

Temporal Wavelet-based Compression for 3D Animated Models

Frédéric Payan¹, Marc Antonini

Laboratoire I3S - UMR 6070 CNRS / Université de Nice - Sophia Antipolis

2000, route des Lucioles - 06903 Sophia Antipolis

Phone: +33 (0)4 92 94 27 22, Fax: +33 (0)4 92 94 28 98

Email: {fpayan,am}@i3s.unice.fr

Abstract

We present an efficient compression scheme for animated sequences of triangular meshes of the same connectivity. The proposed algorithm exploits the temporal coherence of the geometry component by using a temporal wavelet filtering. The quantization of the resulting wavelet coefficients is then optimized by a bit allocation process. This process dispatches the bit budget across the coefficient subbands according to their influence on the quality of the reconstructed sequence for one specific user-given bitrate. The proposed scheme is simple, fast, flexible, and scalable in frame rate and bitrate. Moreover, simulation results show that our approach is competitive for any kind of animated models, whatever the characteristics (parametrically coherent or not, fine/coarse meshes...), contrary to the related works.

Key words: Compression; animation; irregular meshes; temporal wavelet; lifting scheme; bit allocation

¹ Corresponding author. Frédéric Payan is currently in PostDoctoral Research Fellow in the Geometric Modeling group (MGA) of the laboratory LMC-IMAG (CNRS) at Grenoble (France).



Fig. 1. Several frames of the model CHICKEN.

1 Introduction

Today animated 3D objects are widely used in numerous domains (scientific applications, computer games, animation...) to represent realistic visual data. These animated 3D objects are most of the time represented by a sequence of irregular meshes sharing the same connectivity at any frame. The motion is obtained by modifying the positioning of the vertices, frame after frame (see Fig. 1). For complex animated mesh sequences, this involves a very large representation, and the compression is a relevant technique to ease storage or transmission of these data.

Compared to the compression of static meshes [1], the problem of producing compact representations for animated sequences has gone largely unaddressed. The overall scheme of the compression methods in this domain [2–10] is the following: an analysis step first transforms the original geometry to reduce the signal information, and then a predictive coding scheme is applied on the resulting details to exploit the temporal coherence between successive frames. The natural approach for analyzing time-varying data is to exploit the temporal coherence, like in video processing [11–13]. Two of the most relevant approaches for animated sequences introduce for instance the principal component analysis to exploit the temporal coherence of the geometry component [4, 5]. Unfortunately this tool is efficient only for very specific sequences (with few global motion and a number of frames far higher than the number of vertices). Moreover, the time and memory complexity of this tool is high.

Although wavelets are very popular and particularly efficient in video compression for instance [11–13], they are poorly studied in animation compression. Moreover, the two only wavelet-based coders for animated mesh sequences introduce (*spatial*) wavelets for static meshes, and consequently their efficiency depends on the *parametrical coherence* of the mesh sequences [10, 14]. Since we focus on sequences of irregular meshes sharing the same connectivity, one relevant solution to overcome the problem relating to the parametrical coherence is to apply the wavelet filtering along the time axis. Moreover, the fixed connectivity ensures the regularity of the geometry sampling along the time axis, and involves that the wavelet filtering is always valid for any kind of mesh sequences, parametrically coherent or not.

Therefore we describe in this paper a *temporal wavelet-based coder* for animated sequences of meshes of the same connectivity. One key contribution is the use of a temporal wavelet transform during the analysis step to exploit the temporal coherence of such data. The second key contribution of our compression scheme is the encoding of the resulting wavelet coefficients, optimized by a temporal model-based bit allocation initially proposed for the static semiregular meshes [15]. This coding scheme has the advantage to be valid whatever the wavelet used. We will experimentally show that the proposed coding scheme is simple, fast, flexible and competitive for any kind of animated models whatever the characteristics (parametrically coherent or not, fine/coarse meshes...), contrary to the state-of-the-art methods .

The rest of the paper is organized as follows. Section 2 presents the previous works relative to the compression of animated mesh sequences. Section 3 introduces the problem statement and the proposed approach. Section 4 describes the overall scheme of our compression algorithm. Section 5 gives a detailed description of the model-based bit allocation process. Section 6 deals with the temporal wavelet filtering. Simulation results of our coder are given in section 7. Finally, we conclude and propose directions for future works in Section 8.

2 Related Works

The first compression scheme for animated sequences of meshes, proposed by Lengyel, exploits the affine transformations [2]. The author proposes to split a mesh into several submeshes, and computes a rigid-body motion for each submesh. In this way, only a set of affine transformations are needed to represent a submesh, instead of all the displacements of the submesh vertices. Shamir and Pascucci proposed another approach based on the affine transformations, but combined to a multiresolution scheme [3]. Their technique first computes the best affine transformations of the mesh at the first frame to match the subsequent ones, and then encodes the temporal prediction errors. Other schemes based on predictive coding have been proposed to exploit the temporal and spatial correlations of mesh sequences [6, 7, 9]. The overall idea of all these approaches is to predict the displacements of the vertices along the sequence and then to encode the residual errors.

In parallel, Alexa and Müller [4] proposed a coding scheme based on the principal component analysis (PCA) to represent the mesh sequences with only a small number of basis functions. Karni and Gotsman improved this method by further exploiting the temporal coherence and finally encode the PCA coefficients with a second-order linear predictive coding (LPC) [16].

In [8] Briceno *et al.* presented an original approach. The technique is to project each frame onto a 2D image, and then encode the resulting sequence of "2D images" with some well-known video techniques.

Recently, a wavelet-based compression method has been presented by Guskov and Khodakovsky in [10]. They proposed to exploit the *parametric* coherence of some specific animated sequences of meshes, by applying a spatial multiresolution analysis on the frames. Each frame is consequently transformed into several sets of details, in other words the wavelet coefficients. In order to also exploit the temporal coherence, the subbands of coefficients are then progressively encoded by a predictive coding scheme based on a I-frames/P-frames approach (similar to some video compression methods).

In parallel, J.H. Yang *et al.* also proposed a wavelet-based compression algorithm, but for sequences of irregular meshes with changing connectivity [14]. They proposed to construct a semi-regular mesh structure at the first frame and then match it to the other frames by using hierarchical motion estimation technique. They finally use a zerotree coding scheme to compress the motion compensation residuals between the successive frames.

3 Problem statement and proposed approach

Although they are widespread in video processing [11–13], wavelets are not fully exploited in animation compression. Actually, we observe that the works using wavelets in animation compression [10, 14] propose an approach based on spatial wavelets (for static meshes). Such a $3D+t$ approach is evident when the input sequence is a rigid-body motion. In that case, the wavelet coefficients are unchanged at any frame, and this strongly reduces the signal entropy of the moving shape [10]. But the interesting animated mesh sequences are not typically rigid, and consequently the coding performances strongly depend on the stability of the wavelet transform. For instance, the coder of [10] is irrelevant for the *Chicken* animation (Fig. 1), because of the lack of parametric coherence in the shape evolution [17]. As a result, the $3D+t$ approaches of [10, 14] are efficient only for *parametrically coherent mesh sequences*. Moreover, a spatial decomposition is attractive only for fine meshes, in other words sequences with a high number of vertices (highlighted by the authors of [10]).

The objective of our work is to propose a wavelet coder for any kind of animated mesh sequences (parametrically coherent or not, fine or coarse meshes). We prefer using a temporal wavelet transform instead of a spatial one, for several reasons. First, the shared connectivity ensures that the geometry sampling is regular (in time). This implies that the analysis step is always valid, whatever the characteristics (manifold or not, sharp features, coarse meshes). Moreover, using a temporal wavelet overcomes the problem of the parametric coherence. In addition, using wavelets to exploit the temporal coherence ensures a compact representation of the transformed data, whatever the motion, and the number of vertices of the given animation, contrary to the PCA-based methods [4, 5].

Then, to encode the different subbands of temporal details, we choose to expand the model-based coder of [15] to the animated mesh sequences (initially proposed for the semiregular meshes).

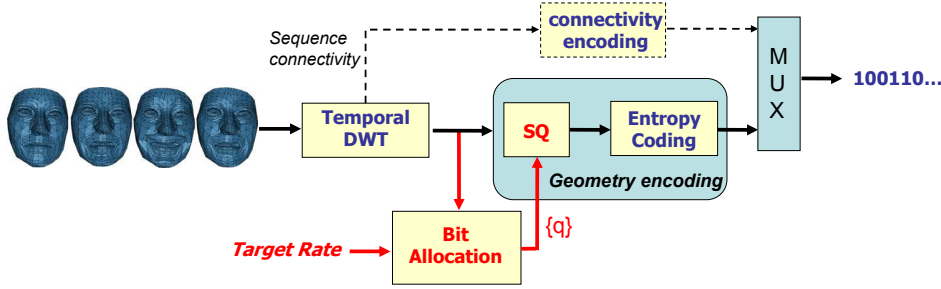


Fig. 2. Overall coding scheme.

4 Overview of our approach

Here is the outline of our coding scheme (Fig. 2).

- **Temporal wavelet transform (WT):** a temporal wavelet filtering first transforms the original geometry component to reduce the signal information. The resulting one-level wavelet decomposition consists in a subband of detail frames (temporal *high-frequency*) and a subband of approximation frames (temporal *low-frequency*), as shown in Fig. 7. Each vertex is represented by a 3D detail (or *wavelet coefficient*) at any frame.
- **Scalar quantization (SQ):** the resulting subbands of frames are then splitted in three 1D subbands according to the coordinates. These *coordinate* subbands are then encoded using uniform scalar quantizers *SQ*. The quantizers depend on the optimal quantization steps computed during the bit allocation process.
- **Bit allocation:** the allocation process optimizes the rate-distortion trade-off relative to the data quantization. The objective is to efficiently dispatch the bit budget across the subbands according to their influence on the quality of the reconstructed mesh sequence for one specific bitrate. A detailed description of this process is given in the next section.
- **Entropy coding:** the quantized details are then encoded with an entropy coder, to further compress the data and finally produce the bitstream. A simplified version of the context-based arithmetic coder of [18] is used.
- **Connectivity coding:** in parallel, the shared connectivity of the original sequence must be encoded and transmitted if we want to reconstruct the sequences after decompression. In this paper, we use the valence-based coder of Touma and Gotsman [19].

5 Bit allocation

5.1 Principle

The general purpose of a bit allocation process is to optimize the trade-off between the bitrate and the quality of the reconstructed data. In this paper, the allocation process allows to minimize the quantization losses of the geometry component for one user-given target bitrate.

Since the temporal wavelet transforms the original sequence in a multilevel structure, the objective of the allocation process is to dispatch the bit budget across the different subbands of the multilevel structure according to their influence on the quality of the reconstructed sequences.

More precisely, our allocation process computes the set of optimal quantizers $\{q^*\}$ minimizing the reconstructed mean square error D_T for one specific user-given target bitrate R_{target} . The solutions $\{q^*\}$ are obtained by solving the problem

$$(\mathcal{P}) \begin{cases} \text{minimize} & D_T(\{q\}) \\ \text{with constraint} & R_T(\{q\}) = R_{target}, \end{cases} \quad (1)$$

with R_T the total bitrate.

5.2 Optimal solution

By using a lagrangian approach, the constrained allocation problem \mathcal{P} can be solved by minimizing the criterion

$$J_\lambda(\{q\}) = D_T(\{q\}) + \lambda(R_T(\{q\}) - R_{target}), \quad (2)$$

with λ the lagrangian operator.

The optimal quantization steps $\{q^*\}$ are obtained by solving the following system:

$$\begin{cases} \frac{\partial J_\lambda(\{q\})}{\partial q} = 0 \\ \frac{\partial J_\lambda(\{q\})}{\partial \lambda} = 0 \end{cases} \quad (3)$$

As introduced in Section 4, the 3D subbands are splitted in three 1D (*coordinate*) subbands, and then encoded separately with three different scalar quantizers. So, the reconstructed mean square error can be defined by

$$D_T(\{q\}) = \sum_{i=0}^N w_i \sum_{j=1}^3 D_{i,j}(q_{i,j}), \quad (4)$$

where $\{w_i\}$ are weights due to the wavelet non-orthogonality [20] (i is the resolution level), $D_{i,j}$ the mean square error relative to the coordinate subband i, j ($j = 1$ for the x -coordinates, $j = 2$ for the y -coordinates, and $j = 3$ for the z -coordinates), and $q_{i,j}$ the associated quantization step. The weights $\{w_i\}$ are defined by $w_0 = (w_{lf})^N$ and $w_i = w_{hf}(w_{lf})^{N-i} \forall i \neq 0$, where w_{lf} and w_{hf} are two weights depending on the wavelet used [20]. Tab. 1 gives for instance the numerical values of these weights for different lifting schemes experimented during this work.

Filter	[2, 0]	[2, 2]	[4, 2]	[6, 2]
w_{lf}	0.75	0.75	0.8203125	0.85299682
w_{hf}	0.5	0.359375	0.3410644531	0.3359718323

Table 1

Weights corresponding to different lifting schemes.

In parallel, the total bitrate R_T can be developed in

$$R_T(\{q\}) = \sum_{i=0}^N \sum_{j=1}^3 a_{i,j} R_{i,j}(q_{i,j}), \quad (5)$$

with $R_{i,j}$ the bitrate relative to the coordinate subband i, j , and $\{a_{i,j}\}$ the coefficients relative to the subsampling step (ratio between the size of the coordinate subband and the total number of samples). By merging (4) and (5) in (2), the system (3) becomes a system of $(3N + 4)$ equations with $(3N + 4)$ unknowns (the set $\{q_{i,j}\}$ and λ):

$$\frac{\frac{\partial D_{i,j}(q_{i,j})}{\partial q_{i,j}}}{\frac{\partial R_{i,j}(q_{i,j})}{\partial q_{i,j}}} = -\lambda \frac{a_{i,j}}{w_i} \quad (6a)$$

$$\sum_{i=0}^N \sum_{j \in J_i} a_{i,j} R_{i,j}(q_{i,j}) = R_{target}. \quad (6b)$$

The solutions of (6a) may be obtained by inverting the equations. But this stage is a complex operation, and we overcome this problem by using an iterative algorithm depending on λ [15].

5.3 Overall Algorithm

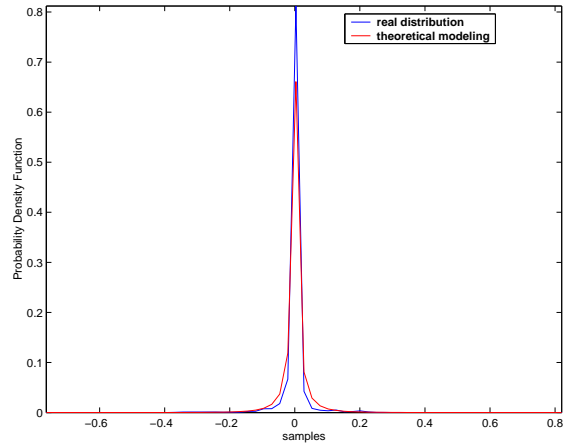
The optimal solutions of the system (6) for the given bitrate R_{target} are computed thanks to the following overall algorithm:

- (1) λ is given. For each set (i, j) , compute $q_{i,j}$ that verifies (6a);
- (2) while (6b) is not verified, calculate a new λ by dichotomy and return to step (1);
- (3) stop.

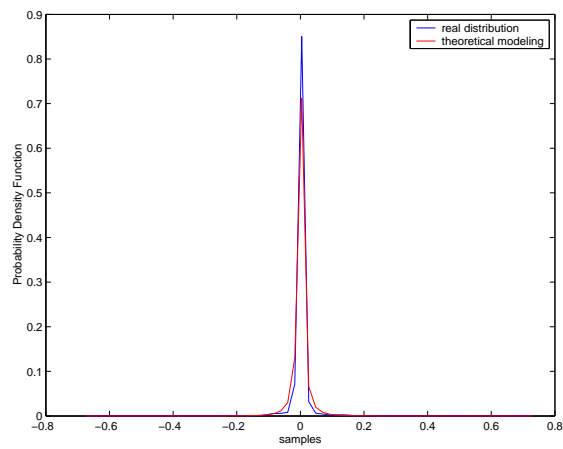
5.4 Distribution of the detail subbands

The computation of the quantization steps $\{q_{i,j}\}$ as solutions of (6a) can be done according to different methods. Using a model-based algorithm is a relevant solution, providing that the processed data have specific statistical properties. Therefore we study the distribution of the coordinate subbands at different decomposition levels.

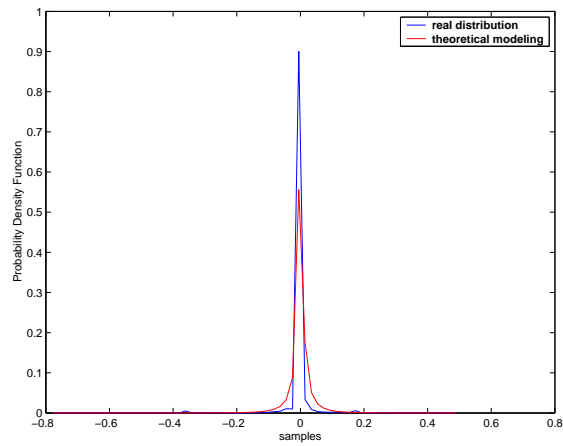
Fig. 3 shows the probability density functions (*pdf*) of three typical coordinate subbands of temporal details (CHICKEN animation). We observe that these *pdf* can be modeled by a generalized gaussian distribution (*GGD*). Hence, we can use the model-based algorithm presented in [15] to compute the solutions of (6a). This leads to a fast and low complex algorithm. The interested reader should refer to [15] for more details and explanations.



(a) X-coordinates (level 1).



(b) Y-coordinates (level 1).



(c) Z-coordinates (level 1).

Fig. 3. Typical *pdf* of coordinate subbands of detail frames (CHICKEN animation). The blue and red curves represent respectively the real distribution and the approximated one modeled by a *GGD*.

5.5 Encoding of the approximation frames

On the other hand, the typical *pdf* of coordinate subbands of approximation frames cannot be modeled by a *GGD* (see Fig. 4(a),4(c), and 4(e)). To overcome this problem, we propose to use a *differential coding*. The overall idea of such a coding is to encode differences between samples instead of samples themselves [15].

So, we deal with the geometric differences between the coordinates of the approximation frames (in function of the vertex indexing), *i.e.*, for each frame t :

$$[V(0, t), V(1, t) - V(0, t), V(2, t) - V(1, t), \dots, V(n_v, t) - V(n_v - 1, t)]$$

.

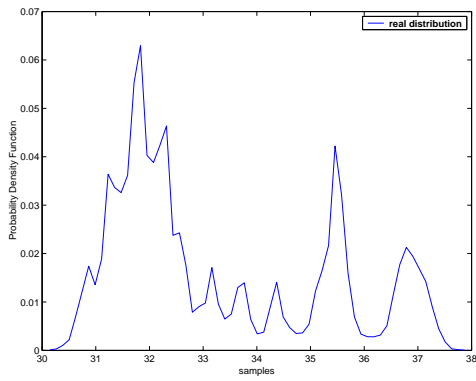
Fig. 4(b), 4(d), and 4(f) show the typical *pdf* of three coordinate subbands of approximation frames (CHICKEN animation), when using the proposed differential coding. We observe that the distribution of the coefficients of the approximation frames finally can be modeled by a *GGD* as well. Moreover, this allows to exploit the spatial correlation between neighbor vertices, and further reduce the data information needed to represent the *LF* sequence.

This differential coding may appear to be a crude solution. Nevertheless, it is sufficient for our algorithm since the model-based algorithm presented in [15] can be finally applied on the *LF* data. This is partly due to the valence-based coder of Touma and Gotsman used to encode the connectivity. It provides a non arbitrary indexing of the vertices sufficiently coherent [19]. Moreover, the low complexity of this approach leads to very fast encoding and decoding of the approximation frames. However, the improvement of this encoding is an interesting direction for future works (see Section 8).

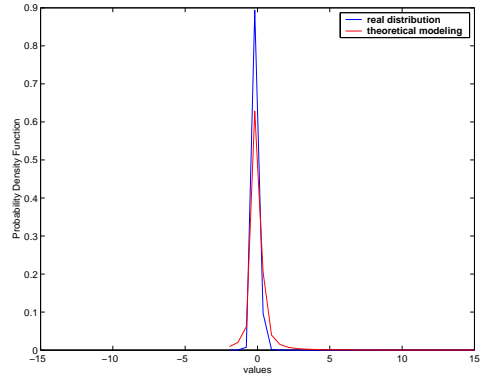
6 Temporal Wavelet Transform for Animated Mesh Sequences

6.1 Principle

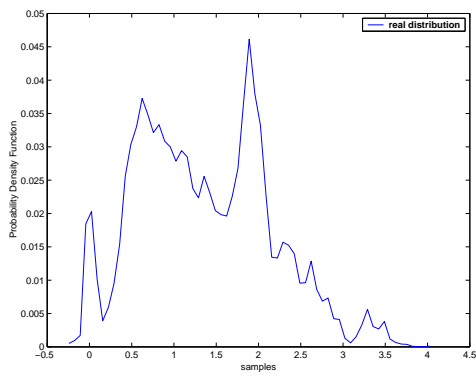
Wavelets are powerful tools widely used in signal processing to provide a multiresolution representation of a given *dD* signal and enable decorrelation in



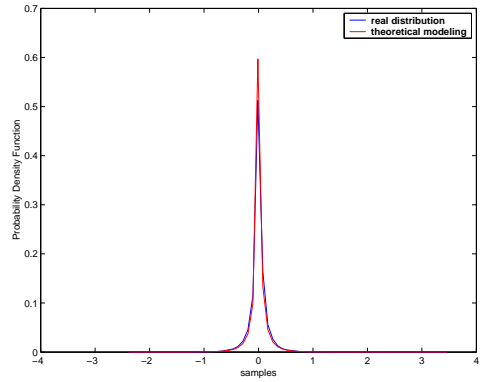
(a) X-coordinate subband without differential approach.



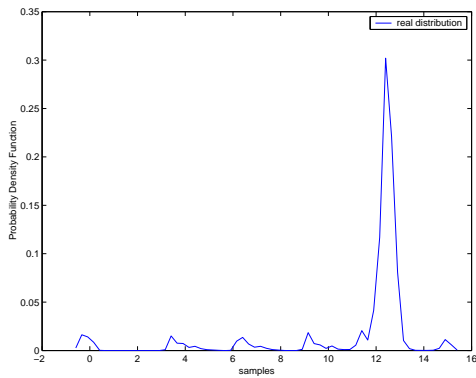
(b) X-coordinate subband with differential approach.



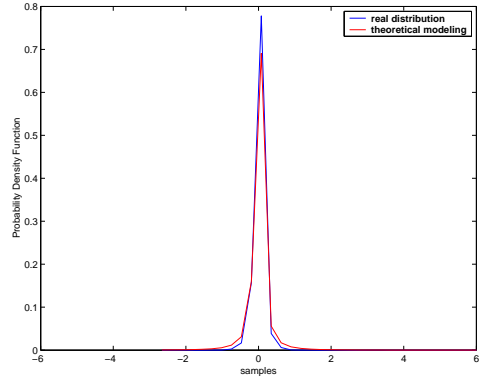
(c) Y-coordinate subband without differential approach.



(d) Y-coordinate subband with differential approach.



(e) Z-coordinate subband without differential approach.



(f) Z-coordinate subband with differential approach.

Fig. 4. Typical probability density function of coordinate subbands of approximation frames, with and without differential coding (CHICKEN animation). The blue and red curves represents respectively the real distribution and the approximated one modeled by a GGD .

space and frequency [21]. In case of time-varying signals ($dD + t$), a wavelet filtering can be also applied along the temporal axis to exploit the temporal coherence (see Fig. 5). This discrete *temporal wavelet transform* (TWT) leads to an efficient energy concentration on the low-pass temporal subbands.

As discussed in Sections 1 and 3, a *TWT* is a natural and relevant approach for the geometry component of animated mesh sequences sharing the same connectivity. The fixed connectivity implies a regular sampling in time and a fixed number of vertices at any frame. Thus, a discrete TWT can be easily applied on this kind of data.

An animated sequence of meshes sharing the same connectivity is defined by a set of T meshes $\{f_1, f_2, \dots, f_T\}$, where f_i represents the i th frame. f_i is defined by its geometry (*i.e.* the positioning of the vertices at the frame i), and a list of triangles describing the connectivity of the vertices at any frame. As the number of vertices is fixed at any frame, we propose in this paper to apply a monodimensional *TWT* on the successive locations of each vertex (see Fig.6) [22, 23]. The principle is the following. Let us define the evolution of the vertex of index i along the time by

$$V_i(i) = \{V(i, 0), V(i, 1), \dots, V(i, T - 1)\}, \quad (7)$$

with $V(i, t)$ the positioning of this vertex at the frame t . For a given vertex i , such a temporal filtering provides 2 sets:

- the set of temporal details (or *high frequency* coefficients) $h^{(1)}(i)$ relative to the set of odd samples $\{V(i, 2k + 1)\}$ (the upper index (1) represents the decomposition level) ;
- the set of *low frequency* coefficients $l^{(1)}(i)$ relative to the set of even samples $\{V(i, 2k)\}$.

Applying this filtering in parallel on each vertex provides that the input sequence of T meshes is finally splitted in two subbands:

- the subband $h^{(1)}$ defined by $T/2$ detail frames;
- the subband $l^{(1)}$ defined by $T/2$ approximation frames.

One can obtain a multiresolution decomposition by subsequent filterings of the approximation subband. This decomposition consists in N subbands of detail frames $\{h^{(r)}\}$ (with r the resolution level), and the approximation subband

$l^{(N)}$. Fig. 7 shows the resulting 2-level decomposition of the FACE animation.

Numerous wavelet families exist depending on their mathematical properties, but also on their method of construction [24]. In this paper, we focus on the *lifting implementation* [25], because it has several advantages over classical constructions: it allows easier and faster implementation (low computational cost), a fully in-place calculation, and the synthesis step can be directly deduced from the analysis step. Moreover, lifting-based implementation of any transversal wavelet filter can be designed [26].

6.2 Lifting Implementation

If we consider a filter bank whose lifting implementation corresponds to a single lifting step (see Fig. 8), then the temporal details and the low frequency coefficients are obtained from the following equations

$$h^{(1)}(i, k) = V(i, 2k + 1) - P(\{V(i, 2k)\}), \quad (8)$$

$$l^{(1)}(i, k) = V(i, 2k) + U(h^{(1)}(i)), \quad (9)$$

where P and U are respectively the predict and update operators. Fig. 9 shows for instance the principle of the lifting scheme $[2, 0]$ with 2 levels of decomposition. Note that the synthesis step is easily obtained reversing the same lifting steps with opposite signs:

$$V(i, 2k) = l^{(1)}(i, k) - U(h^{(1)}(i)), \quad (10)$$

$$V(i, 2k + 1) = h^{(1)}(i, k) + P(\{V(i, 2k)\}). \quad (11)$$

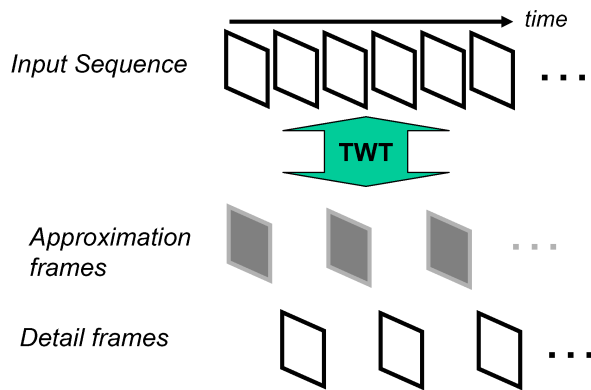


Fig. 5. Temporal wavelet filtering, one level of decomposition.

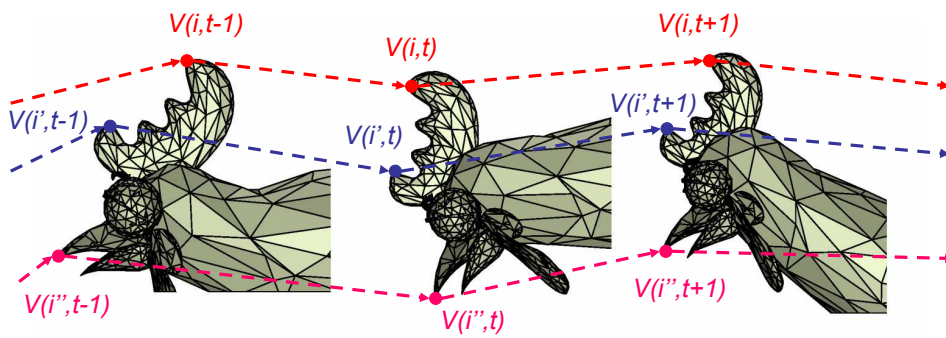


Fig. 6. Proposed temporal wavelet filtering for animated sequences of meshes sharing the same connectivity. The previous and next positionings of one vertex are used to compute the associated wavelet coefficient at the instant t .

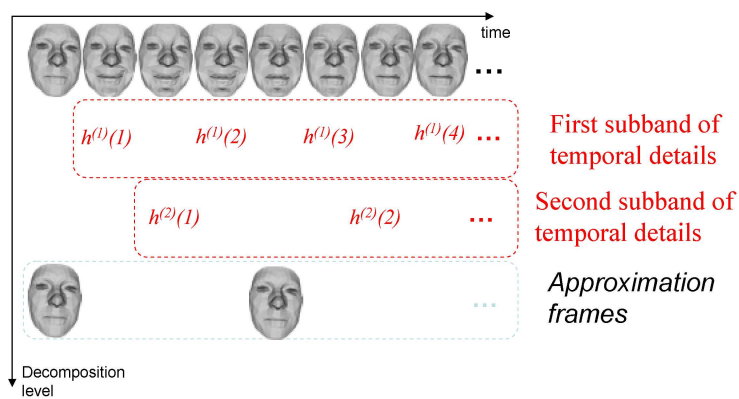


Fig. 7. The 2-level decomposition of the FACE animation.

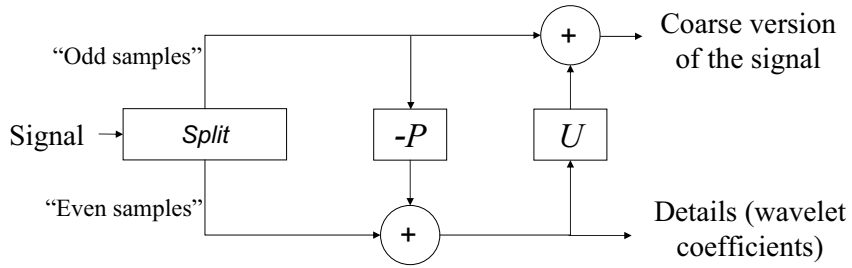


Fig. 8. Principle of the lifting implementation.

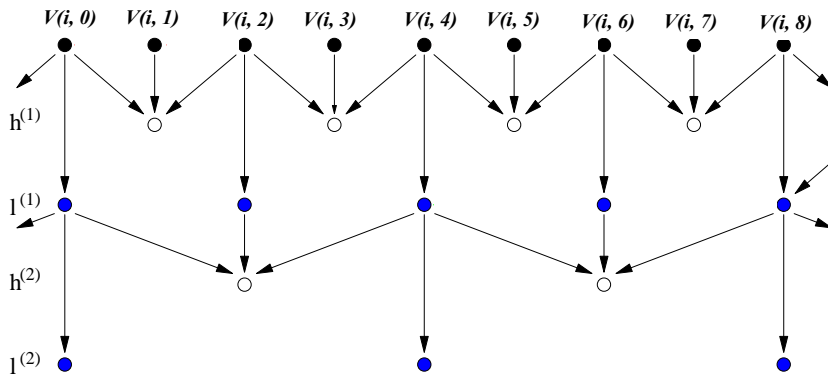


Fig. 9. Principle of the lifting scheme $[2, 0]$ with 2 levels of decomposition. $V(i, t)$ is the position of the vertex i in the frame t .

7 Experimental results

First we compare the efficiency of several classical lifting schemes often used in video compression [13,27]: the schemes [2, 0], [2, 2], [4, 2] and [6, 2]. The corresponding coefficients of the predict and update operators are given in Tab. 2. Nevertheless, other wavelet families may be more relevant in animation compression, for instance the spline wavelets particularly useful for representing multiresolution curves [28]. In this way, a deep algebraic and comparative analysis of different wavelet families is one of the most interesting direction for future works.

We study three well-known animations with different characteristics (see Tab. 3): FACE, COW and CHICKEN. To evaluate the coding performances of our coder, we use the metric error introduced by Karni and Gotsman in [5]. In the rest of the paper, this metric is called *KG error*, and is expressed in percent. This metric, corresponding to the relative discrete L_2 -norm both in time and space, is given by

$$KG\ error = 100 \frac{\|G - \hat{G}\|}{\|G - E(G)\|}, \quad (12)$$

where G is a matrix of dimension $(3 \times n_v, T)$ containing the geometry of the original sequence, \hat{G} the quantized version of the geometry, and $E(G)$ an average matrix in which the t^{th} column is defined by

$$\left(\bar{X}_t(1 \dots 1), \bar{Y}_t(1 \dots 1), \bar{Z}_t(1 \dots 1) \right)^t, \quad (13)$$

with \bar{X}_t , \bar{Y}_t , and \bar{Z}_t the mean values of the coordinate sets of each frame t .

Fig. 10, 11 and 12 show the curves *KG Error/bitrate* of our coder for the three sequences according to the different lifting schemes (with three levels of decomposition). The bitrate is given in bits per vertex per frame. We observe that the scheme [2, 0] provides the worst coding performances, whereas the schemes [4, 2] or [6, 2] provides the best performances. This is due to the fact that the schemes [4, 2] and [6, 2] take into account a larger neighborhood, and consequently leads to a better prediction.

Nevertheless, the schemes [2, 2], [4, 2] and [6, 2] provide similar results at low

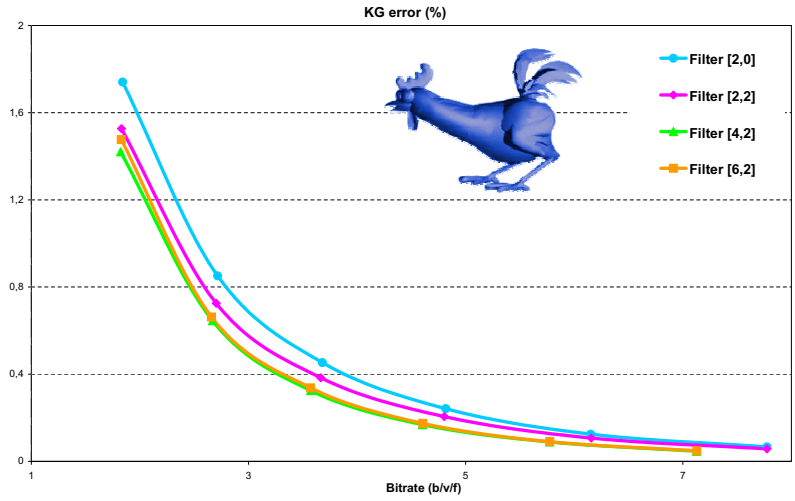


Fig. 10. Curve $KG Error/bitrate$ for the CHICKEN animation according to different lifting schemes.

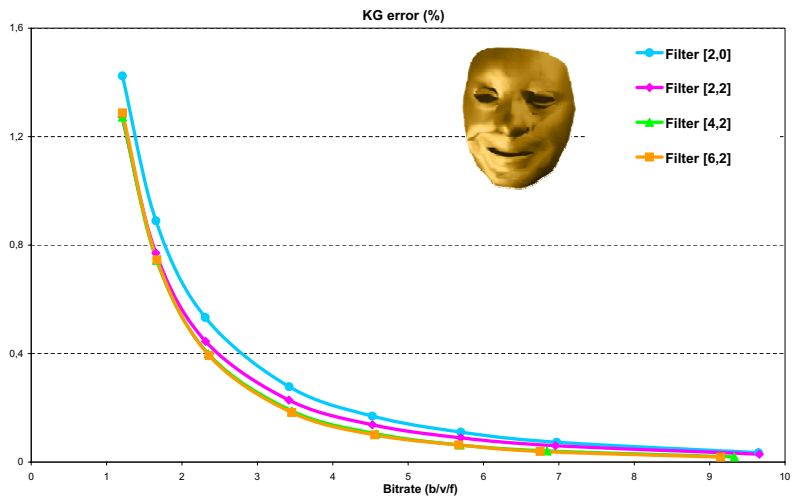


Fig. 11. Curve $KG Error/bitrate$ for the FACE animation according to different lifting schemes.

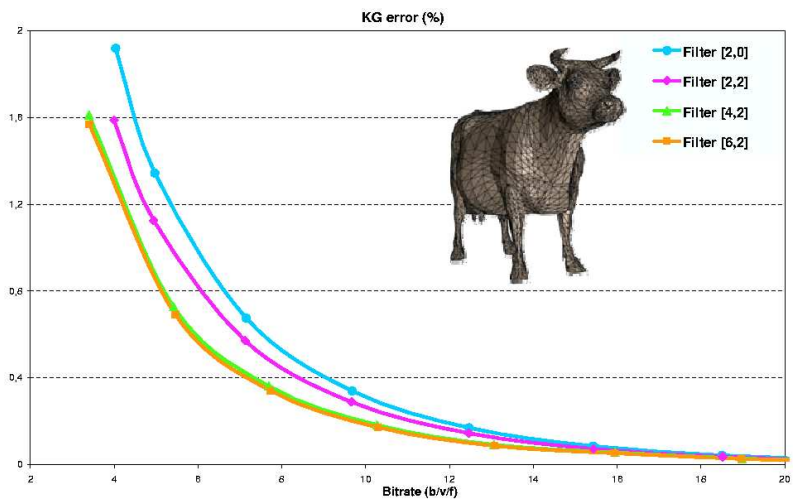


Fig. 12. Curve $KG Error/bitrate$ for the COW animation according to different lifting schemes.

bitrates for the FACE animation. The very small vertex displacements between subsequent frames of this animation involves that the prediction step provides similar results (same wavelet coefficients), whatever the complexity of the operator P used.

As the filter $[4, 2]$ requires less computing resources in processing time and memory usage than the filter $[6, 2]$ (25% fewer operations, see Tab. 4) for similar results, we finally conclude that the filter $[4, 2]$ is the most interesting among the experimented filters.

We also study the influence of the number of decomposition levels. Fig 13, 14 and 15 show the curves $KG \text{ Error}/\text{bitrate}$ for the three same sequences in function of the number of decomposition levels (when using the filter $[4, 2]$). For the FACE, the best results are obtained with 7 decomposition levels. For the two other animated sequences, the best results are obtained with 4 decomposition levels. This difference is mainly due to the repetitive nature of the FACE, and the spatially restricted displacements of the vertices. These characteristics imply a larger temporal coherence (compared to the sequences CHICKEN and COW).

To show the interest of our approach, we also compare the performances of our coder with several state of the art coders:

- the coder for static meshes of Touma and Gotsman [19] denoted by TG ;
- the PCA-based coder for mesh sequences of Alexa and Müller [4] denoted by PCA ;
- the coder for mesh sequences of Karni and Gotsman [5] denoted by KG . It combines the coder PCA to a linear prediction coding;
- the coder *Dynapack* of Ibarria and Rossignac [9];
- the spatial wavelet-based coder of Khodakovsky and Guskov [10], denoted by AWC .

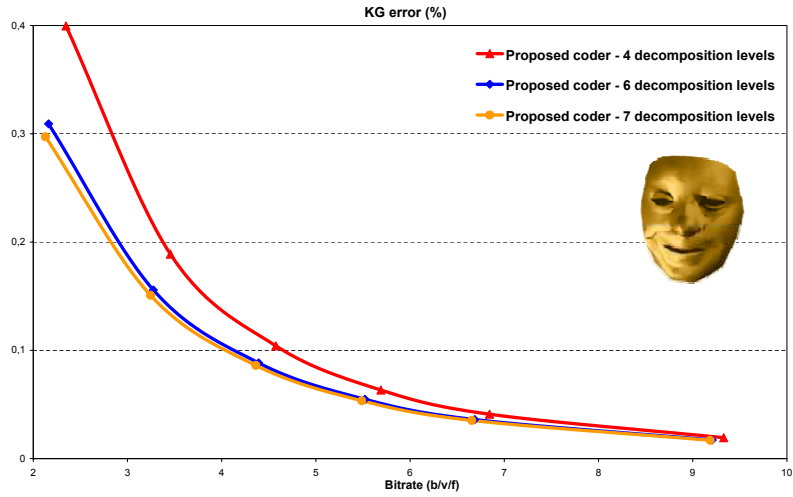


Fig. 13. Curve $KG Error/bitrate$ for FACE according to different numbers of decomposition levels (filter [4, 2]).

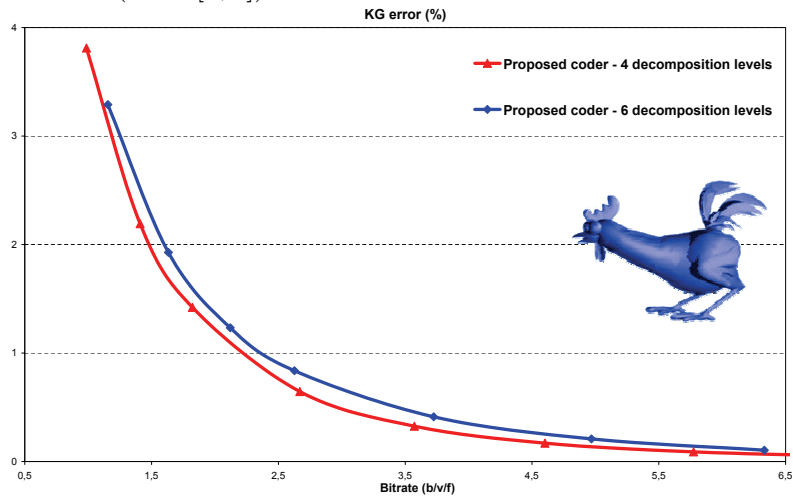


Fig. 14. Curve $KG Error/bitrate$ for CHICKEN according to different numbers of decomposition levels (filter [4, 2]).

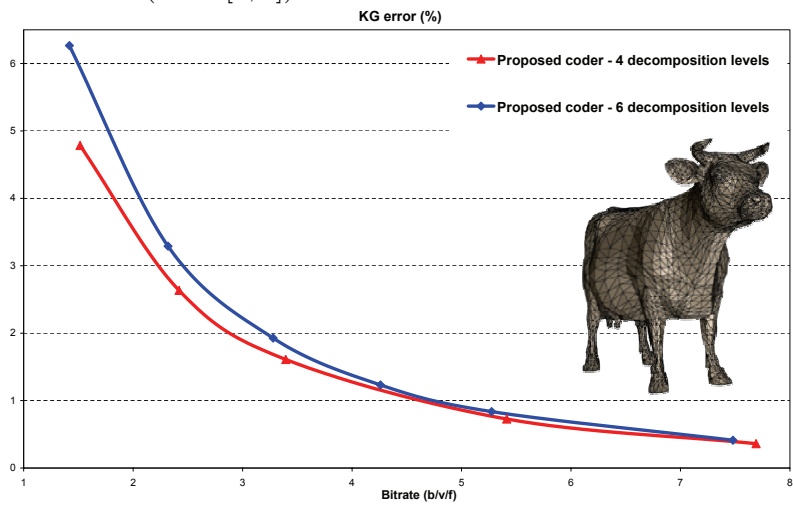


Fig. 15. Curve $KG Error/bitrate$ for COW according to different numbers of decomposition levels (filter [4, 2]).

All the results relative to these methods are extracted from [5] and [10], that is why some results are missing. This is also why we only use the *KG error* of [5] as quality criterion. Notice that the bit budget needed to encode the connectivity of the animated sequences is included in the total bitrate.

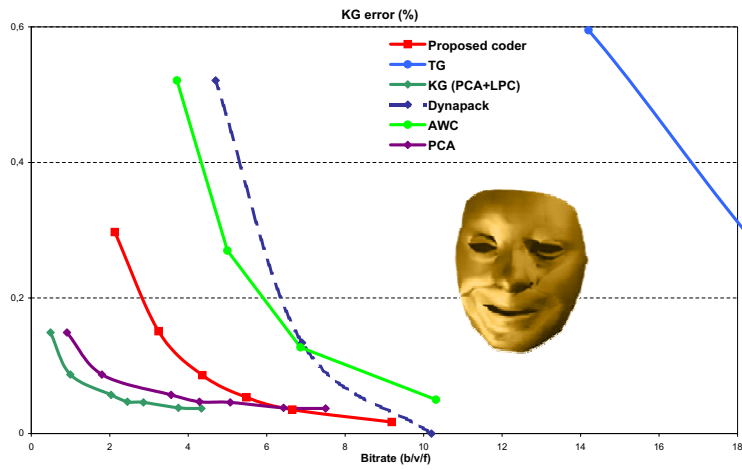
First, Fig. 16(a) shows the curve *KG Error*/bitrate for the sequence FACE. We observe that the PCA-based methods (*PCA* and *KG*) provide the best coding performances. This is not surprising given the characteristics of this sequence. It turns out that the *PCA*-based methods are particularly performant for long sequences of relatively coarse meshes [29]. On the other hand, our method is significantly more efficient than the other methods, and especially the coder *AWC*. Given the coarseness of this animated sequence (small number of vertices), the spatial multiresolution nature of this coder cannot be fully exploited. As expected, the proposed temporal wavelet filtering does not depend on such characteristics. It shows one of the advantages of a temporal decomposition over a spatial decomposition.

Now, we observe the coding performances of our approach on the COW (Fig. 16(b)), and on the CHICKEN (Fig. 16(c)). We recall that these animations are defined by short sequences of finer meshes and present much larger deformations than the FACE. First, we observe that for these two sequences our method clearly outperforms the PCA-based methods². It turns out that the PCA-based methods are inefficient for sequences with fine meshes. This is due to the size of the “payload” archive of eigenmodes that becomes too large in case of short sequences [5]. Our simple temporal wavelet filtering do not provide such a side information during decompression. This shows the interest of our approach over the PCA-based methods.

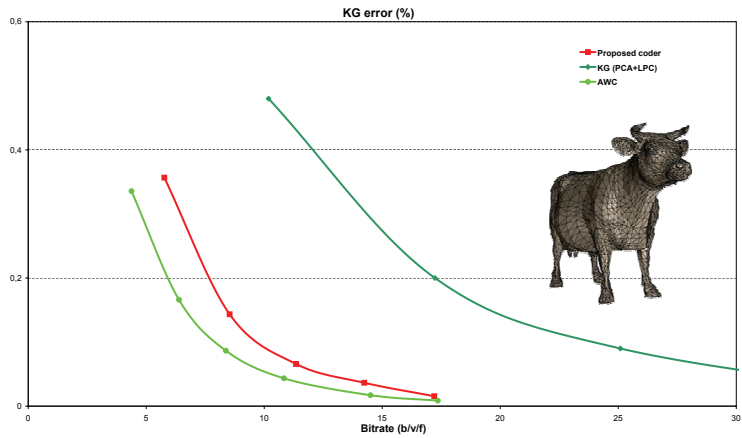
Second, we observe our coding scheme provides better coding performances than *AWC* for the FACE, but not for the COW. We assume this is due to the very small local deformations of the COW, involving the parametric coherence, frame after frame. Nevertheless, compared to *AWC*, our method gives very competitive coding performances.

Concerning the sequence of the CHICKEN, we cannot compare our coder with *AWC* (Fig. 16(c)), because the results are not given in the corresponding pa-

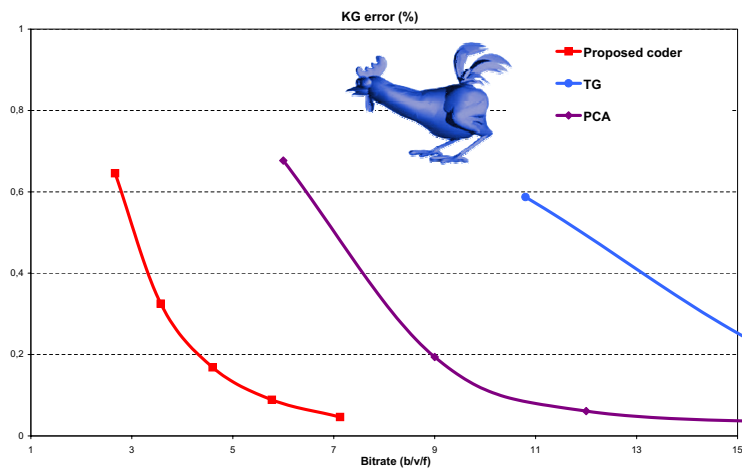
² The results of *KG* for the CHICKEN is missing because the authors claim that their method is not relevant for such a data. Consequently, the numerical values are not given in their paper [5].



(a) FACE (7 decomposition levels).



(b) Cow (4 decomposition levels).



(c) CHICKEN (4 decomposition levels).

Fig. 16. Curves *KG Error*/bitrate relative to different compression methods.

per [10]. Anyway, the method AWC is inefficient for such a sequence because of the lack of parametric coherence and the characteristics of this animated sequence (non manifold and sharp features) [17].

From a compression point-of-view, we can conclude that our approach is competitive for any kind of animated sequences, and particularly relevant when the meshes are fine (more powerful than the PCA-based methods) and non parametrically coherent (more powerful than the method AWC).

8 Conclusions and future works

In this paper we have introduced a simple and efficient wavelet-based coder for any kind of animated sequences of irregular meshes sharing a same connectivity. One key contribution of this work is to propose a temporal wavelet filtering to process the geometry component of such data. Such a temporal analysis has the advantage to overcome the problem of the parametric coherence of a given animation assumed by previous wavelet coders in animation compression. The second key contribution is a model-based bit allocation that optimizes the quantization process. The resulting coder is simple, fast, and flexible (any 1D wavelet can be used). Experimentally, we show that our approach is always valid and competitive, whatever the characteristics of the animated mesh sequences (contrary to the state-of-the-art methods).

There are a lot of interesting directions for future works. The most important is a deep algebraic and comparative analysis of the numerous 1D wavelet families (orthogonal, spline...). Our future works should also concern the encoding of the approximation frames. Currently, the solution is crude but sufficient to exploit the model-based algorithm of [15] during the allocation process. Nevertheless, some more relevant solutions may improve the coding performances of our coder since the spatial coherence of the approximation frames are not fully exploited (using the works of Sorkine [30, 31] for instance). Another interesting work should be a specific implementation to obtain a bitstream structure standing the temporal scalability, but also the random access across the time. This should be evident since our approach already allows the temporal scalability. Furthermore, this new implementation should not increase significantly the bitstream size.

9 Acknowledgements

The sequence CHICKEN is the property of Microsoft Inc. and the sequence FACE was kindly generated by Demetri Terzopoulos. We are particularly grateful to Zachi Karni for providing us with the sequences CHICKEN and FACE, and Igor Guskov for providing us with the sequence COW and the results of his method [10]. We are also particularly grateful to the master student Yasmine Boulfani for providing us with some simulation results of the proposed method. All the results of the state of the art methods are extracted from the paper [5] or provided by Igor Guskov. We also want to acknowledge the anonymous reviewers for their advices, which permitted to improve the quality of this paper.

References

- [1] P. Alliez, C. Gotsman, Recent advances in compression of 3D meshes, in: Proceedings of the Symposium on Multiresolution in Geometric Modeling, 2003.
- [2] J. E. Lengyel, Compression of time-dependent geometry, in: ACM Symposium on Interactive 3D Graphics, 1999, pp. 89–96.
- [3] A. Shamir, V. Pascucci, Temporal and spatial level of details for dynamic meshes, in: Proceedings of Virtual Reality Systems and Techniques, 2001, pp. 423–430.
- [4] M. Alexa, W. Muller, Representing animations by principal components, *Comput. Graph. Forum* 19, 3.
- [5] Z. Karni, C. Gotsman, Compression of soft-body animation sequences, *Computers and Graphics* 28 (2004) 25–34.
- [6] J. Ahn, C. Kim, C. Kuo, Y. Ho, Motion compensated compression of 3D animation models, *IEEE Electronics Letters* 37 (24) (2001) 1445–1446.
- [7] J. Yang, C. Kim, S.-U. Lee, Compression of 3D triangle mesh sequences based on vertex-wise motion vector prediction, *IEEE Trans. Circuits Syst. Video Technol.* 12 (12) (2002) 1178–1184.
- [8] H. Briceno, P. Sander, L. McMillan, S. Gotler, H. Hoppe, Geometry videos: a new representation for 3D animations, *ACM Symp. Computer Animation* (2003) 136–146.

- [9] L. Ibarria, J. Rossignac, Dynapack: space-time compression of the 3D animations of triangle meshes with fixed connectivity, *ACM Symp. Computer Animation* (2003) 126–135.
- [10] I. Guskov, A. Khodakovsky, Wavelet compression of parametrically coherent mesh sequences, in: *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, 2004.
- [11] D. Turaga, M. van der Schaar, B. Pesquet-Popescu, Complexity scalable motion compensated wavelet video encoding, *IEEE Transactions on Circuits and Systems for Video Technology* 15 (8) (2005) 982–993.
- [12] N. Mehrseresht, D. Taubman, Spatial scalability and compression efficiency within a flexible motion compensated 3d-dwt, in: *Proc. IEEE International Conference on Image Processing (ICIP2004)*, Singapore, 2004.
- [13] M. Cagnazzo, T. André, M. Antonini, M. Barlaud, A model-based motion compensated video coder with jpeg2000 compatibility, in: *Proc. IEEE International Conference on Image Processing (ICIP2004)*, Singapore, 2004.
- [14] J.-H. Yang, C.-S. Kim, S.-U. Lee, Progressive compression of 3D dynamic sequences, in: *Proc. ICIP 2004*, 2004.
- [15] F. Payan, M. Antonini, An efficient bit allocation for compressing normal meshes with an error-driven quantization, *Computer Aided Geometric Design, Special Issue on Geometric Mesh Processing* 22 (2005) 466–486.
- [16] Z. Karni, C. Gotsman, 3D mesh compression using fixed spectral basis, *Graphics interface Conference Proceedings* (2001) 1–8.
- [17] I. Guskov, Private communication.
- [18] F. Payan, M. Antonini, 3D multiresolution context-based coding for geometry compression, in: *Proceedings of IEEE International Conference in Image Processing (ICIP)*, Barcelona, Spain, 2003.
- [19] C. Touma, C. Gotsman, Triangle mesh compression, *Graphics Interface'98* (1998) 26–34.
- [20] F. Payan, M. Antonini, Mean square error approximation for wavelet-based semiregular mesh compression, *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 12 (2006) 649–657.
- [21] S. Mallat, A theory for multiresolution signal decomposition: the wavelet representation, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11 (7) (1989) 674–693.

- [22] F. Payan, Y. Boulfani, M. Antonini, Temporal lifting scheme for the compression of animated sequences of meshes, in: Proceedings of IEEE International Workshop VLBV 2005, Sardinia, Italy, 2005.
- [23] F. Payan, M. Antonini, Wavelet-based compression of 3d mesh sequences, in: Proceedings of IEEE ACIDCA-ICMI'2005, Tozeur, Tunisia, 2005.
- [24] M. Vetterli, C. Herley, Wavelet and filter banks: theory and design, IEEE transactions on Acoustic Speech Signal Processing 40 (9) (1992) 2207–2232.
- [25] W. Sweldens, The lifting scheme: A custom-design construction of biorthogonal wavelets, Appl. Comput. Harmon. Anal. 3 (2) (1996) 186–200.
- [26] I. Daubechies, W. Sweldens, Factoring wavelet transforms into lifting steps, Tech. rep. (1996).
URL citeseer.ist.psu.edu/daubechies96factoring.html
- [27] A. Calderbank, I. Daubechies, W. Sweldens, B. Yeo, Wavelet transforms that map integers to integers, Applied and Computational Harmonic Analysis 5 (3) (1998) 332–369.
- [28] E. J. Stollnitz, T. DeRose, D. H. Salesin, Wavelets for Computer Graphics: Theory and Applications, Morgan Kaufmann Publishers, San Francisco, CA, USA, 1996.
- [29] Z. Karni, C. Gotsman, Spectral compression of mesh geometry, In ACM SIGGRAPH Conference Proceedings (2000) 279–286.
- [30] O. Sorkine, D. Cohen-Or, S. Toledo, High-pass quantization for mesh encoding, in: Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing, Eurographics Association, 2003, pp. 42–51.
- [31] D. Chen, D. Cohen-Or, O. Sorkine, S. Toledo, Algebraic analysis of high-pass quantization, ACM Transactions on Graphics 24 (4) (2004) 1259–1282.

Lifting Scheme	Predict operator
[2, 0]	$[-\frac{1}{2} \quad -\frac{1}{2}]$
[2, 2]	$[-\frac{1}{2} \quad -\frac{1}{2}]$
[4, 2]	$[\frac{1}{16} \quad -\frac{9}{16} \quad -\frac{9}{16} \quad \frac{1}{16}]$
[6, 2]	$[-\frac{3}{256} \quad \frac{25}{256} \quad -\frac{75}{128} \quad -\frac{75}{128} \quad \frac{25}{256} \quad -\frac{3}{256}]$
Lifting Scheme	Update operator
[2, 0]	—
[x , 2]	$[\frac{1}{4} \quad \frac{1}{4}]$

Table 2
Predict and update operators of different lifting schemes.

Sequence	T	n_v
CHICKEN	399	2916
FACE	10001	539
COW	204	2904

Table 3
Characteristics of the animations.

Lifting Scheme	[2, 0]	[2, 2]	[4, 2]	[6, 2]
Complexity	6 <i>ovf</i>	12 <i>ovf</i>	18 <i>ovf</i>	24 <i>ovf</i>

Table 4
Computational cost in operations per vertex per frame (*ovf*) according to different schemes.