 2. The active gate g in Fig. 2 has no relations within these three cases. In such a situation, we increase a relation called jump relation. In the jump relation, no triangles are removed. The operations to be performed is to set the flag of the active edge and to move active gate *g* along the boundary edges *B* until a boundary edge whose flag is not set.

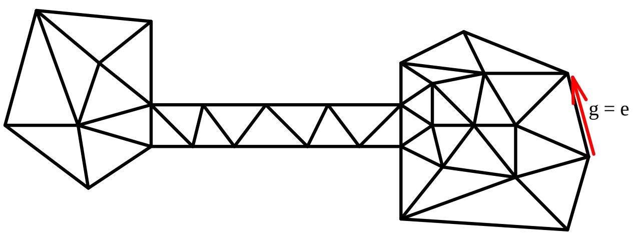

Fig. 1: A triangle mesh

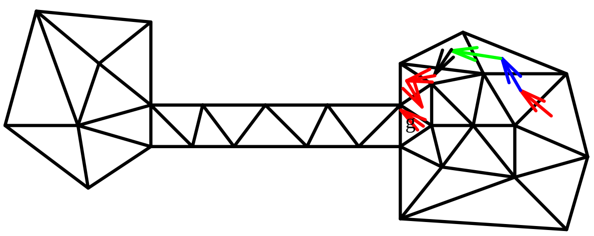

Fig. 2: Triangles are removed by following successive arrows

The compressed format is simple for our algorithm. Suppose the boundary edges are compressed in a compact representation. Only the relation C involves a new interior vertex. Only in such a situation, a vertex coordinates must be encoded. For the simplicity of decompression, the code length of coordinates for a vertex is assumed to be fixed. Only an indicator is necessary to be stored for other relations. Therefore it is easy to decompress the compressed data and to obtain the original triangle meshes.

4. Experimental studies and conclusions

The area of 3D compression will grow significantly over the next few years. There are many techniques to be used. A famous method, Edgebreaker, has been proposed to compress the connectivity information of simply connected manifold triangle meshes down to between 1.3 and 2 bits per triangle. We make an experiment with many 3D object files. The experimental result is listed in Table 1. It is almost that the storage bit for a triangle required by our algorithm is less than the one required by the Edgebreaker.

The execution time of compression stage for our algorithm and Edgebreaker is similar. In the decompression stage, Edgebreaker needs two passes to complete its decompressing work. Our algorithm needs only single pass for decompression. Therefore our algorithm works better than the Edgebreaker not only in a compact compressed data but also in a quick decompressing stage.

There is a drawback within our algorithm. When the object composed with many thin branches, the average storage bits for our algorithm is large. In such a situation, many jump cases will occur within our algorithm. There are no triangle to be removed within the jump case. Therefore, the number of jump cases must be decreased to reduce the average storage bit for a triangle. It is a hard work for improving our algorithm in the future.

References

1. R. Bar-Yehuda and C. Gotsman, Time/space tradeoffs for polygon mesh rendering, ACM Trans. Graphics, Vol. 15, No. 2, pp. 141-152, Apr. 1996.
2. M. Deering, Geometry compression, Computer Graphics Proc. SIGGRAPH'95, pp. 13-20, Aug. 1995.
3. F. Evans, S. Skiena, and A. Varshney, Optimizing triangle strips for fast rendering, Proc. IEEE Visualization'96, pp. 319-326, 1996.
4. S. Gumhold and W. Strasser, Real time compression of triangle strips for fast rendering, Proc. ACM SIGGRAPH'98, pp. 133-140, July 1998.
5. H. Hoppe, Progressive meshes, Proc. ACM SIGGRAPH'96, pp. 99-108, Aug. 1996.
6. J. Li and C.C Kuo, Progressive coding of 3D graphic models, Proc. IEEE, pp. 1052-1063, June, 1998.
7. G. Taubin and J. Rossignac, Geometric compression through topological surgery, ACM Trans. Graphics, Vol. 17, No. 2, pp. 84-115, Apr. 1998.
8. J. Rossignac, Edgebreaker: connectivity compression for triangle meshes, IEEE Trans. Visualization and Computer Graphics, Vol. 5, No. 1, pp. 47-61, 1999.

# triangles	our algorithm		Edgebreaker	
	# bits	bits/triangle	# bits	bits/triangle
50	56	1.12	84	1.68
240	174	0.72	387	1.61
392	484	1.23	650	1.66
448	302	0.67	711	1.59
534	828	1.55	877	1.64
624	534	0.86	984	1.58
768	2066	2.69	1389	1.81
1148	940	0.82	1825	1.59
1354	2498	1.84	2421	1.79
1388	2446	1.76	2336	1.68
2000	3580	1.79	3575	1.79
2268	1498	0.66	3499	1.54
2594	1940	0.75	4018	1.55
2992	4940	1.65	5027	1.68
3634	4403	1.21	6039	1.66
3640	4015	1.10	5938	1.63
3732	4194	1.12	6105	1.64
4204	4430	1.05	6890	1.64
5116	7302	1.43	8372	1.64
5660	9132	1.61	9572	1.69
6128	5941	0.97	9788	1.60
8468	16701	1.97	14871	1.76
8956	9407	1.05	14635	1.63
9558	15248	1.60	16302	1.71
10412	8094	0.78	16322	1.57
19208	15836	0.82	29919	1.56
20928	23831	1.14	34169	1.63
34404	49111	1.43	57199	1.66
39698	64512	1.63	67630	1.70
49172	37164	0.76	78431	1.60
75616	74406	0.98	123069	1.63
111000	58430	0.53	167605	1.51
average	13164.94	1.19	21231.48	1.6

Table 1: experimental results