Adaptive coarse-to-fine quantization for optimizing rate-distortion of progressive mesh compression

Ho Lee[1], Guillaume Lavoué[2], Florent Dupont[1]

Université de Lyon, CNRS
[1]Université Lyon 1, LIRIS, UMR5205, F-69622, France
[2]INSA-Lyon, LIRIS, UMR5205, F-69621, France
Email: {hlee, glavoue, fdupont}@liris.cnrs.fr

Abstract

We propose a new connectivity-based progressive compression approach for triangle meshes. The key idea is to adapt the quantization precision to the resolution of each intermediate mesh so as to optimize the rate-distortion trade-off. This adaptation is automatically determined during the encoding process and the overhead is efficiently encoded using geometrical prediction techniques. We also introduce an optimization of the geometry coding by using a bijective discrete rotation. Results show that our approach delivers a better rate-distortion behavior than both connectivity-based and geometry-based compression state of the art methods.

1 Introduction

Nowadays, 3D graphics models are widely used in many applications such as Computer-Aided Design, scientific visualization, virtual reality, video gaming and e-commerce. As the considerable number of applications shows, 3D geometric models are becoming as popular as the other multimedia data like audio, sound, image and video. This amount of 3D models has rapidly grown to satisfy needs of representing objects or scenes with more and more realism. This increasing size of 3D data and the ever growing use of Web-based applications have introduced the necessity of efficient compression algorithms in order to reduce the storage size and transmission time over the network. In this context, progressive compression techniques are particularly useful, since they enable a quick rendering of a coarse approximation of the original model with a small number of bits, and then deliver more refined versions as more bits are transmitted. In other words, progressive compression permits to transmit different levels of details (LOD) in a coarse-to-fine way. Before to describe related works, we recall here that 3D triangular meshes have two main components : *geometry* and *connectivity*. Geometry consists of vertex positions in the 3D space and connectivity describes how theses positions are connected together.

1.1 Previous works on progressive compression

Firstly, the concept of progressive compression was introduced by Hoppe [9]. This new mesh representation, so-called progressive mesh, consists in simplifying successively a given mesh by using edge-collapses, which remove one vertex and two faces adjacent to the edge. The reconstruction (at the decompression stage) is accomplished by the inverse operation called vertex split. Each edge collapse is chosen based on geometric criteria to obtain the better approximations of the original mesh, hence the reconstruction needs extra bits for the localization of the inverse operations. This method has been extended by several researchers to improve the compression ratio. Taubin et al. [16] reduced the connectivity cost to 7-10 bits-per-vertex (bpv) by using forest split operation instead of vertex split operation. Pajarola et al. [12] proposed compressed progressive meshes. In their representation, vertex splits are applied in batches, performing 7 bpv for the connectivity. The geometry of each vertex is coded based on the butterfly-like prediction using its neighbors. Karni et al. [11] proposed a similar approach: the authors first built a sequence of edges which traverses all mesh vertices. Due to good locality and continuity properties of this sequence, applying edge collapses

between two adjacent edges of the sequence leads to an improvement of compression rates and also rendering speed. In their work, Cohen-Or et al. [6] proposed the patch coloring technique for progressive transmission. This algorithm removes iteratively a set of vertices. The hole induced by each vertex removal is re-triangulated in a deterministic way. This algorithm encodes the connectivity with an average of 6 bpv.
Based on the existing valence-driven single rate approaches [17] [2], Alliez and Desbrun [1] extended the valence-driven scheme for progressive encoding. The authors iteratively applied decimating conquest alternating with cleansing conquest to get different levels of details. Decimation conquest traverses the mesh from patch to patch based on a deterministic gate-based traversal; when the valence of the front vertex of the actual gate is less or equal to 6, this vertex is removed and the patch is re-triangulated. Similarly, cleansing conquest decimates only vertices of valence 3. During conquests, valences of removed vertices and some supplementary null-patch symbols are encoded to allow the exact reconstruction of the connectivity. Connectivity can be compressed to an average of 3.7 bpv.
All the algorithms described above are connectivity-driven techniques, meaning that the priority is given to the connectivity coding. Therefore, the coding of geometry is often not optimal because the geometry redundancy reduction is constrained by the traversal of the connectivity coding. As geometry data are often larger than connectivity data, geometry-driven algorithms give very good results and have been heavily studied more recently.
Gandoin and Devillers [7] proposed the first geometry-driven algorithm based on the kd-tree space subdivision. This algorithm is performed by two passes. The first pass consists of encoding only geometry data. They recursively divide space in two cells until there is only one vertex in each cell. The number of vertices in one cell is then encoded. The second pass encodes the connectivity change caused by each cell subdivision. In terms of compression ratio, this compression technique outperforms connectivity-driven algorithms and even can compete with the single rate coders like [8] [14] [17]. Peng and Kuo [13] improved the kd-tree algorithm [7] by using octree cell subdivision. Each cell is divided into eight child-cells and instead of the number of vertices, they encode whether each child cell is empty or not. Using an efficient prediction for both connectivity and geometry coding, they achieved an improvement of 10 to 20% comparing to [7].

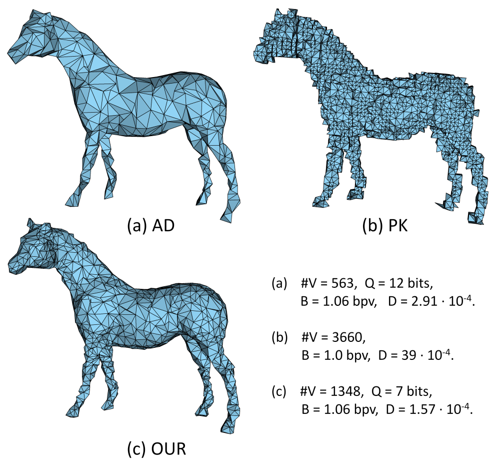
(a) AD
(b) PK
(c) OUR
(a) #V = 563, Q = 12 bits, B = 1.06 bpv, D = $2.91 \cdot 10^{-4}$.
(b) #V = 3660, B = 1.0 bpv, D = $39 \cdot 10^{-4}$.
(c) #V = 1348, Q = 7 bits, B = 1.06 bpv, D = $1.57 \cdot 10^{-4}$.

Figure 1: The Horse model resulting from different approaches at a similar bit rate: Alliez and Desbrun [1](AD), Peng and Kuo [13](PK) and our algorithm.

1.2 Overview and contributions

The increasing use of Web-based applications targeting low computational power devices such as mobile phones or PDAs calls for better approximation of model for the same amount of bits transmitted. Therefore, the attention starts now to focus on the improvement of the quality of intermediate meshes.
Usually, geometry-driven algorithms perform better than connectivity-driven algorithms in terms of lossless compression ratio. However, the approximation quality of meshes at low resolutions is poor, since these techniques produce stair-like appearances in the intermediate meshes, due to the excessive number of vertices regarding the low quantization precision. On the contrary, the quantization precision is higher than the necessity for connectivity-guided algorithms at low bit rate. In this context, we propose a novel connectivity-driven

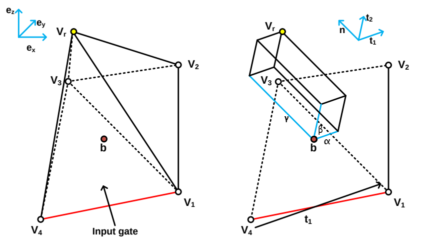

Figure 2: Both barycentric prediction and Frenet coordinate frame are used for geometry coding in [1]. Input gate is colored in red. The position of barycenter b is calculated from boundary vertices of the current patch. Vertex to be encoded, V_r, is expressed as an offset (α, β, γ) from barycenter b in the local coordinate frame.

approach that combines these two different schemes in order to enhance the quality of intermediate meshes in terms of rate-distortion (R-D) performance.

Figure 1 illustrates a result of our algorithm: at a given low bit rate, the quality of the decompressed model is much better than state-of-the-art algorithms [1] and [13].

Our contributions are as follows :

1. Observing that the high precision of vertex positions imposed by the initial quantization is not necessary for meshes of low resolution, we adopt an adaptive quantization scheme for intermediate meshes in order to improve the progressive mesh coder in terms of R-D performance.
2. We propose also an improvement of the geometry coding of [1], by using a bijection between two sets of 3-tuples of integers through a rotation operation. This technique brings a compression gain between 3 and 25%.

This paper is organized as follows: in Section 2, we present our geometric coder based on the bijection. Section 3 deals with improvement of quality of intermediate meshes using adaptive quantization. In Section 4, we give experimental results, and a conclusion follows.

2 Improvement of geometry coder

In general, connectivity-driven methods yield better quality of intermediate meshes than geometry-driven ones. In this context, our work is based on the algorithm of Alliez and Desbrun [1] which is the most efficient connectivity-driven algorithm. For the geometry coding, Alliez and Desbrun first applied a global and uniform quantization to the coordinates of the mesh vertices. Then, they used both the barycentric prediction and the approximate Frenet coordinate frame, separating normal and tangential components to further optimize the bit rate. To build the base vectors of this local coordinate frame, they first approximated the normal vector from normals of triangles of the patch and the two other base vectors, t_1 and t_2, are computed by using this normal vector and the gate of the patch, as shown in Figure 2. The normal n and the barycenter b approximate locally the tangent plane of surface. Then, the vertex V_r to be encoded is projected on the new base vectors and becomes $V_r = b + \alpha \cdot t_1 + \beta \cdot t_2 + \gamma \cdot n$. The new coordinates (α, β, γ) represent the relative position of the vertex V_r from the barycenter in the local Frenet frame. As (α, β, γ) obtained by the projection are usually floating numbers, these coordinates are rounded to the nearest signed integers in order to be effectively encoded. This rounding operation, however, can introduce a loss of information, meaning that the exact position of V_r cannot be reconstructed at the decompression stage. To overcome this inconvenience, the authors introduced a post-quantization step. Yet, this post-quantization adds supplementary information to the geometry coder, resulting in an increased compressed file size.

The key idea to improve the geometry coding consists in eliminating this post-quantization step, while allowing to retrieve the exact position of the encoded vertex. The projection used in [1] to find the new coordinates in the Frenet frame can be considered as a rotation of the coordinate system. Hence, the elimination of the post-quantization step turns into finding the bijection between two sets of 3-tuples of integers through a rotation.

In [4], Carstens et al. proposed such a bijection. However, the objective of their work was not related to 3D mesh compression. Therefore, we modified their scheme to optimize the compression bit rates by minimizing angles of rotation to encode. First, they proved that a rotation matrix, R, is equal to the

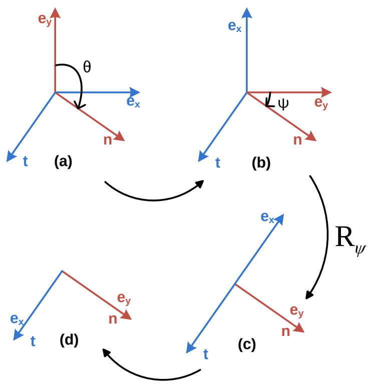

Figure 3: An example of minimization of rotation angle. (a) Initial coordinate system (e_x, e_y) and rotated coordinate system (t, n). Initial rotation angle is θ. (b) Interchange of axis between e_x and e_y. (c) After rotation by angle ψ, axis e_y and n are aligned. (d) After changing orientation of axis e_x, transformation of coordinate system with a minimum angle is achieved.

product of three shear matrices, $S_3 \circ S_2 \circ S_1$.

$$\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} = \begin{pmatrix} 1 & \lambda_1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \lambda_2 & 1 \end{pmatrix} \begin{pmatrix} 1 & \lambda_3 \\ 0 & 1 \end{pmatrix} \tag{1}$$

where θ is the rotation angle in the counterclockwise sense, $\lambda_1 = \lambda_3 = -\tan(\theta/2)$ and $\lambda_2 = \sin(\theta)$.

Then, they proposed a bijection method by using a novel rounding scheme based on these three shear matrices.

$$x' = [S_3^{-1} \circ [S_2^{-1} \circ [S_1^{-1} \cdot x]]] \tag{2}$$

The notation [ ] denotes the rounding process of coordinates to the nearest integers. For a given vector, $x = (i, j)^T$, where i and j are integers, the principle of the bijection is to first applied a multiplication between the inverse matrix S_1^{-1} and x. Then, the components of the resulting vector from the multiplication are rounded to the nearest integers. After performing identically with S_2^{-1} and S_3^{-1}, x' which corresponds to x in the rotated coordinate system is obtained. The authors demonstrated the bijection method in 2D for simplicity. We have applied their work for our case in 3D. In 3D, the rotation of the coordinate system can be seen as a composition of three rotations. In other words, any rotation may be described using three angles, θ, ϕ and ψ. These angles are calculable from the base vectors of Frenet frame, $(t_1,\ t_2,\ n)$. Hence, the bijection between two sets of 3-tuples of integers is possible using, this time, nine shear matrices as explained above in equations (1) and (2) based on three angles, θ, ϕ and ψ. The resulting 3-tuple of integers, which represents the relative coordinates of the vertex V_r in the local Frenet frame, is then encoded.

However, in terms of compression efficiency, this bijection method is not optimal. Assuming that rotation angles affect distributions of the coordinates, we propose a more efficient bijection to further improve the geometry coding by minimizing these angles. When calculating each rotation angle, we minimize it by interchanging axis and by changing the orientation of the axis of the coordinate system. The axis interchanging and the axis orientation change are performed in a deterministic way, allowing the coder and the decoder to obtain the same angles, without any supplementary information to encode. An example of angle minimization in 2D is illustrated in Figure 3. Our scheme allows to obtain almost all rotation angles comprised between -$\pi/4$ and $\pi/4$. This new bijection method brings a supplementary compression gain for all models. Table 1 shows geometry compression rates of our geometric coder before (Our(B)) and after angle optimization (Our(BA)), compared to the result of [1]. All models are quantized using 10 bits. Our geometric coders outperform Alliez and Desbrun's coder and angle optimization brings an additional compression gain between 0.2 and 0.5 bpv.

Table 1: Comparison of geometry compression rates in bpv.

Models	# v	AD [1]	Our(B)	Our(BA)
Tiger	2738	12.7	12.1	**11.7**
Mannequin	11703	10.0	9.8	**9.3**
Venusbody	11362	10.2	8.6	**8.1**
Torus	36450	3.6	3.0	**2.8**

3 Rate-Distortion optimization by adaptive quantization

3.1 Objectives and outline of algorithm

Rate-Distortion (R-D) performance is an important criterion to evaluate the efficiency of progressive compression. The term distortion indicates geometric error of intermediate meshes comparing to the original mesh, and the other term, rate, designates the required number of bits for the reconstruction.

At the beginning of geometry coding, quantization is performed to the mesh vertices to reduce the geometry data amount. Quantization is a lossy procedure and introduces a geometric error. To avoid a significant deformation of the original 3D object, typically 8 to 12 bits are used as quantization precision. This high precision is effectively needed for meshes at high resolutions where the number of elements is important. However, this high precision is not necessary for meshes at low resolutions, which are composed of low numbers of elements. Coarse meshes which have been quantized using different precision (3 to 12 bits) give quite the same quality in terms of distortion. At very low resolution, the geometric error induced by quantization is insignificant compared to that caused by the reduced number of mesh elements.

Of course, quantization influences also the size of

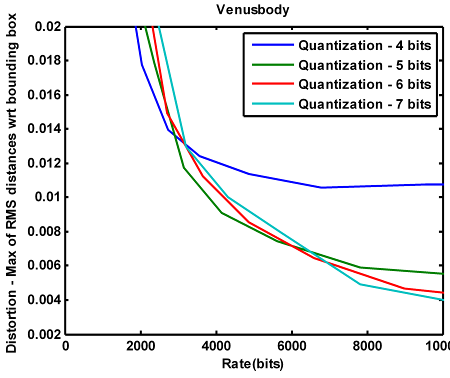

Figure 4: R-D curves with different bits (4, 5, 6 and 7 bits) used for quantization applied on Venusbody. Each curve possesses the best R-D performance at the specific range of bit rates.

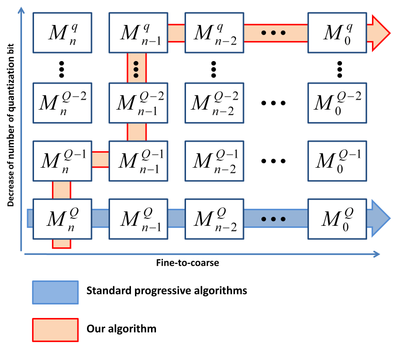

Figure 5: Our algorithm (red arrow) and standard progressive algorithms (blue arrow).

the compressed file. Therefore, applying a quantization with few bits can improve the R-D performance at low bit rates. Figure 4 shows the R-D curves of the model Venusbody respectively quantized using 4, 5, 6 and 7 bits. For the distortion, the maximum of two RMS distances normalized to the bounding box diagonal is measured using the METRO tool [5]. In Figure 4, we can observe that the R-D curve of the progressive algorithm using 4-bit quantization shows the best R-D performance until 3000 bits. Then, the R-D curve of 5-bits quantization becomes the best up to 6000 bits, and so on. Relying on this observation, our idea to improve the R-D performance is to apply an adaptive quantization for intermediate meshes. Figure 5 describes our algorithm, comparing to other connectivity-driven techniques. Traditionally, a progressive compression coder decimates iteratively an initial mesh, M_n^Q, whose vertices are quantized using Q bits. After n iterations, the base mesh, M_0^Q is obtained. Our algorithm decreases also quantization precision during coding process. Therefore, an improvement of R-D performance is possible by finding the best way, combining a series of decimation processes and a series of diminutions of quantization precision. This algorithm needs two main issues:

- Automatic determination of next operation which leads to the better R-D performance.
- Decreasing quantization precision and efficient encoding of the inverse operations.

These operations are processed at the encoding step.

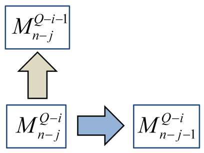

Figure 6: For a given intermediate mesh, a determination is made between a decimation operation and a diminution of quantization resolution.

3.2 Optimal quantization determination

As shown in Figure 6, for a given intermediate mesh, M_{n-j}^{Q-i}, obtained after j iterations of simplification and after reduction of i bits of quantization, the next operation can be a decimation, leading to M_{n-j-1}^{Q-i}, by applying one iteration of simplification, or a diminution of quantization resolution, leading to M_{n-j}^{Q-i-1}. Between these two cases, we select the one which improves more the R-D performance. To perform automatically this determination, the distortion increase ΔD, the coding bit increase ΔB, and their ratio $R = \Delta D/\Delta B$, are calculates for both cases.
For this task, meshes M_{n-j}^{Q-i-1} and M_{n-j-1}^{Q-i} are constructed. Then, for the decimated mesh M_{n-j-1}^{Q-i}, we compute ΔB_{dec} by evaluating the entropy of connectivity and geometry information. The distortion ΔD_{dec} is calculated using the decimated mesh and the original mesh. Similarly, the coding bit ΔB_{quan} (See Section 3.3) and ΔD_{quan} are obtained. The determination is then possible by comparing $R_{dec} = \Delta D_{dec}/\Delta B_{dec}$ and $R_{quan} = \Delta D_{quan}/\Delta B_{quan}$. If $R_{dec} < R_{quan}$, we choose the decimation as the next operation, else the diminution of quantization resolution is selected.
Note that the R-D performance can be optimized in terms of various geometric criterion, like RMS distance, Hausdorff distance or any user-defined metric.

3.3 Diminution of quantization resolution

Here, we present the decrease of the quantization resolution and the technique to encode efficiently the inverse operation. Initial quantization using b bits, divides the axis aligned bounding box of the input mesh into $2^b * 2^b * 2^b$ cubic cells. Then, a vertex inside one cell is moved to its center position. If we replace b by $b-1$, the dimension of the each cube becomes twice along the three axis, and each vertex is moved to the center of the new cube. Therefore, the diminution of quantization resolution can be considered as an octree structure. Initial cubes are the child-cells and the new bigger cube is the parent-cell of these child-cells. At the decoding step, the inverse operation has to be processed. Basically, the index of the corresponding child-cell has to be encoded (8 possibilities).

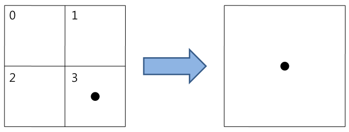

Figure 7: A 2D example of diminution of quantization resolution. In this case, the index 3 is encoded.

Each displacement of vertex needs 3 bits without any prediction. We adopt a prediction method from geometry-driven compression algorithms. Among existing techniques [13] [10] [15], we adopt the prediction of Peng et al. [13], which reveals to be the most efficient method for our case.
Indices of child-cells are reordered taking into account their priority values calculated using vertices in the vicinity of parent-cell and their distances to the centroid of the parent-cell.
The effectiveness of the prediction is illustrated in Figure 8 which illustrates the indices distribution without and with prediction.

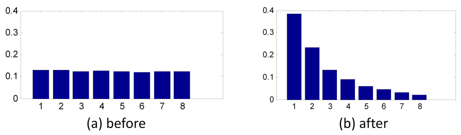

Figure 8: Histogram of child-cell indices before and after prediction.

4 Experimental results

4.1 Lossless compression

Table 2 shows lossless compression results applied on various reference meshes using 10 and

12 bits coordinate quantization. We compare our coding bit rates with the algorithm of Alliez and Desbrun [1](AD). Some results obtained by the geometry-driven octree based algorithm of Peng and Kuo [13](PK) are also given. In terms of overall compression rate, our improved geometric coder (BA) brings a gain between 2.9 to 25.0% (12.1% on average for meshes in Table 2) compared to (AD). We can see that the adaptive quantization (BAQ) demands extra bits regarding (BA). However, for the most part, this overhead is slight and for the models in Table 2, the compression rate of (BAQ) is always better than (AD). However, Peng and Kuo's approach is still better than ours. Lossless compression rates of (AD) are calculated using the software provided by Pierre Alliez. Yet, this software does not give rates of intermediates meshes. Therefore, for the comparison of R-D performance, we use our implementation which yields similar results.

4.2 R-D performance comparison

Figure 9 and Figure 10 show respectively the R-D curves for the Venusbody mesh of 11,362 vertices and the Venushead mesh of 8,268 vertices, both quantized using 10 bits for the lossless precision. The vertical axis is the maximum of two RMS distances with respect to the bounding box. In these figures, we see that our algorithm combining the bijection and the adaptive quantization, (BAQ), improves the approximation quality compared to our approach (BA) using only the bijection and AD aproach. Figure 12 illustrates four different levels of details of (BAQ) and (AD) in similar bits applied on the Venusbody model. Our method gives the better result regarding the distortion. In Figure 11, we compare our algorithm with the octree coder of Peng and Kuo [13](PK) for the Rabbit mesh of 67,039 vertices quantized using 12 bits. The approach of Alliez and Desbrun performs significantly better than Peng and Kuo's algorithm and again, our algorithm gives the best result.
For those cases, we improved the R-D performance in terms of RMS distance. However, our algorithm is generic enough to enhance the R-D performance using any geometric error metric such as Hausdorff distance or even user-defined perceptual metrics.

Table 2: Compression rates for various reference meshes in bpv.

Models	# v	Q bit	PK	AD	Our(BA)	Our(BAQ)
Fandisk	6475	10	13.3	17.4	15.6	16.7
Venusbody	11362	10	–	14.1	11.7	12.0
Horse	19851	12	16.6	20.9	20.3	20.6
Torus	36450	10	–	4.0	3.0	3.2
Torus	36450	12	11.8	–	4.8	5.7
Mannequin	11703	10	–	13.6	13.0	13.5
Tiger	2738	10	–	15.3	14.0	14.3
Rabbit	67039	12	14.8	–	16.2	16.4
Dinosaur	14070	10	–	18.5	15.2	15.6
Foot	10016	12	–	25.8	21.7	21.8
Venushead	8286	10	–	19.5	18.1	18.2

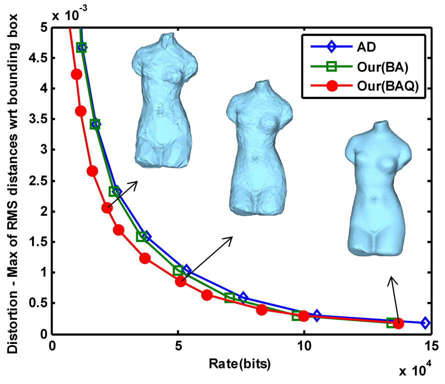

Figure 9: Rate-Distortion curve for the Venusbody.

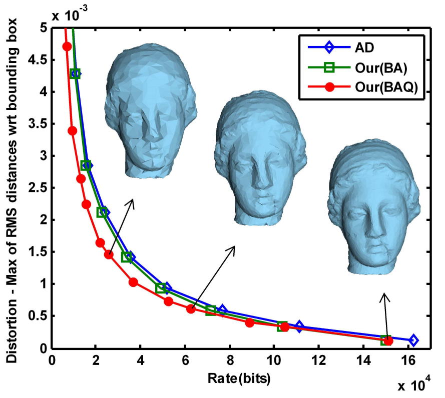

Figure 10: Rate-Distortion curve for the Venushead.

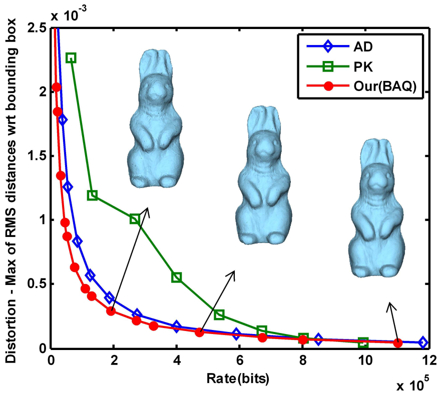

Figure 11: Rate-Distortion curve for the Rabbit.

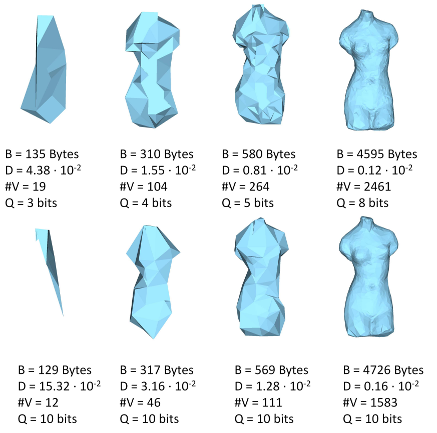

Figure 12: Comparison of intermediate meshes in similar bits between Alliez and Desbrun's algorithm (bottom) and our algorithm (top) for the Venusbody.

5 Conclusion

We have presented a new method for progressive compression based on geometry coding optimization and adaptive quantization. Although our method is basically connectivity-driven, we have adopted mechanism of geometry-guided algorithm by using the decrease of quantization precision. This mixed connectivity-geometry scheme allows us to optimize the trade-off between the number of vertices and the quantization precision for a given bit budget, improving the R-D performance. Our algorithm allows to improve the quality of intermediate meshes in terms of any metric such as geometric distances or user-defined perceptual metrics.
Future work will concern extension of our work to take into account associated properties like colors or normals, and improvement of the determination of quantization precision to further optimize the R-D performance.

Acknowledgement

We would like to thank Pierre Alliez for providing us with the software and Jingliang Peng for sending us the intermediate models from his work. This work has been supported by French National Research Agency (ANR) through COSINUS program (project COLLAVIZ n°ANR-08-COSI-003).

References

[1] P. Alliez and M. Desbrun. Progressive compression for lossless transmission of triangle meshes. In *ACM SIGGRAPH,* 198–205, 2001.

[2] P. Alliez and M. Desbrun. Valence-driven connectivity encoding for 3D meshes. In *Eurographics,* 480–489, 2001.

[3] C. L. Bajaj, V. Pascucci and G. Zhuang. Single resolution compression of arbitrary triangular meshes with properties. In *Proceedings of the Data Compression Conference,* 247–256, 1999.

[4] H.-G. Carstens, W. A. Deuber, W. Thumser and E. Koppenrade. Geometrical bijections in discrete lattices. *Combinatorics, Probability and Computing,* 8:109–129, 1999.

[5] P. Cignoni, C. Rocchini and R. Scopigno. Metro: Measuring error on simplified surfaces. *Computer Graphics Forum,* 17(2):167–174, 1998.

[6] D. Cohen-Or, D. Levin and O. Remez. Progressive compression of arbitrary triangular meshes. In *IEEE Visualization Conference Proceedings,* 67–72, 1999.

[7] P.-M. Gandoin and O. Devillers. Progressive lossless compression of arbitrary simplicial complexes. *ACM Transactions on Graphics,* 21(3):372–379, 2002.

[8] S. Gumhold and W. Straßer. Real time compression of triangle mesh connectivity. In *ACM SIGGRAPH,* 133-140, 1998.

[9] H. Hoppe. Progressive meshes. In *ACM SIGGRAPH,* 99–108, 1996.

[10] Y. Huang, J. Peng, C.-C.J. Kuo and M. Gopi. Octree-based progressive geometry coding of point clouds. In *Eurographics Symposium on Point-Based Graphics,* 103–110, 2006.

[11] Z. Karni, A. Bogomjakov and C. Gotsman. Efficient compression and rendering of multiresolution meshes. In *IEEE Visualization Conference Proceedings,* 347–354, 2002.

[12] R. Pajarola and J. Rossignac. Compressed progressive meshes. *IEEE Transactions on Visualization and Computer Graphics,* 6(1):79–93, 2000.

[13] J. Peng and C.-C.J. Kuo. Geometry-guided progressive lossless 3D mesh coding with octree (OT) decomposition. In *ACM SIGGRAPH,* 609–616, 2005.

[14] J. Rossignac. Edgebreaker: Connectivity compression for triangle meshes. *IEEE Transaction on Visualization and Computer Graphics,* 5(1):47–61, 1999.

[15] R. Schnabel and R. Klein. Octree-based point-cloud compression. In *Eurographics Symposium on Point-Based Graphics,* 111–120, 2006.

[16] G. Taubin, A. Guéziec, W. Horn and F. Lazarus. Progressive forest split compression. In *ACM SIGGRAPH,* 123–132, 1998.

[17] C. Touma and C. Gotsman. Triangle mesh compression. In *Proceedings of Graphics Interface,* 26–34, 1998