



# A Wavelet-Based Progressive Compression Scheme For Triangle Meshes : Wavemesh

Sébastien Valette, Rémy Prost

## ► To cite this version:

Sébastien Valette, Rémy Prost. A Wavelet-Based Progressive Compression Scheme For Triangle Meshes : Wavemesh. IEEE Transactions on Visualization and Computer Graphics, 2004, 10 (2), pp.123-129. 10.1109/TVCG.2004.1260764 . hal-00537014

**HAL Id: hal-00537014**

**<https://hal.science/hal-00537014>**

Submitted on 17 Nov 2010

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# A Wavelet-Based Progressive Compression Scheme for Triangle Meshes : Wavemesh

Sébastien Valette and Rémy Prost, *Member, IEEE*  
CREATIS\*, Lyon, France

**Abstract**—This paper proposes a new lossy to lossless progressive compression scheme for triangular meshes, based on a wavelet multiresolution theory for irregular 3D meshes. Although remeshing techniques obtain better compression ratios for geometry compression, this approach can be very effective when one wants to keep the connectivity and geometry of the processed mesh completely unchanged. The simplification is based on the solving of an inverse problem. Optimization of both connectivity and geometry of the processed mesh improves the approximation quality and the compression ratio of the scheme at each resolution level. We show why this algorithm provides an efficient means of compression for both connectivity and geometry of 3D meshes and it is illustrated by experimental results on a various set of reference meshes, where our algorithm performs better than previously published approaches for both lossless and progressive compression.

**Index Terms**—wavelets, irregular meshes, compression, multiresolution

## I. INTRODUCTION

NOWADAYS, 3D models are used in a wider and wider range of applications, from Computer Aided Design (CAD) to online shopping, where transmission time can be a key issue for large models. In this context, progressive transmission gives the opportunity to render quickly a rough approximation of the original model, which can be refined as more data arrive at the receiver end, until the 3D object is exactly reconstructed. Therefore, progressive transmission algorithms must perform the best possible tradeoff between compression, that is the smallest bit rate for the original model, and distortion, that is the quality of the coarser models. 3D triangular models have two main components: the geometry, i.e. the vertex coordinates, and the connectivity, i.e. the way vertices are linked to form the faces of the mesh. The next section of this paper addresses briefly progressive transmission algorithms for 3D triangular models. In section III, we present the overview of a new progressive transmission algorithm that we call Wavemesh, based on a wavelet scheme for 3D irregular meshes [21], [22], which is an extension of the regular scheme proposed by Lounsbery [15]. Sections IV and V are respectively dedicated to connectivity and geometry compression of the models. Section VI illustrates the efficiency of our algorithm, from lossy progressive transmission to lossless compression of the original model, and a conclusion follows.

\*:Research and Applications Center for Image and Signal Processing, CNRS research unit (UMR 5515). email: {vallette,prost}@creatis.insa-lyon.fr

## II. RELATED WORK

The first algorithm for progressive representation on meshes was introduced by Hoppe [6], [7]. This progressive representation is based on successive mesh simplification by edge contractions, which remove one vertex at a time. The inverse, that is the reconstruction, is achieved by vertex splits. Edge contractions are chosen so that the approximations stay close to the original mesh, using a given geometric criterion. Inspired by this idea, several approaches were proposed for progressive transmission. Pajarola and Rossignac implemented a compressed version of progressive meshes [16], encoding the mesh connectivity with an average of 7.2 bits per vertex. Taubin et al. combined simultaneous vertex splits to create the so-called progressive forest split algorithm [19], reducing the connectivity cost to 7 bits per vertex. Karni et al. improved the edge contraction sequence and the geometry coding to enhance both progressive transmission rate-distortion tradeoff and rendering speed of the processed meshes [10]. Cohen-Or et al. [5] propose a progressive transmission based on successive vertex removal followed by deterministic retriangulation. Vertices are removed according to their valence and their geometric properties. The deterministic retriangulation leads to an average connectivity compression of 6 bits per vertex.

Inspired by various single rate mesh coders [2], [20], Alliez and Desbrun introduced a progressive mesh encoding technique [1], where the connectivity of the mesh is reconstructed by transmitting only the valence of the vertices, plus some supplementary codes called null-patch. A two-stage simplification scheme keeps the mesh connectivity as regular as possible, leading to an inverse  $\sqrt{3}$  subdivision [14] for regular meshes. The vertices can also be removed according to a geometric criterion to improve the quality of the approximations. This approach compresses the mesh connectivity to an average of 3.69 bits per vertex. Karni and Gotsman [11], [12] proposed spectral geometry compression, where the geometry is projected on an orthogonal vector space, constructed with the eigenvectors of the mesh connectivity laplacian matrix. This scheme provides good mesh approximations, even with few transmitted coefficients. However, this algorithm is not fully progressive, as the mesh connectivity remains the same, only the geometry of the mesh changes with the resolution. In [13], Khodakovsky et al. present a pure geometry coder, where the input model is remeshed and provides the best rate-distortion tradeoffs so far, when the user does not need to keep the original connectivity of the 3D mesh.













Original face 		
Subdivided face (unchanged)	Subdivided face (1 to 2)	Subdivided face (1 to 3)
 1 (unchanged)	 2	  5 6
Subdivided face (1 to 4)	 3	  7 8
 11 (Lounsbery)	 4	  9 10

Fig. 1. Possible cases of subdivision

### III. A PROPOSAL FOR A WAVELET-BASED PROGRESSIVE TRANSMISSION SCHEME : WAVEMESH

Our progressive coder is based on the wavelet scheme construction for triangular meshes [21], [22], which is an extension of the regular case [15]. First, a mesh  $M^j$  is simplified according to an inverse irregular subdivision scheme where each face can be subdivided into four, three or two faces, or remain unchanged. Figure 1 shows the possible cases of subdivision. The algorithm performing such simplification is described in detail in [22]. Unfortunately, some connectivity configurations do not allow the algorithm to merge the faces according to the subdivision cases, due to its one-pass behavior. In these cases, an edge flip between two neighbor triangles is performed so that the faces can be merged (an example of such edge flip is shown in figure 2). This algorithm found no mesh it could not simplify efficiently. An example of such inverse irregular subdivision is shown in figure 3. Note that we implemented two ways of simplifying meshes: one approach is geometrically blind and simplifies meshes so that the mesh connectivity is best encoded without considering the mesh geometry. The second approach uses a Wavelet Geometrical Criterion (WGC), which tends to improve the rate-distortion efficiency of the algorithm for progressive compression. The WGC takes into account both the sharpness and wavelet coefficient magnitude of the candidate vertices : if a vertex to be removed is sharp, then its corresponding wavelet coefficients must satisfy a geometrical constraint given in detail in [22]. If the constraint is not fulfilled, then the vertex cannot be removed in that way. Such geometry-based simplification decreases the connectivity compression efficiency, but improves the rate-distortion efficiency of the algorithm.

After the simplification is complete, one can build a hierarchical relationship between the original mesh  $M^j$  (figure 3.a) and the simplified one. Therefore, the geometry of  $M^j$  (figure 3.b) can be computed by approximation of  $M^j$  by applying



Fig. 2. An edge flip for two adjacent faces

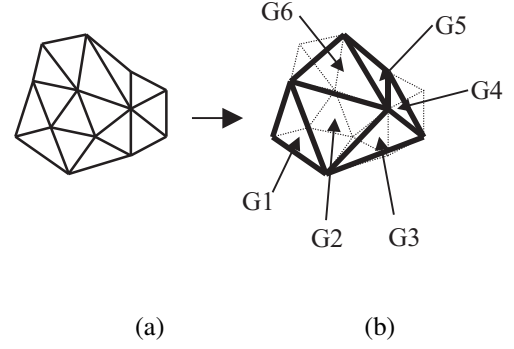


Fig. 3. Mesh simplification by inverse irregular subdivision : (a) the original mesh and (b) its simplified version.

the wavelet decomposition, with two analysis filters  $A^j$  and  $B^j$ . Let us call  $C^j$  the  $v^j \times 3$  matrix giving the coordinates of each vertex of the mesh  $M^j$  having  $v^j$  vertices. Then we have :

$$C^{j-1} = A^j \cdot C^j \quad (1)$$

$$D^{j-1} = B^j \cdot C^j \quad (2)$$

During this decomposition, two materials are computed:  $C^{j-1}$ , i.e. the geometrical approximation of the mesh, and the detail coefficients  $D^{j-1}$ , the so-called wavelet coefficients. Starting from the approximation mesh  $M^{j-1}$ , we can reconstruct exactly the original mesh if we have all of the following datasets:

- the type of subdivision for each face of the mesh
- the position of each performed edge flip
- the wavelet coefficients  $D^{j-1}$

The first two datasets enable us to reconstruct the mesh connectivity and to build the synthesis filters  $P^j$  and  $Q^j$ . With these filters, we can reconstruct the high resolution mesh geometry  $C^j$  given the low resolution mesh geometry  $C^{j-1}$  and the wavelet coefficients  $D^{j-1}$ :

$$C^j = P^j \cdot C^{j-1} + Q^j \cdot D^{j-1} \quad (3)$$

As a consequence, this scheme leads to an efficient progressive compression scheme, if the progressive data can be encoded in a smart manner. Next we describe how we encode each of the three datasets.

### IV. CONNECTIVITY COMPRESSION

For a good survey of algorithms dealing with compression of mesh connectivity, the reader can refer to [18]. Considering a mesh  $M^j$  with  $v^j$  vertices,  $n^j$  triangular faces and  $e^j$  edges,

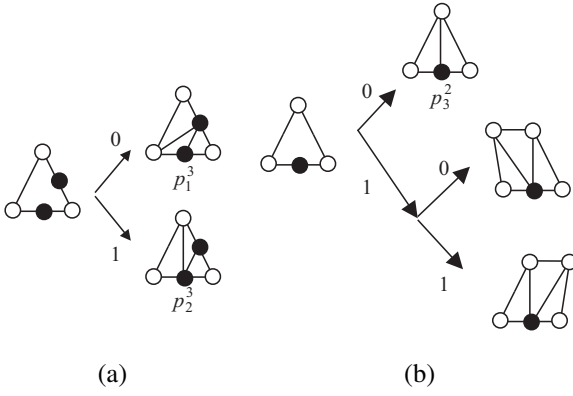


Fig. 4. some overhead data cases : when a face is subdivided into three faces (a), one bit has to be transmitted to subdivide the face properly. When a face is subdivided into two faces (b), one bit is transmitted in order to know if an edge flip is performed, and if such an edge flip has to be performed, a second supplementary bit is transmitted in order to know where the flip is performed

the bit amount of the uncompressed data representing the mesh connectivity is equal (in bits per face) to:

$$B = 3\log_2(v^j) \quad (4)$$

The proposed algorithm provides a good means of compressing this data, as the only information needed to reconstruct the highest resolution level connectivity is the connectivity of the base mesh (the lowest resolution level) and the different subdivision steps. Starting from the lowest resolution level, there is no need to store or transmit the face descriptions to reconstruct higher levels, only the subdivisions are necessary, which means the amount of information needed to reconstruct the connectivity of the mesh is an average of 1 to 3 bits per face.

For a given mesh  $M^j$ , coding its subdivision step in order to create  $M^{j+1}$  requires 3 types of information:

- D1: The data relating the subdivision step itself i.e. 1 bit for each edge of the mesh, to know if a new vertex is created on the considered edge.
- D2: Some overhead data, when one face is subdivided into three (as shown in figure 4.a), one supplementary bit per face subdivided into 3 is necessary.
- D3: Some overhead data showing when edge flips were performed during the mesh simplification process. We manage to perform edge flips only in two configurations: between a face subdivided into 2 and an unchanged face, or between two faces subdivided into 2. As a consequence, the overhead data consists at most of 1 or 2 supplementary bits per face subdivided into 2. If no flip has to be performed, a 0 is transmitted. Otherwise, a 1 is transmitted and a supplementary bit is needed in order to know which edge to flip (figure 4.b).

Then the global amount of bits  $B_s^j$  describing the subdivision step can be majored as follows:

$$B_s^j < e^j + 2n^j \quad (5)$$

Using Euler's equation, for a mesh with few boundaries and low genus, one can write the following assumption: the

number of vertices  $v^j$ , the number of edges  $e^j$ , and the number of faces  $n^j$  of a given mesh  $M^j$  have the following property [17]:

$$n^j \approx \frac{2}{3}e^j \approx 2v^j \quad (6)$$

Equation (5) now becomes :

$$B_s^j < \frac{7n^j}{2} \quad (7)$$

This amount of information creates (during the subdivision process)  $n^{j+1} - n^j$  faces and  $v^{j+1} - v^j$  vertices. So the average number of bits  $\Delta F^j$  needed per created face is equal to:

$$\Delta F^j = \frac{B_s^j}{n^{j+1} - n^j} = \frac{B_s^j}{(r^j - 1)n^j} \quad (8)$$

where the merging ratio  $r^j$  is defined as:

$$r^j = \frac{n^{j+1}}{n^j} \quad (9)$$

Combining equations (7) and (8) gives:

$$\Delta F^j < \frac{7}{2(r^j - 1)} \quad (10)$$

We can see that the compression efficiency of the proposed algorithm highly depends on the merging ratio  $r^{j-1}$ .

Taking the example of the mesh  $M^j$  in figure 5.a, our algorithm selects which vertices to remove (figure 5.b) to build a coarser mesh  $M^{j-1}$  (figure 5.c). The total amount of bits required to reconstruct the higher resolution mesh  $M^j$  connectivity consists of:

- 18 bits for the creation of children vertices (1 bit for each edge). After this step the construction of the 5 gray-colored faces in figure 5.d is straightforward.
- 2 bits for the faces subdivided into three (figure 5.e).
- 3 bits for the faces subdivided into two (only one of the 3 bits will be equal to 1, as only one edge swap has to be performed, figure 5.f)
- 1 bit for the edge swap (figure 5.g).

Finally, 24 bits are required to reconstruct the connectivity of  $M^j$  (figure 5.g), i.e. to add 10 faces to  $M^{j-1}$ . So these 10 supplementary faces were coded with an average of  $\frac{24}{10} = 2.4$  bits per face.

This compact connectivity compression scheme can be improved by using entropy coding : equation (10) shows that the higher  $r^j$  is, the higher the compression ratio will be. As a consequence, the compression step consists in merging the mesh faces four by four as often as possible to achieve a high compression factor. Hence, when a high compression factor is achieved, the bits of the dataset D1 will mostly be equal to 1. Then we reduce the size of D1, which is the biggest of the three datasets in our experiments, using binary arithmetic coding.

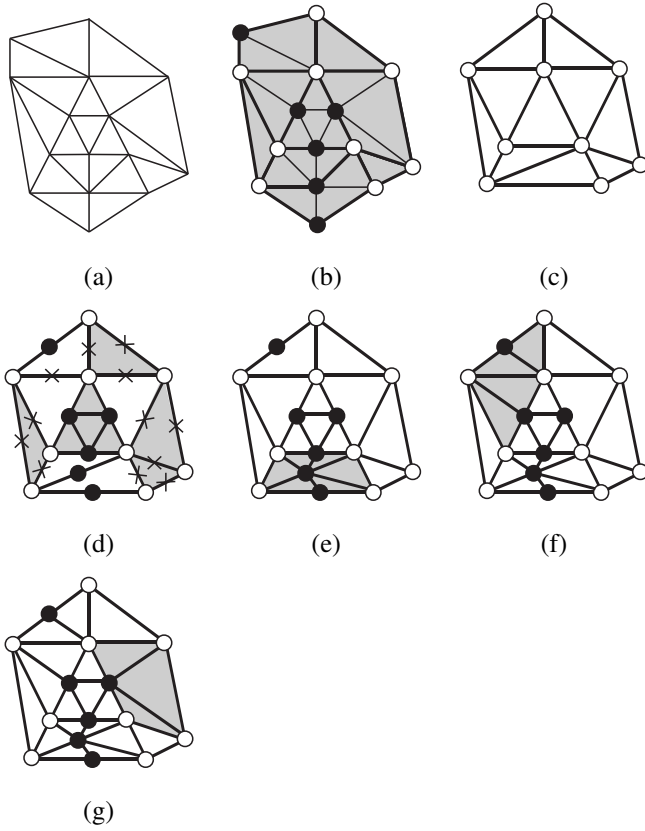


Fig. 5. Mesh reconstruction. With a few bits, our algorithm is able to reconstruct the original mesh (a) starting from the simplified mesh (c) : first, new vertices are created, and the unchanged faces and faces subdivided in four are directly created (d), then new bits arrive to reconstruct faces subdivided into three (e) and into two (f). Last comes the data for edge flips (g) and the original mesh connectivity is reconstructed

## V. GEOMETRY COMPRESSION

This sections show how we code the geometric properties of the meshes. We consider vertices with integer coordinates in order to perform lossless compression. A possible post-processing of this geometry is a scaling, to recover the original range of the vertices coordinates. In addition, a number of rounding operations have to be introduced in the multiresolution analysis-synthesis scheme. For a brief demonstration, we shall come back to the construction of the matrix filters  $A^j$ ,  $B^j$ ,  $P^j$  and  $Q^j$  used in equations (1), (2) and (3).

### A. Exact integer analysis-synthesis scheme via the lazy filter-bank

First, we introduce  $A_{lazy}^j$ ,  $B_{lazy}^j$ ,  $P_{lazy}^j$  and  $Q_{lazy}^j$  as the "lazy" filterbank. These filters do not perform any approximation, since during the analysis (where some vertices are removed, as the mesh is simplified) the coordinates of the remaining vertices stay unchanged. Due to the simple structure of the analysis matrices an exact integer analysis-synthesis scheme can be constructed using the Rounding Transform [8], [9]. Consider that the entries of the coordinates matrix  $C^j$  are integer. Perfect reconstruction is possible, as  $A_{lazy}^j$  contains only numbers 0 and 1 and  $B_{lazy}^j$  contains only numbers 0,

1 and  $\frac{1}{2}$ . The forward and inverse Rounding Transforms are defined by (11) and (12), respectively:

$$\begin{bmatrix} C_{lazy}^{j-1} \\ D_{lazy}^{j-1} \end{bmatrix} = \begin{bmatrix} A_{lazy}^j \\ B_{lazy}^j \end{bmatrix} \cdot C^j \quad (11)$$

$$C^j = \begin{bmatrix} P_{lazy}^j | Q_{lazy}^j \end{bmatrix} \begin{bmatrix} C_{lazy}^{j-1} \\ D_{lazy}^{j-1} \end{bmatrix} \quad (12)$$

where  $\begin{bmatrix} \cdot \\ \cdot \end{bmatrix}$  and  $\begin{bmatrix} \cdot & \cdot \end{bmatrix}$  are block matrices and where  $\lfloor \cdot \rfloor$  and  $\lceil \cdot \rceil$  are the floor and ceiling operators, respectively. Note that the floor operator can be used in the inverse transform if the ceiling operator is used in the forward transform. Due to the integer entries of  $A_{lazy}^j$  it results:

$$C_{lazy}^{j-1} = A_{lazy}^j \cdot C^j \quad (13)$$

In contrast as the entries of  $B_{lazy}^j$  are not integers we have:

$$D_{lazy}^{j-1} = \lfloor B_{lazy}^j \cdot C^j \rfloor \quad (14)$$

According to the inverse Rounding Transform (12) reconstruction is calculated as follows :

$$C^j = \begin{bmatrix} P_{lazy}^j \cdot C_{lazy}^{j-1} + Q_{lazy}^j \cdot D_{lazy}^{j-1} \end{bmatrix} \quad (15)$$

This scheme is illustrated by figure 6.a.

### B. Exact integer analysis-synthesis with the lifting scheme

An effective filter-bank for approximation computing directly derives from the "lazy" filter-bank modified by the lifting scheme [18]:

$$A^j = A_{lazy}^j + \alpha^j \cdot B_{lazy}^j \quad (16)$$

$$B^j = B_{lazy}^j \quad (17)$$

$$P^j = P_{lazy}^j \quad (18)$$

$$Q^j = Q_{lazy}^j - P_{lazy}^j \cdot \alpha^j \quad (19)$$

where  $\alpha^j$  is a  $v^{j-1} \times (v^j - v^{j-1})$  matrix chosen to ensure that  $C^{j-1}$  is the best approximation of  $C^j$  i.e. is built in order to make the wavelet functions more orthogonal to the scaling functions. By replacing  $A^j$  by its definition (16) into (11) it follows:

$$C^{j-1} = \lfloor A_{lazy}^j \cdot C^j + \alpha^j \cdot B_{lazy}^j \cdot C^j \rfloor \quad (20)$$

As all the entries of  $A^j \cdot C^j$  are integers it results :

$$C^{j-1} = A_{lazy}^j \cdot C^j + \lfloor \alpha^j \cdot B_{lazy}^j \cdot C^j \rfloor \quad (21)$$

According to equations (17) and (11):

$$D^{j-1} = \lfloor B_{lazy}^j \cdot C^j \rfloor \quad (22)$$

Following [3] we modify the simultaneous processing defined by (21) and (22) to a sequential processing :

$$D^{j-1} = \lfloor B_{lazy}^j \cdot C^j \rfloor \quad (23)$$

$$C^{j-1} = A_{lazy}^j \cdot C^j + \lfloor \alpha^j \cdot D^{j-1} \rfloor \quad (24)$$

Then the corresponding inverse is

$$C^j = \left\lceil P_{lazy}^j \cdot (C^{j-1} - \lfloor \alpha^j \cdot D^{j-1} \rfloor) + Q_{lazy}^j \cdot D^{j-1} \right\rceil \quad (25)$$

Equations (23), (24 and (25) now give the integer-to-integer version of the multiresolution wavelet scheme defined in equations (1), (2) and (3), as illustrated in figure 6.b. This scheme can then be used for progressive compression. The wavelet decomposition transforms coordinates into wavelet coefficients with a histogram concentrated around the zero value, making them well suited for entropy coding. Then we use adaptive arithmetic coding to compress the size of the wavelet coefficients  $D^{j-1}$ , which are the only data to transmit for the mesh geometry construction, as the synthesis filters can be constructed with the mesh subdivision data already transmitted for the connectivity reconstruction. Finally, for short, we call our algorithm Wavemesh.

## VI. RESULTS

### A. Progressive Compression

During our experiments, we used our Wavemesh algorithm in three different configurations:

- I : Without any lifting : the "lazy" approach (see [22] for details)
- II : With the lifting scheme (using a 0-disc wavelet support)
- III : with the lifting scheme and WGC activated to optimize the algorithm in a rate-distortion sense [22].

Figure 7 shows the rate-distortion curve for the Fandisk mesh of 6475 vertices quantified to 10 bit per coordinates, for the three different approaches. The vertical axis is the mean square error returned by the Metro tool [4], in terms of percentage of the mesh bounding box. The lifting scheme considerably improves the approximation quality in comparison with the "lazy" approach. The WGC included in the third approach increases again the efficiency of the third, which provides the best results in this rate-distortion competition.

Next follow the progressive compression results obtained by our algorithm compared to other algorithms : Alliez and Desbrun's valence-based progressive mesh [1], Karni et al.'s progressive meshes [10] and Spectral compression [11] (results taken from [10]).

Figure 8 shows the results on the Venus head mesh of 8268 vertices (quantization : 12 bits per coordinate), where our algorithm performs better than Karni et al.'s, even for its "lazy" version. The lifted versions are also close to those obtained with the Spectral Compression, which is not a really lossless algorithm. Note that for this mesh, the addition of WGC didn't improve the results for this mesh.

Figure 9 shows the results for the Venus body mesh of 11362 vertices (10 bits/coordinate quantization). Again, our algorithm performed better than Karni & Gostman's approach. Our method obtains results similar to Alliez & Desbrun's for high bitrates, but performs significantly better for bitrates below 40,000 bits. For this mesh, the best progressive compression is obtained by Spectral Compression. Again, with this mesh, the WGC did not significantly improve the results.

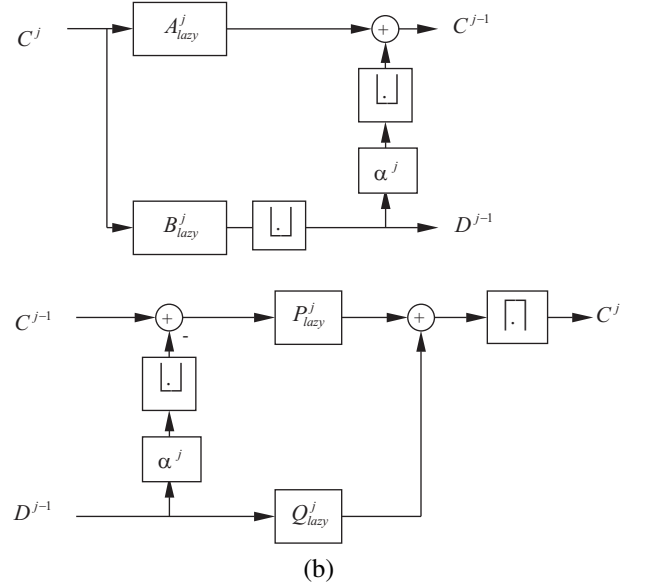
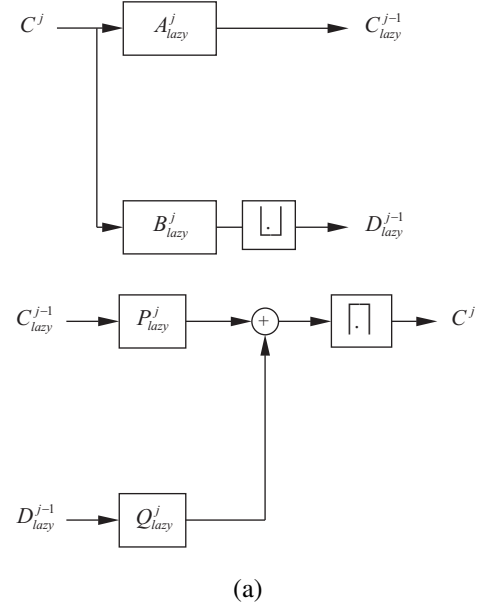


Fig. 6. one level integer-to-integer wavelet analysis and synthesis schemes: (a) with lazy wavelets, (b) with lifted lazy wavelets

### B. Lossless Compression

Table I shows lossless compression results obtained with our algorithm on various reference meshes, for both 10 bits and 12 bits coordinate quantization. Some results obtained by Alliez & Desbrun are also given. The processing time is also given for each mesh. Our current implementation encodes meshes with the average speed of 2000 vertices/s. For the Fandisk mesh and the Venus head, we provide 2 different results : the first result (a) was obtained by compressing the mesh without WGC and the second result (b) was obtained with WGC, for an improved progressive transmission.

In terms of connectivity compression, except for the highly regular torus, we obtain a gain of 15% to 36% compared to Alliez & Desbrun which had obtained previously the best



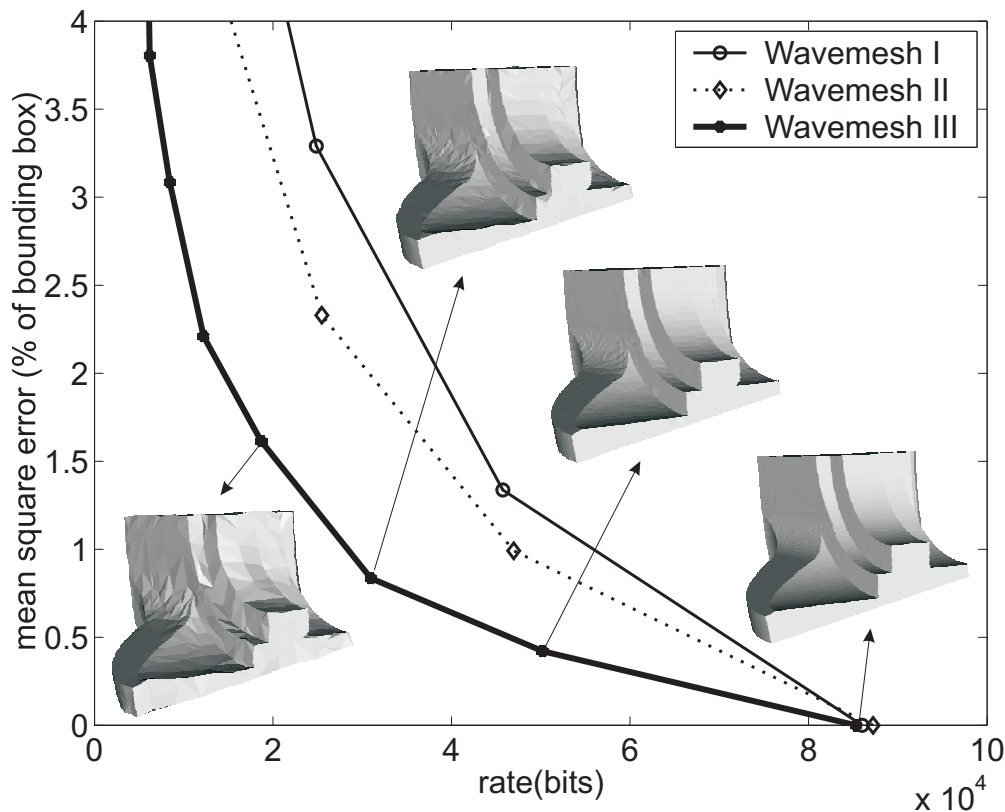


Fig. 7. Rate-distortion curve for the Fandisk

results so far [1]. Our geometry compression scheme also leads to smaller bitrates (up to 20% better). Also, if we compare the results obtained with and without WGC, we can see that WGC improves the general compression rate for the fandisk mesh by about 0.5 bit/vertex, with a less effective connectivity compression (about 0.5 more bits/vertex) but with a much more effective geometry compression (about 1 bit/vertex below). Then, for this mesh, using WGC, we improve both progressive transmission and lossless compression. We do not obtain such conclusion when considering the Venus head mesh, where WGC improved neither the progressive transmission quality nor the lossless compression rate. WGC may be only well adapted to meshes with relatively flat surfaces with a low number of sharp edges, and may not perform well on smooth meshes, which is an issue to be further investigated.

## VII. CONCLUSION

We propose a new wavelet-based lossy to lossless mesh compression algorithm : Wavemesh, which outperforms previously reported works on both progressive and lossless compression. This new connectivity and geometry coding scheme comes in addition to a wavelet scheme for irregular meshes [22], which comes as an extension of the original wavelet scheme for subdivision connectivity meshes [15]. Our connectivity coding strategy is based on a compact representation for an irregular subdivision scheme which allows the reconstruction of any irregular mesh, improved by arithmetic

coding. The geometry coding scheme is based on an entropy coding of wavelet coefficients. The current implementation clearly demonstrates the superiority of the approach over previous ones. Further work may introduce performance-speed tradeoff, connectivity coding optimizations in terms of a better inverse problem solver, color and texture coding, and near lossless coding of the mesh geometry for better progressive transmission performance.

## ACKNOWLEDGMENTS

This work is in the scope of the scientific topics of the GdR-PRC ISIS research group of the French National Center for Scientific research.

The authors would like to thank the reviewers both for their helpful comments and suggestions. Their criticisms helped the authors to improve the quality of this paper.

## REFERENCES

- [1] P. Alliez and M. Desbrun, "Progressive encoding for lossless transmission of 3d meshes," in *ACM Siggraph Conference Proceedings*, 2001, pp. 198–205.
- [2] P. Alliez and M. Desbrun, "Valence-driven connectivity encoding of 3d meshes," in *Eurographics Conference Proceedings*, 2001, pp. 480–489.
- [3] R.C. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo, "Wavelet transforms that map integers to integers," *Applied and Computational Harmonic Analysis*, vol. 5(3), pp. 332–369, 1998.
- [4] P. Cignoni, C. Rochini, and R. Scopigno, "Metro : Measuring error on simplified surfaces," *Computer Graphics Forum*, no. 17(2), pp. 167–174, 1998.

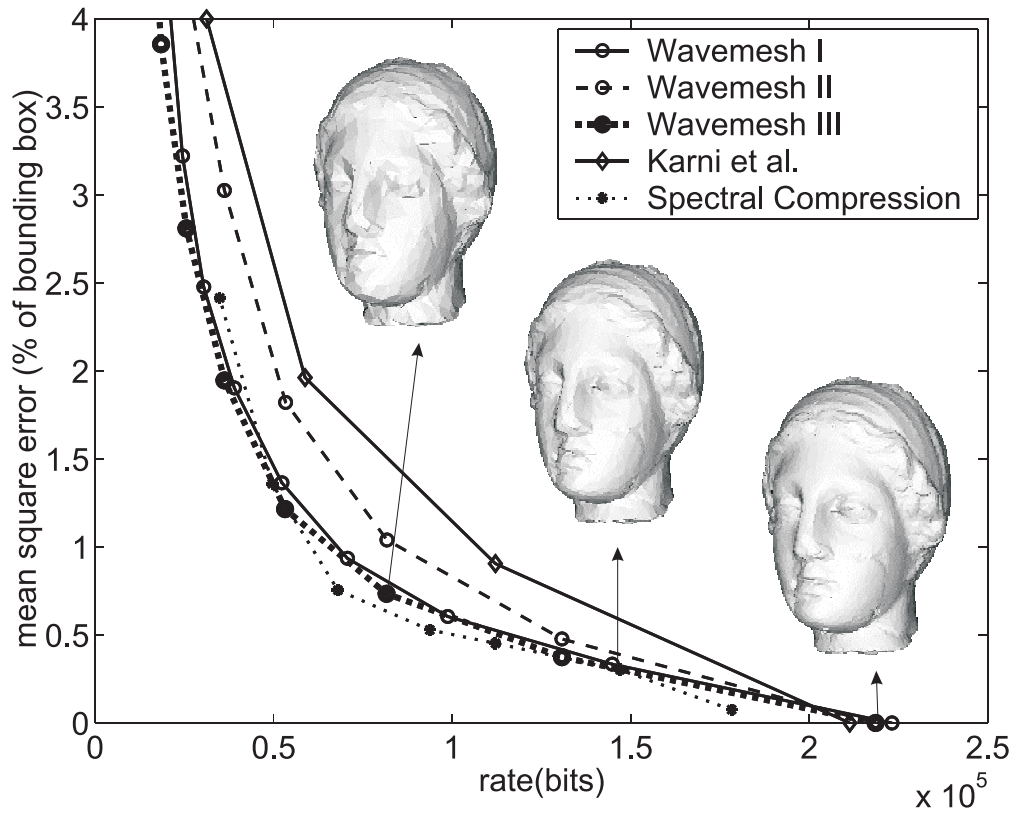


Fig. 8. Rate-distortion curve for the Venus head

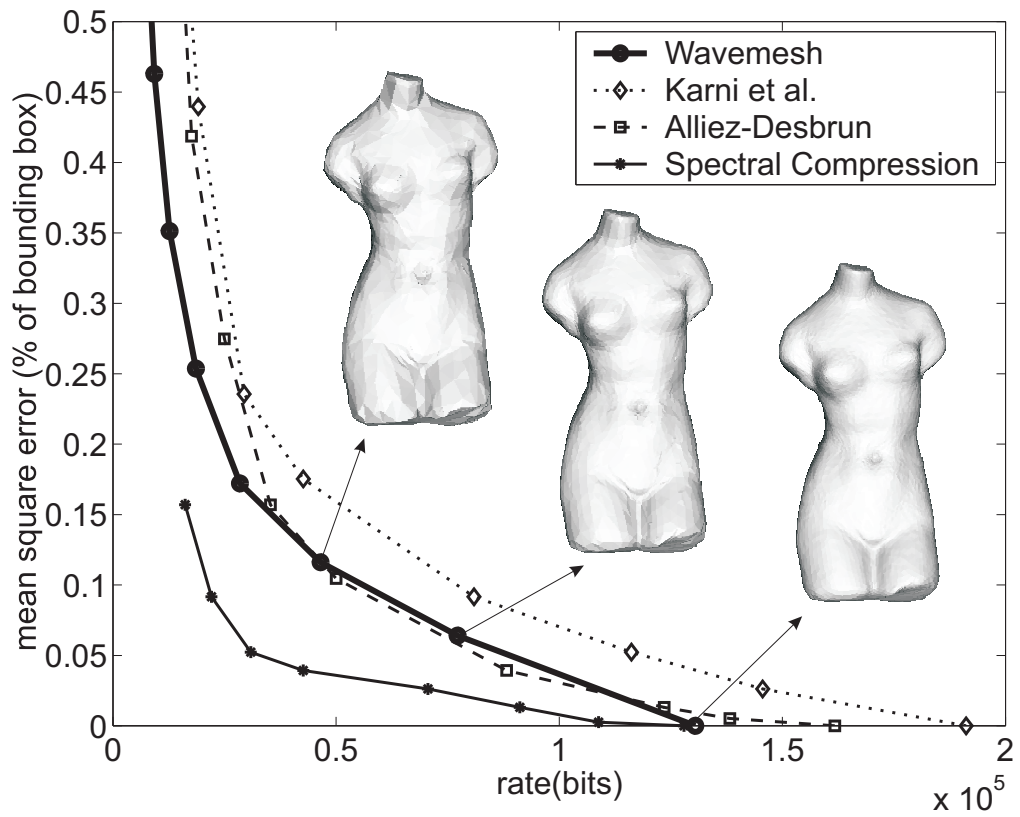


Fig. 9. Rate-distortion curve for the Venusbody



TABLE I  
LOSSLESS COMPRESSION RESULTS FOR VARIOUS REFERENCE MESHES AND SEVERAL APPROACHES (RESULTS IN BITS/VERTEX)

Model	Vertices	Connectivity(bits/v)		Geometry (10 bits)			Geometry (12 bits)		Time (s)
		AD01	Ours	AD01	Ours	Total	Ours	Total	
Blob	8036→4	?	3.39	?	14.35	17.74	17.39	20.78	3.56
Bunny	34834→22	?	2.76	?	8.5	11.26	13.48	16.24	14.5
Eight	766→15	?	2.62	?	15.36	17.98	21.34	23.96	0.47
Fandisk (a)	6475→4	?	2.59	?	10.89	13.48	15.9	18.49	2.39
Fandisk (b)	6475→4	4.99	3.16	12.34	9.94	13.1	14.76	17.92	2.98
Horse	19851→4	4.61	3.91	?	10.87	14.78	16.76	20.67	10.63
Mannequin	11703→4	3.58	2.70	9.98	9.62	12.32	15.48	18.18	4.69
Tiger	2738→4	2.67	1.72	12.67	12.34	14.06	18.58	20.3	0.92
Torus	36450→8	0.39	0.48	3.58	3.58	4.06	5.79	6.27	8.9
Venus Head (a)	8268→4	?	4.97	?	15.17	20.14	21.53	26.5	6.15
Venus Head (b)	8268→4	?	5.46	?	15.18	20.64	21.54	27	8.04
Venus Body	11362→4	3.59	2.77	10.15	8.63	11.4	14.3	17.07	4.71

- [5] D. Cohen-Or, D. Levin, and O. Remez, "Progressive compression of arbitrary triangular meshes," in *IEEE Visualization 99*, 1999, pp. 67–72.
- [6] H. Hoppe, "Progressive meshes," in *ACM Siggraph 96 Conference Proceedings*, 1996, pp. 99–108.
- [7] H. Hoppe, "Efficient implementation of progressive meshes," *Computer & Graphics*, vol. 22, 1998.
- [8] H.Y. Jung, T.Y. Choi, and R. Prost, "Rounding transform and its application for pyramid structured coding," *IEEE Transactions on Image Processing*, vol. 7(2), pp. 234–237, 1998.
- [9] H.Y. Jung and R. Prost, "Lossless subband coding system based on rounding transform," *IEEE Transactions on Signal Processing*, vol. 46(9), pp. 2535–2540, 1998.
- [10] Z. Karni, A. Bogomjakov, and C. Gotsman, "Efficient compression and rendering of multi-resolution meshes," in *Proceedings of IEEE Visualization*, 2002.
- [11] Z. Karni and C. Gotsman, "Spectral Compression of Mesh Geometry," in *ACM Siggraph 00 Conference Proceedings*, 2000, pp. 279–286.
- [12] Z. Karni and C. Gotsman, "3d mesh compression using fixed spectral bases," *Proceedings of Graphics Interface*, pp. 1–8, 2001.
- [13] A. Khodakovsky, P. Schröder, and W. Sweldens, "Progressive Geometry Compression," *ACM Siggraph Conference Proceedings*, pp. 271–278, 2000.
- [14] L. Kobbelt, " $\sqrt{3}$  subdivision," in *ACM Siggraph Conference Proceedings*, 2000, pp. 103–112.
- [15] M. Lounsbery, *Multiresolution Analysis for Surfaces of Arbitrary Topological Type*, PhD thesis. Dept. of Computer Science, University of Washington, 1994.
- [16] R. Pajarola and J. Rossignac, "Compressed Progressive Meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 6(1), pp. 79–93, 2000.
- [17] J. Rossignac, "EdgeBreaker : Connectivity Compression for Triangle Meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 5(1), pp. 47–61, 1999.
- [18] W. Sweldens, "The lifting scheme : A custom-design construction of biorthogonal wavelets," *Applied and Computational Harmonic Analysis*, vol. 3, pp. 186–200, April 1996.
- [19] G. Taubin, W. Horn, J. Rossignac, and F. Lazarus, "Geometry Coding and VRML," in *Proceedings of the IEEE, Special issue on Multimedia Signal Processing*, June 1998, vol. 86(6), pp. 1228–1243.
- [20] C. Touma and C. Gotsman, "Triangle Mesh Compression," *Graphics Interface 98 Conference Proceedings*, pp. 26–34, 1998.
- [21] S. Valette, Y. S. Kim, H. J. Jung, I. Magnin, and R. Prost, "A multiresolution wavelet scheme for irregularly subdivided 3d triangular mesh," in *IEEE Int. Conf on Image Processing ICIP99*, October 1999, vol. 1, pp. 171–174.
- [22] S. Valette and R. Prost, "Wavelet based multiresolution analysis of irregular surface meshes," *IEEE Transactions on Visualization and Computer Graphics*, to appear, available upon request, 2003.



**Sébastien Valette** Was born in France, in 1975. He received the M.S. Degree from the Electrical Engineering Department, at the National Institute for Applied Sciences (INSA) of Lyon, France, in 1998. He obtained the PhD Degree at INSA of Lyon in 2002. His research interests include 3D processing, wavelets, progressive compression and multiresolution analysis.



**Rémy Prost** received his doctorate degree in Electronics Engineering and his "Docteur es Sciences" degree from Lyon University and the National Institute of Applied Sciences (INSA), Lyon, France, in 1977 and 1987 respectively. He is currently a professor in the Department of Electrical Engineering at INSA -Lyon. Both his teaching and research interests include digital signal processing, inverse problems, image data compression, multiresolution algorithms, wavelets, and meshes processing. He leads the 'Volume (3D) Image Processing' project in the CREATIS Laboratory (CNRS #5515) at INSA-Lyon. Since 1982 he is a member of the IEEE.