Computer Graphics International 2011

Cluster-based Random Accessible and Progressive Lossless Compression of Colored Triangular Meshes for Interactive Visualization

Adrien Maglo · Ian Grimstead · Céline Hudelot

Abstract This paper presents a new lossless, cluster-based randomly-accessible and progressive 3D triangular mesh compression scheme. The scheme is targeted at the visualization of large meshes, supporting the encoding of vertex colors. We have extended the previous approach of [3] by using the progressive mesh encoder of [6] to compress the face clusters. The schemes were modified to prevent duplication of the boundary geometry information between cluster vertices. We propose a visualization framework that exploits the features of our codec, with solutions given to solve the problem of cracks appearing between clusters during visualization. Results demonstrate that acceptable compression ratios can be achieved using the introduced features.

Keywords Mesh compression · Random accessible · Progressive · Visualization

1 Introduction

With growing computational capabilities, the size of the numerical simulations continues to increase, with the output of such computations often being 3D meshes that need to be visualized. So, techniques must be used to reduce the quantity of 3D data to enable it to be stored, transmitted and visualized. Our progressive and randomly-accessible compression scheme enables large meshes to be compressed, for storage and interactive visualization.

Adrien Maglo · Céline Hudelot
MAS laboratory, Ecole Centrale Paris, France.
E-mail: {adrien.maglo, celine.hudelot}@ecp.fr

Ian Grimstead
Cardiff School of Computer Science & Informatics, Cardiff University, United Kingdom.
E-mail: i.j.grimstead@cs.cardiff.ac.uk

2 Previous Works

Three main types of mesh compression methods can be identified: single-rate methods, progressive methods that produce a multiresolution representation of the mesh and randomly-accessible methods that allow the decompression of only specified parts of the mesh. We do not investigate the first two types in this paper; for more information on these types, we direct the reader to mesh compression review articles such as [7].

Randomly-accessible compression schemes enable the interactive visualization of large meshes because only the requested parts are kept during the decompression. One of the approaches consists of dividing the mesh into clusters, compressing them independently and allows their decompression in an independent manner. For instance, [3] first segments a triangular mesh into ”charts” with planarity and compactness criteria. Then, each chart is independently compressed with a single rate algorithm. However, to prevent duplication of the geometry information of the chart border vertices, the boundaries or ”wires”, are encoded separately. The connectivity graph that describes how charts border each other, the ”wire-net mesh”, is encoded as a polygonal mesh and also compressed. A slightly inferior compression ratio is achieved compared to single-rate algorithms. Our scheme is based on this framework.

[8] also proposed a similar method that preserves the cache coherency of the mesh layout, thus allowing efficient and fast access to the data. However the compression ratio is lower than [3].

More recently, [4] proposed a hierarchical approach that compresses a mesh by recursively splitting it and encoding the resulting boundaries. The compression ratio is lower than the previous approaches (1.5 bpv (bits per vertex) for connectivity and 14 bpv for geometry with a 12 bit quantization), but it has the advantage of a finer decompression granularity.

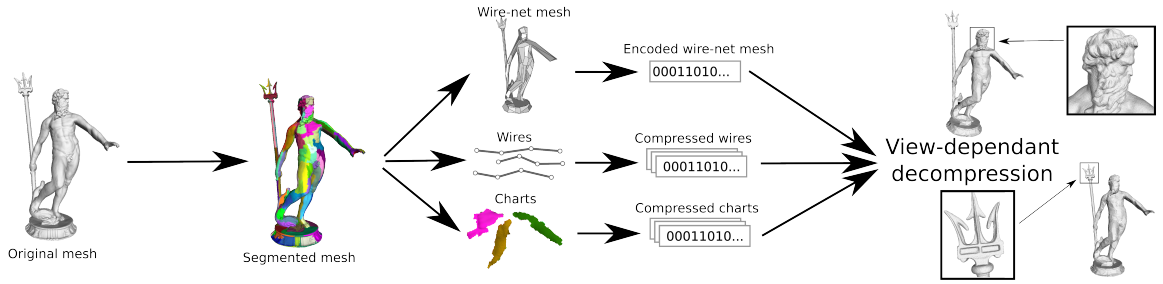

Fig. 1 Overview of our progressive and randomly accessible compression scheme.

The issue with randomly-accessible compression schemes during visualization is that non-requested parts are totally skipped—users cannot see them. However, selecting different levels of detail for each part of the model (high for the interesting parts and lower for others) could be of merit. Therefore, there have been attempts to combine progressive and multi-resolution approaches, such as [5]. The decompression granularity is high but there is a high cost in terms of compression ratio (11 bpv for connectivity and 21 bpv for geometry with a 12 bits quantization). A cluster based progressive compression approach has also been proposed by [2]. The authors segment a mesh into "meaningful parts" and then compress them with an improved version of the progressive encoder from [1]. Nevertheless, they do not mention how they process the boundary vertices to avoid geometry information duplication or cracks appearing between clusters. Besides, no compression ratios obtained with the segmented models are provided.

3 Proposed Compression Scheme

We propose a new randomly-accessible and progressive mesh compression scheme (see fig. 1) based on the original work of [3] summarised in sec. 2. We extended their work by replacing their single-rate chart encoder with the progressive encoder from [6]. In order to prevent duplication of the geometry information of the chart border vertices, we modified this progressive mesh encoder to code these vertices with the wire data.

3.1 Mesh Segmentation

The first step of our compression scheme is mesh segmentation, using the same region growing algorithm algorithm as [3]. This algorithm takes the number of charts to generate as its main parameter; it first randomly selects one seed face per chart. At each iteration, it tries connecting all the surrounding faces of each chart not already connected. It chooses the face to connect with the lowest cost; the cost function, detailed in [3], takes into account planarity, compactness and regularity in the number of triangles per chart. Once all the faces are assigned to one chart, the chart centroids are selected as the new seed faces. A new iteration is then started, at the end of this iteration, the final mesh segmentation is produced.

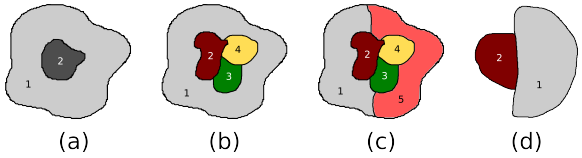

Fig. 2 Chart topological problems that may appear after the segmentation.

The segmentation quality is very important as it directly affects the final compression ratio. The more regular and flat the charts are, the better the compression ratio will be.

3.2 Topological Problem Handling

As described in [3], the connectivity between charts is encoded as a polygonal mesh. Therefore, the chart topology must match a set of requirements. If at the end of the segmentation the following cases are encountered, the segmentation must be modified:

- An island chart, a chart surrounded by a single neighbouring chart (see fig. 2 a).
- A donut shaped chart, a chart that fully surrounds several charts (see fig. 2 b).
- Two charts sharing two distinct boundaries (see fig. 2 c).
- A border chart with only one neighbouring chart (see fig. 2 d). This case is not cited by [3] but can appear when compressing a mesh with boundaries.

We would like to highlight other topological problems that were not mentioned in [3]. Meshes with a high genus can be problematic as their segmentation can generate many donut shaped charts, which must split or merged. Moreover, in the case of the original mesh containing holes, the segmentation can create non-manifold charts. Our chart compression algorithm is restricted to handling manifold meshes,

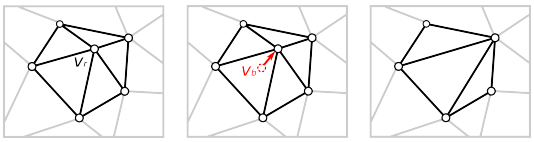

Fig. 3 Vertex removal operation from [1]. From left to right: the original patch, the center vertex removal with the geometry prediction error from the patch barycenter (red) and the retriangulation.

so some faces must be connected to another neighbouring chart to create a fan, making this chart manifold.

3.3 Wire Compression

To compress the wires we also used a simple linear predictor, such as [3], for the vertex positions and color components. We encoded the prediction errors with an arithmetic coder. We made the choice to build all the wire data and then generate a static model stored with the compressed data for the arithmetic coder.

3.4 Chart Compression

We reused the progressive mesh compression scheme that can handle vertex colors of [6], which is based on [1]. The main idea of this algorithm is to decimate the original mesh in several deterministic patch by patch traversals, also followed by the decoder. A patch is a set of adjacent faces with a single vertex at its center (v_r). To simplify the mesh, each patch center vertex with a valence less than 7 is removed; the patch is then retriangulated (see fig. 3). Connectivity encoding consists of encoding the centre vertex valence. Geometry encoding consists in taking the patch barycentre (v_b) as the vertex position predictor and encodes the error prediction in the Frenet Coordinate Frame. Considering the border vertices, only vertices with a valence of 3 or 4 are removed. They are encoded with specific symbols that are different from valence 3 and 4 inner vertices.

We modified the compressor and decompressor to integrate it in our randomly-accessible framework. First, we prevent wire-net mesh vertices from being removed during the decimation passes in order to be able to locate the boundaries after decompression.

Besides, in order to not duplicate the border vertices geometry data in the two adjacent charts, we created two new connectivity symbols to encode the border vertices belonging to a wire: one for border valence 3 vertices and one for valence 4 vertices. These symbols are followed by another symbol that allows to find the index (i_r) of v_r in the wire. This symbol is determined by first computing v_b. Then, the index (i_n) of the vertex in the wire that is the nearest from v_b is computed. The encoded symbol S_c, corresponds to $i_r - i_n$

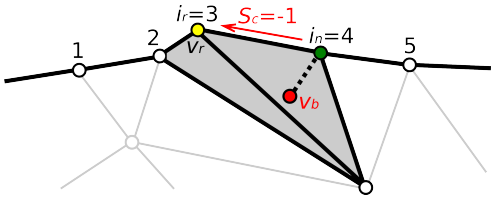

Fig. 4 Encoding chart border vertices. The numbers are the vertices indices in the wire.

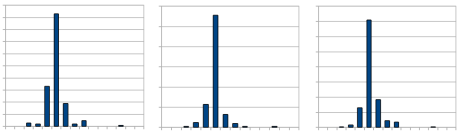

Fig. 5 Typical distributions of S_c.

(see fig. 4). As can be seen in fig. 5, for most cases S_c is null. Therefore, it can be efficiently entropy encoded. The achieved gains can be found on tab. 1 (normal vs no wire columns).

3.5 Experimental Compression Results

We compared our compression results with the progressive approach in [6] to compare the costs introduced by the random-accessibility. As each chart is compressed independently in our approach, the chart compression can easily be multi-threaded. The compression time is on average twice larger than [6] because of the segmentation and topology problem handling steps of our scheme.

The compression ratio are highly dependent of the segmentation quality and the number of charts. Generally, the bigger the charts are, the better the ratios are. With about 20,000 faces per chart, the compression ratios are also very good compared to [4,5]. Figures are provided in table 1.

4 Decompression Scheme

The main advantage of our scheme is that it can decompress any chart at any requested level of detail. We created a visualization framework that exploits these features.

4.1 Selecting the Levels of Detail of Each Chart

To select an appropriate level of detail l for each chart, at decompression time we extract each chart's average normal $\mathbf{d}_c$ from its coarse level. l is chosen by calculating a scalar product between $\mathbf{d}_c$ and the view direction $\mathbf{d}_v$ (see fig. 6 a). If the scalar product value is greater than a set parameter α, the chart is decompressed at its finest level. Otherwise, the level of detail is given by a linear curve for which the

Table 1 Experimental compression results with a 12 bits quantization. The last column gives the cost of the normal compression with our scheme compared to [6]. The times have been obtained on a desktop computer with an Intel Core i7 processor at 2.80GHz and 8Go of RAM with 8 threads.

Model	# of vertices	Our scheme						[6]			Comp. ratio cost
		# of charts	Comp. ratio (bpv)			Times (s)		Comp. ratio (bpv)	Times (s)		
			Normal	No wire	Geom. constraints	Comp.	Decomp.		Comp.	Decomp.	
Ramesses	826266	333	15.5	16.5	16.7	99	20	12.4	48	101	25.3%
		166	14.3	15.0	15.2						15.5%
		84	13.5	14.1	14.3						9.6%
		37	13.0	13.4	13.5						5.3%
		21	12.6	12.9	13.0						2.1%
Neptune	2003932	204	10.4	11.0	11.5	250	48	9	123	240	14.7 %
XYZ RGB Dragon	3609600	362	9.4	10.0	10.4	475	84	7.8	226	526	20.4 %
XYZ RGB Thai Statue	4999996	526	11.3	11.8	12.5	950	125	-	-	-	-
EDF tank (colored)	230068	114	19.8	22.5	24.5	57	5	20.4	13	25	-2.8%

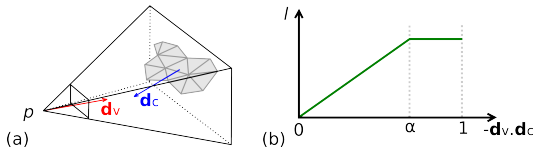

Fig. 6 Choosing the decompression level of detail (l) for each chart.

highest level of detail corresponds to α and the lowest to 0, when the view direction is perpendicular to the chart normal (see fig 6 b).

The adaptation scheme also takes into account the screen resolution in order to avoid excessively refining a mesh when the new details cannot be observed at the given screen resolution. This is achieved by computing for a given chart level of detail the average surface of a triangle S_t. Given the screen resolution, the view frustum and the distance between the viewpoint and the chart d, we determine the number of triangles with the surface S_t on a plane perpendicular to the view direction and far of d we could display. Then, we deduce the number of pixels per triangle. If this number is below a threshold, the mesh refinement is stopped.

4.2 Dealing with Boundaries

One significant drawback of the progressive and randomly-accessible mesh compression schemes based on clusters is that they produce cracks between parts that may be observed as holes in the original model. As a first solution to solve this problem, we implemented a quick chart stitching algorithm. For a given boundary between two charts, vertices of the boundary with the highest number of vertices are moved to the positions of the vertices of the boundary with the lowest number of vertices. The remaining holes are then triangulated; the cracks vanish but some visualization artifacts still remain. A better stitching can be achieved by checking that v_r is not inside the new border patch after the retriangulation, but this is carried out at the cost of a lower compression ratio (see table 1, geom. constraints column).

4.3 Decompression Time

The decompression can be multi-threaded because each chart is processed independently. Therefore, compared to [6], a faster full decompression is achieved in our experiments (see tab. 1).

5 Conclusion

We presented our randomly accessible and progressive lossless compression algorithm that can handle vertex colors. Targeted for the visualization of big meshes, it combines the multiresolution and the random accessibility features with an acceptable cost in term of compression ratio. The compression and decompression can be multi-threaded. Future work includes cache techniques and user need predictions in our visualization framework.

Acknowledgements This work has been supported by French National Research Agency (ANR) through COSINUS program (project COLLAVIZ n°ANR-08-COSI-003).

References

1. Alliez, P., Desbrun, M.: Progressive compression for lossless transmission of triangle meshes. In: 28th annual conference on Computer graphics and interactive techniques, SIGGRAPH '01 (2001)
2. Cheng, Z.Q., Liu, H.F., Jin, S.Y.: The progressive mesh compression based on meaningful segmentation. The Visual Computer **23** (2007)
3. Choe, S., Kim, J., Lee, H., Lee, S.: Random accessible mesh compression using mesh chartification. IEEE Transactions on Visualization and Computer Graphics **15** (2009)
4. Courbet, C., Hudelot, C.: Random accessible hierarchical mesh compression for interactive visualization. In: Symposium on Geometry Processing, SGP '09 (2009)
5. Kim, J., Choe, S., Lee, S.: Multiresolution random accessible mesh compression. Computer Graphics Forum **25** (2006)
6. Lee, H., Lavoué, G., Dupont, F.: New methods for progressive compression of colored 3D Mesh. In: International Conference on Computer Graphics, Visualization and Computer Vision (WSCG) (2010)
7. Peng, J., Kim, C.S., Kuo, C.C.J.: Technologies for 3d mesh compression: A survey. Journal of Visual Communication and Image Representation **16** (2005)
8. Yoon, S.e., Lindstrom, P.: Random-accessible compressed triangle meshes. IEEE Transactions on Visualization and Computer Graphics **13** (2007)