# Wavelet-based Compression of 3D Mesh Sequences

Frédéric Payan, 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 original wavelet-based compression algorithm for sequences of 3D meshes with fixed connectivity. This algorithm, based on a temporal lifting scheme exploits the high temporal coherence of the geometry of successive frames in order to reduce the information needed to represent the original sequence. The resulting sets of wavelet coefficients are then optimally quantized by the mean of a bit allocation process. The proposed allocation process allows to dispatch the bit budget across the wavelet sequences according to their influence on the quality of the reconstructed sequence for one specific user-given bitrate. Simulation results show that the proposed algorithm provides better compression performances than some state of the art coders.**

## I. Introduction

Today animated sequences of 3D meshes (Fig. 1) are more and more exploited to represent realistic visual data in many domains: computer games, character animation, physical simulations... Such data are generally represented by a sequence of irregular meshes of fixed connectivity, and could be thus encoded frame by frame by any compression method for static irregular meshes. Since the connectivity remains the same along the sequence, a more relevant approach is to consider the animated sequences of meshes as geometry deformations of one single static mesh (e.g. the first frame of the sequence). An efficient way to compress animated sequences is thus to encode the first frame, and then the geometric displacements of the vertices from frame to frame. This can be done according to several techniques [1]–[9].

The first works concerning the compression of animated sequences of 3D meshes with a fixed connectivity were done by [1], [2] and exploited the affine transformations. In [1] Lengyel proposed to split a mesh into several submeshes, and considered each submesh has a rigid-body motion. 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 also proposed an approach based on the affine transformations, but in the same time exploited a multiresolution approach [2]. Their technique was to find the best affine transformations between each frame and the first one, and to encode the temporal prediction errors. Other predictive coding schemes have been then proposed [5], [6], [8] in order to exploit the temporal and spatial correlations of animated mesh sequences. These works were proposed to predict all the vertex positions or displacements, and finally encode the residual errors. In parallel, Alexa and Müller [3] 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) [10]. In [7] Briceno *et al.* presented an original approach. The technique was to project each frame onto a 2D image, and then encode the whole ”2D image sequence” with some well-known video techniques. Recently, a wavelet-based compression method was presented by Guskov and Khodakovsky in [9]. They proposed to apply a multiresolution analysis on the frames to exploit the spatial coherence, and progressively encode the resulting details with a predictive coding scheme.

In this paper, we describe an original approach based on a temporal wavelet transform to encode animated sequences of meshes. We propose precisely to use a monodimensional lifting scheme applied directly on the positions of the vertices across time in order to exploit the high temporal correlation between the geometry of the successive frames.

The rest of the paper is organized as follows. Section II presents the proposed approach. Section III describes the main steps of our compression algorithm based on a temporal wavelet transform. Section IV gives then a detailed description of the proposed temporal wavelet transform. Section V presents the allocation process included in the proposed coder. Simulation results of the proposed coder are given and compared to results of several state of the art methods in Section VI. Finally, we conclude and propose future works in Section VII.

## II. Proposed Approach

As shown in the previous section, there are few works about compression of animated mesh sequences. Moreover, to our knowledge, despite the efficiency of the wavelet-based compression algorithms (in image or video processing since several decades and in static mesh processing since several years) only Guskov and Khodakovsky proposed a wavelet-based approach [9]. In their works, a wavelet transform is used to exploit the spatial coherence of a single frame. 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

Fig. 1. Several frames of the animated sequence of 3D meshes called DOLPHIN.

encoded thanks to a predictive coding. This approach is thus a *3D+t* wavelet-based approach, *i.e* a spatial wavelet transform followed by a coding exploiting the temporal coherence.

Since it has been recognized that the *t+2D* wavelet-based approaches are generally more efficient than the *2D+t* ones [11], we propose in this paper a *t+3D* wavelet-based coder for mesh sequences with fixed connectivity. More precisely we exploit a monodimensional lifting scheme directly applied on the positions of the vertices across time in order to exploit the temporal coherence. The different details subbands and the main temporal components are then encoded with a model-based coder initially proposed for the static semiregular meshes [12].

## III. Overview of our compression algorithm

The proposed compression algorithm is presented in Fig. 2.

The main steps are:

- **Temporal wavelet transform**: a temporal wavelet transform is first applied on the original mesh sequence. If a multilevel wavelet decomposition is processed, the original sequence is transformed in a *low frequency (LF) sequence* and several sequences of *high frequency (HF) details*.
- **Scalar quantization (SQ)**: each sequence is splitted in 3 subbands of coordinates. These subbands will be treated separately in the rest of the algorithm. The quantization parameters are determined by an allocation process.
- **Bit allocation**: this is an essential step of the proposed coding scheme. The allocation process allows to solve the rate-distortion problem relative to the data quantization, that is minimize the losses due to the quantization process and, in the same time, minimize the bitstream size of the compressed sequence. The allocation process proposed here dispatches the bit budget across the wavelet sequences according to their influence on the quality of the reconstructed mesh sequence for one specific bitrate.
- **Entropy coding**: Once quantized, the wavelet coefficients are entropy coded to produce the bitstream. For this, a simplified version of the context-based arithmetic coder of [13] is used.
- **Connectivity coding**: in order to reconstruct the quantized mesh sequence after decompression, the connectivity of the original mesh sequence must be also encoded and transmitted. Since the connectivity is the same for each sequence frame, we simply encode the connectivity of the first frame with the coder of Touma and Gotsman [14].

## IV. Temporal Wavelet Transform for 3D Mesh Sequences

### A. Principle of the Lifting Scheme

The lifting scheme is a second-generation wavelet transform that easily provides a multiresolution representation of signals, and enables decorrelation in space and frequency [15]. During analysis, the idea of a lifting scheme is to first split the original data in 2 subbands: the first subband contains the samples of odd indices and the second one contains the samples of even indices (Fig. 3). Two operators are then used. The *prediction* operator $P$ is first applied to obtain the *HF* subband (or wavelet coefficients). The *update* operator $U$ is then applied to obtain the *LF* signal (in other words a coarser representation of the original data). During the synthesis step, the process order and the sign of the operators only need to be inverted to obtain the original data from the wavelet coefficients.

There are a lot of lifting schemes, which depend on the dimension of the neighborhood used to compute the wavelet coefficients and the *LF* signal. So, a lifting scheme is generally defined by a pair $[n, m]$, where $n$ and $m$ are respectively the dimension of the prediction operator and the dimension of the update operator [16].

### B. Temporal Lifting Scheme

The lifting scheme can be exploited to decorrelate data in space and frequency, but in case of signals with a temporal

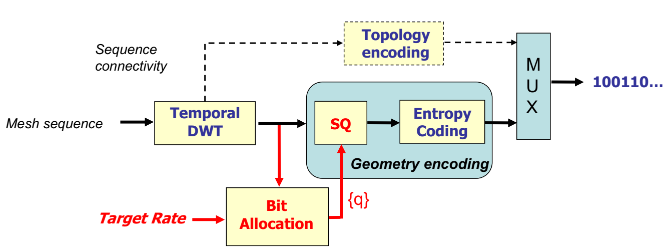



Fig. 2. Overall coding scheme.

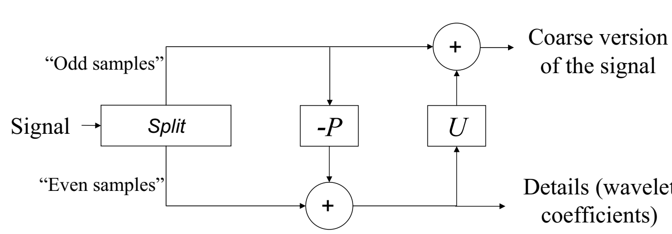



Fig. 3. Principle of the lifting scheme.

dimension, that is a signal $dD + t$ ($d$=1,2, or 3), a lifting scheme can be also applied along the time axis (see Fig. 4). This approach, called *temporal lifting scheme* allows to exploit the high temporal coherence existing in the processed data in order to reduce the information needed to represent the original sequence.

To our knowledge, the temporal lifting scheme has never been exploited in compression of such animated sequences of meshes (*i.e.* each frame of the sequence has the same connectivity). In the next section, we present how a temporal lifting scheme can be applied to exploit the temporal coherence of the geometry of an animated sequence of meshes.

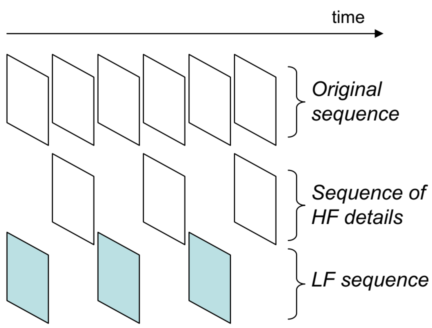



Fig. 4. Temporal wavelet transform on one level.

### C. Proposed Temporal Lifting Scheme

Formally, an animated mesh sequence is represented by a set of $T$ static meshes $\{f_1, f_2, ..., f_T\}$, called *frames*. A frame $f_i$ is defined by its geometry, *i.e.* the set of vertex coordinates at time $i$, and a list of triangles describing how the vertices are connected. In this paper, we focus only on the animated mesh sequences with fixed connectivity. The list of triangles $T_i$ is consequently the same for all the frames.

As a lot of related papers [1], [3]–[8], we consider the animated sequences of meshes as geometry deformations of one single static mesh (the first frame of the sequence). The main idea is thus to apply a monodimensional lifting scheme on the successive positions of each vertex (see Fig. 5). Note that, contrary to most of video coders we do not have to use a motion estimation algorithm to match the treated vertices with vertices of previous frames before applying the temporal wavelet transform [11]. We emphasize that the data processed are geometric positions in space. As the connectivity is the same for each frame involving a fixed number of vertices along the sequence, the motion of each vertex is implicit.

The principle of the proposed approach is the following. The evolution of the vertex of index $i$ along the time is defined by

$$V_t(i) = \{V(i,0), V(i,1), ..., V(i,T-1)\},$$

with $V(i,t)$ its euclidean position at the instant $t$ (corresponding to the frame $t$). For each vertex $i$,

1) The set $V_t(i)$ is first splitted in two sets, in function of the indices: the set of "even samples" $\{V_e(i)\}$ and the set of "odd samples" $\{V_o(i)\}$:

$$\{V_e(i)\} = \{V(i,2k)\} \quad (1)$$
$$\{V_o(i)\} = \{V(i,2k+1)\} \quad (2)$$

with $k = [0..nbv/2]$, $nbv$ being the number of vertices of each frame.

2) For each odd sample $V(i,2k+1)$, the associated wavelet coefficient $h(i,k)$ is computed in function of the set of even samples ($\{V_e(i)\}$) and of the prediction operator $P$ :

$$h(i,k) = V(i,2k+1) - P(\{V_e(i)\}) \quad (3)$$

So, we obtain the set of wavelet coefficients $h^{(1)}(i)$ relative to the vertex $i$.

3) The *LF* coefficient $l(i,k)$ relative to each even sample $V(i,2k)$ is now computed in function of the update operator $U$ and of the wavelet coefficients previously computed:

$$l(i,k) = V(i,2k) + U\left(\{h^{(1)}(i)\}\right) \quad (4)$$

So, we obtain the set of *LF* coefficients $l^{(1)}(i)$ relative to the vertex $i$.

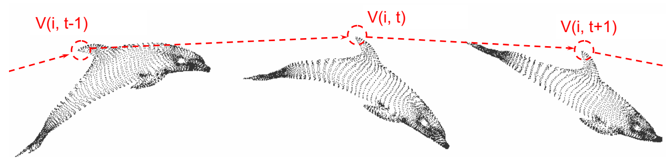



Fig. 5. Proposed approach. The previous and next positions of one vertex are used to compute the associated wavelet coefficient at the instant $t$.

Fig. 6 shows the principle of the lifting scheme $[2,0]$ with 2 levels of decomposition [11], [16], applied on the different positions of the $i^{th}$ vertex.

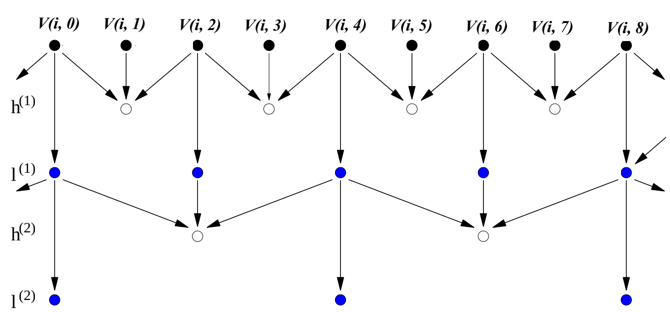



Fig. 6. A 2-level decomposition, with the lifting scheme $[2,0]$. $V(i,t)$ is the position of the vertex $i$ in the frame $t$.

Hence, after a one-level decomposition, that is once this process applied on each vertex, the sequence of $T$ meshes is splitted in two "subsequences":

- a sequence of $T/2$ *HF* detail sets denoted by $h^{(1)}$;
- the *LF* sequence of $T/2$ meshes denoted by $l^{(1)}$, that is a coarse version of the original sequence.

By applying $N$ times such a decomposition on the *LF* sequence previously computed, a multilevel decomposition is obtained, with $N$ sequences of *HF* details $\{h^{(r)}\}$ (with $r$ the resolution index), and a coarse version $l^{(N)}$ of the original sequence. For instance, Fig. 7 shows the resulting data of a 3-level decomposition on some frames of the sequence FACE.

### *D. Complexity*

One advantage of this temporal lifting scheme is its low computational cost. During the analysis, the computation of the wavelet coefficients (Prediction step) with a filter $[n, m]$ represents only $2n$ arithmetic operations per sample. In parallel, the computation of the *LF* data (Update step) needs only $2m$ arithmetic operations per sample. So, the computational cost of the analysis of the proposed lifting scheme for a whole mesh sequence is $2 * (n + m) * T/2 * nbv * 3$ arithmetic operations, or equivalently $3(n+m)$ arithmetic operations per vertex per frame ($ovf$). The computational cost of the synthesis is obviously the same. Table I gives the computational cost of several lifting schemes frequently used. The complexity is given in $ovf$.

This represents finally a very low computational cost contrary to other analysis methods like [3], [4], [9]. Moreover, during decompression no side information is needed to reconstruct the mesh sequence, contrary to the methods based on *PCA* [3], [4], which need an important "payload" due to the transmission of the basis vectors.

As a result, compared to other well-known approaches based on an analysis tool [3], [4], [9], our method requires much less computing resources in processing time ($3(n+m)$ $ovf$ for the analysis but also for the synthesis, which is a great advantage.

### *E. Comparison of different Lifting Schemes*

As explained in Section IV-A, several lifting schemes exist, depending on the dimension of the prediction and update operators. Here, we compare the efficiency of different lifting schemes (by using the coder presented in the next sections). We particularly deal with the schemes $[2, 0]$, $[2, 2]$, $[4, 2]$, and $[6, 2]$ of [11], [16] and exploit a 4-level decomposition. For the comparison, we use three well-known sequences with different features (see Table II): FACE, COW and CHICKEN.

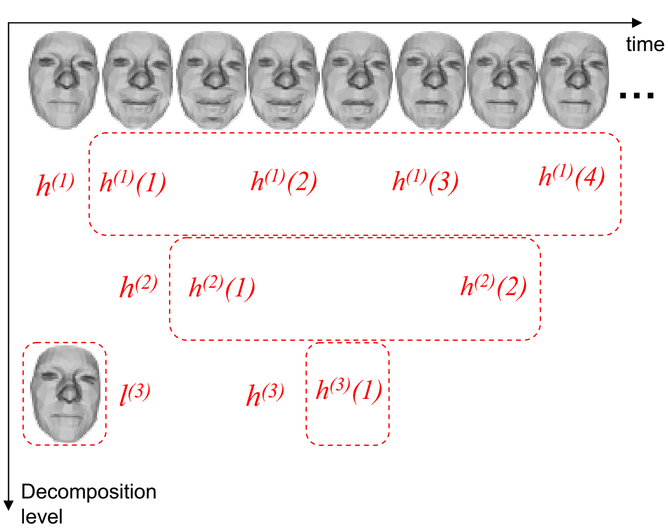



Fig. 7. 3-level decomposition of 8 frames of the sequence FACE.

| Lifting Scheme | $2,0$ | $2,2$ | $4,2$ | $6,2$ |
|---|---|---|---|---|
| Complexity | 6 $ovf$ | 12 $ovf$ | 18 $ovf$ | 24 $ovf$ |

TABLE I

COMPUTATIONAL COST ACCORDING TO DIFFERENT SCHEMES.

To evaluate the performances of the different lifting schemes, we use the metric error introduced by Karni and Gotsman in [4]. 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

$$KG\ error = 100\frac{||G - \hat{G}||}{||G - E(G)||}, \tag{5}$$

where $G$ is a matrix of dimension $(3 \times nbv, T)$ 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

$$\left(\bar{X}_t\,(1\ ...\ 1)\,,\bar{Y}_t\,(1\ ...\ 1)\,,\bar{Z}_t\,(1\ ...\ 1)\right)^T, \tag{6}$$

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

Fig. 8, 9 and 10 show the curves *KG Error*/bitrate for the three sequences according to the different lifting schemes and the proposed coder presented in section III. The bitrate is given in bits per vertex per frame (reached by the mean of the allocation process described in Section V). Globally, we observe that the proposed coder exploiting the scheme $[2, 0]$ provides the worst coding performances, whereas the best coding performances are obtained when the schemes $[4, 2]$ or $[6, 2]$ are used.

We point out that we obtain the expected results. Actually, the prediction operators of the schemes $[4, 2]$ and $[6, 2]$ "capture" more efficiently the vertex displacements than the schemes $[2, 0]$ and $[2, 2]$ since the latter take into account less neighbor samples. So, the prediction errors, *i.e.*, the wavelet coefficients are smaller, and the coding scheme is finally more efficient.

| Sequence | # frames | # vertices by frame |
|---|---|---|
| CHICKEN | 399 | 2916 |
| FACE | 10001 | 539 |
| COW | 204 | 2904 |

TABLE II

FEATURES OF THE SEQUENCES USED.

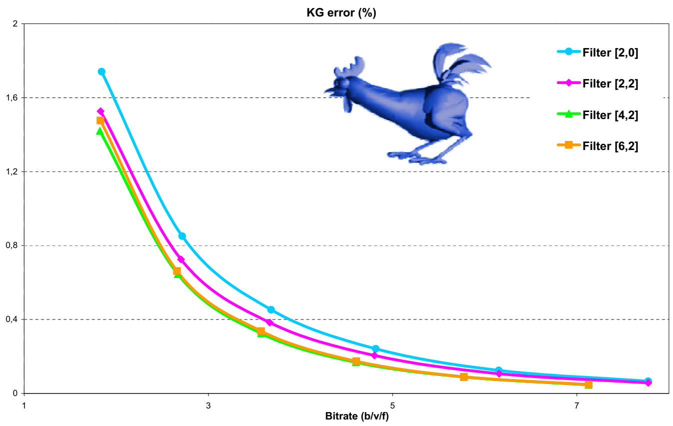



Fig. 8. Curve *KG Error*/bitrate for CHICKEN according to the different lifting schemes and the proposed coder.

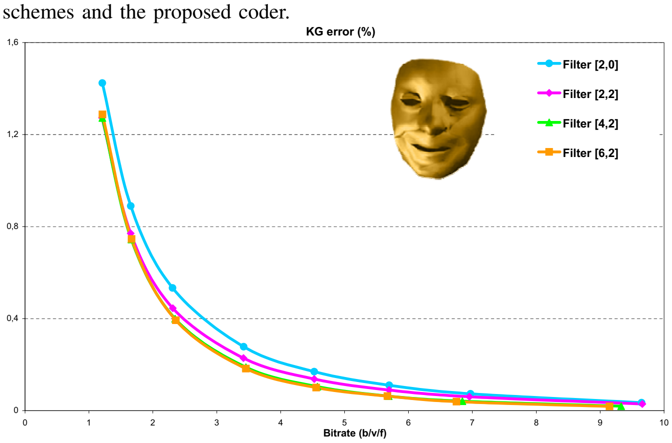



Fig. 9. Curve *KG Error*/bitrate for FACE according to the different lifting schemes and the proposed coder.

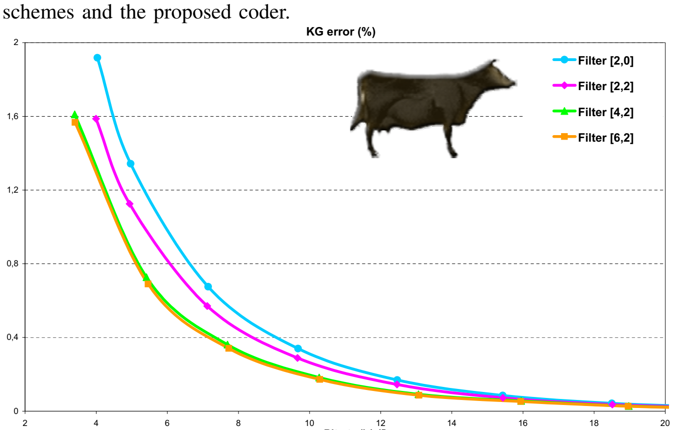



Fig. 10. Curve *KG Error*/bitrate for COW according to the different lifting schemes and the proposed coder.

However, at low bitrates we observe for the sequence FACE that the schemes $[2,2]$, $[4,2]$ and $[6,2]$ provide similar results. This is understandable, because globally this sequence does not present a high motion. The vertex displacements being small between successive frames, the prediction operators of the filters $[2,2]$, $[4,2]$ and $[6,2]$ give almost the same wavelet coefficients. Furthermore, at low bitrates the coefficients are so coarsely quantized that the proposed coder finally gives similar coding performances whatever the filter used.

As the filter $[4,2]$ requires less computing resources in processing time and memory usage than the filter $[6,2]$ ($25\%$ fewer operations) for similar results, we conclude it is more relevant to use the filter $[4,2]$ for an efficient encoding of the animated mesh sequences used here.

## V. BIT ALLOCATION

### A. Principle

Once the temporal wavelet transform applied on the original sequence, the resulting sequences of wavelet coefficients and the *LF* one need to be encoded. To optimize the quantization process, we introduce a bit allocation in the compression algorithm.

The objective of an allocation process is to optimize the trade-off between rate and distortion (after lossy compression and decompression). In these works, the lifting scheme transforms the original sequence in several detail sequences. So, our allocation process aims to dispatch the bit budget across the wavelet sequences according to their influence on the quality of the quantization of the mesh sequence for one specific bitrate.

More precisely, the allocation process presented here allows to compute the set of optimal quantizers $\{q^*\}$, which minimizes 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

$$(\mathcal{P}) \left\{ \begin{array}{ll} \text{minimize} & D_T\left(\{q\}\right) \\ \text{with constraint} & R_T\left(\{q\}\right) = R_{target}, \end{array} \right. \quad (7)$$

with $R_T$ the total bitrate.

### B. Optimal solution

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

$$J_\lambda(\{q\}) = D_T\left(\{q\}\right) + \lambda(R_T\left(\{q\}\right) - R_{target}), \quad (8)$$

with $\lambda$ the lagrangian operator.

The optimal quantization steps $\{q^*\}$ are obtained by solving the following system [12]:

$$\left\{ \begin{array}{l} \frac{\partial J_\lambda(\{q\})}{\partial q} = 0 \\ \frac{\partial J_\lambda(\{q\})}{\partial \lambda} = 0 \end{array} \right. . \quad (9)$$

As the wavelet transform is processed in parallel on the three coordinates of the vertex positions, we propose to encode separately the three sets of coordinates of each sequence with different non uniform scalar quantizers (SQ). So, the reconstructed mean square error can be defined by

$$D_T(\{q\}) = \sum_{i=0}^{N} w_i \sum_{j=1}^{3} D_{i,j}(q_{i,j}), \qquad (10)$$

with $\{w_i\}$ the weights due to the non-orthogonality of the lifting schemes used [17] ($i$ represents the resolution level), $D_{i,j}$ the mean square error relative to the coordinate set $i, j$ and $q_{i,j}$ the associated quantization step ($j = 1$ for the $x$-coordinates, $j = 2$ for the $y$-coordinates, and $j = 3$ for the $z$-coordinates). In parallel, the total bitrate $R_T$ can be developed in

$$R_T(\{q\}) = \sum_{i=0}^{N} \sum_{j=1}^{3} a_{i,j} R_{i,j}(q_{i,j}), \qquad (11)$$

with $\{a_{i,j}\}$ the coefficients depending on the subsampling, and corresponding to the ratio between the size of the $(i,j)^{th}$ set of coordinates and the total number of samples [12]. By merging (10) and (11) in (8) and developing the system (9), we obtain the following system of $(3(N+1))$ equations with $(3(N+1)+1)$ unknowns (the set $\{q_{i,j}\}$ and $\lambda$):

$$\frac{\frac{\partial D_{i,j}(q_{i,j})}{\partial q_{i,j}}}{\frac{\partial R_{i,j}(q_{i,j})}{\partial q_{i,j}}} = -\lambda \frac{a_{i,j}}{w_i} \qquad (12a)$$

$$\sum_{i=0}^{N} \sum_{j \in J_i} a_{i,j} R_{i,j}(q_{i,j}) = R_{target}. \qquad (12b)$$

In order to obtain the optimal quantization steps analytically, (12a) requires to be inverted. Unfortunately, this stage is impossible due to the complexity of the equations. To overcome this problem, an iterative algorithm depending on $\lambda$ is generally proposed [12].

### C. Overall Algorithm

The optimal solutions of system (12) for the given bitrate $R_{target}$ are then computed thanks to the following overall algorithm:

1) $\lambda$ is given. For each set $(i, j)$, compute $q_{i,j}$ that verifies (12a);
2) while (12b) is not verified, calculate a new $\lambda$ by dichotomy and return to step 1;
3) stop.

The computation of the quantization steps $\{q_{i,j}\}$ as solutions of (12a) can be done according to different methods. As we observe the probability density function (PDF) of each *HF* coordinate set can be modeled by a Generalized Gaussian Distribution (GGD), the model-based allocation process and the associated algorithm presented in [12] can be applied. See [12] for more details.

### D. Encoding of the LF sequence

The PDF of the three coordinate sets of the *LF* sequence cannot be modeled by a GGD. To overcome this problem, we use a *differential coding*. The main idea of such a coding is to encode the differences between the samples instead of the samples themselves [18]. It has been shown in previous works [12] that using such a coding involves that the PDF of the LF data can be finally modeled by a GGD.

In this paper, we propose a *Geometric differential (GD) coding*. The idea is to process the differences between coordinates in function of the vertex indexes, that is *for each frame* $t$, the coder deals with the sequence

$$[V(0,t) - V(1,t), V(1,t) - V(2,t), ..., V(nbv-1,t) - V(nbv,t)].$$

So, we exploit the spatial correlation existing between neighbor vertices.

## VI. Comparison with other Coders

In this section we compare the coding performances of the proposed coder with some state of the art coders:

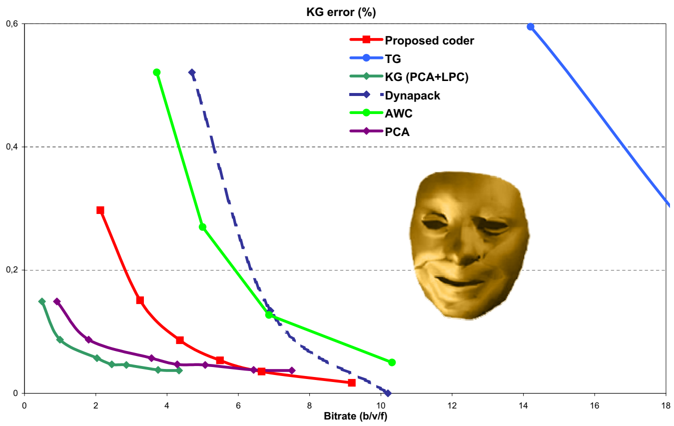
- the coder for static meshes of Touma and Gotsman [14] denoted by *TG*;
- the PCA-based coder for mesh sequences of Alexa and Müller [3] denoted by *PCA*;
- the coder for mesh sequences of Karni and Gotsman [4] denoted by *KG*. It includes the *PCA* approach of [3] and a linear prediction coding;
- the coder *Dynapack* of Ibarria and Rossignac [8];
- the wavelet-based coder of Khodakovsky and Guskov [9], denoted by *AWC*.

All the results relative to these methods are extracted from [4] and [9]. Therefore some results are missing.

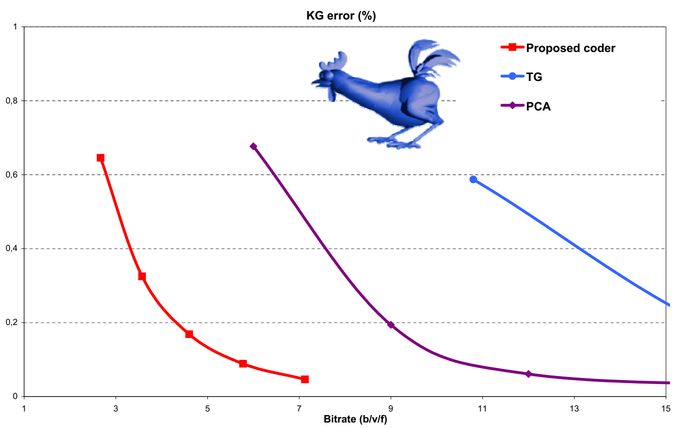
Figure 11(a) shows that for the sequence Face our method is more efficient than the methods *TG*, *Dynapack* and *AWC*. On the other hand, the methods *PCA* and *KG* provide significant better coding performances than our algorithm. This is understandable given the specific features of this sequence: the number of frames is much higher than the number of vertices, which is an high advantage for the *PCA*-based methods [4].

In parallel, for the sequence Chicken (Fig. 11(b)) we observe that our method is significantly better than *TG* but also better than *PCA*. Contrary to Face, the sequence Chicken has a much smaller number of frames than vertices, and it is recognized that the methods *PCA* or *KG* are not suitable for such cases [4]. Besides, we observe that our method also gives better results than *KG* (and logically *PCA*) for Cow (Fig. 11(c)) which has features similar to Chicken.

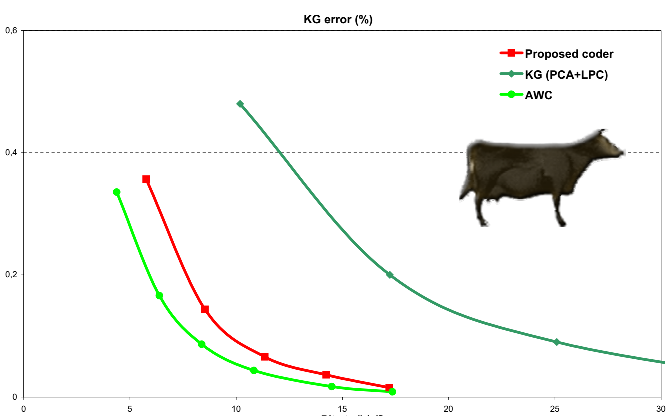
On the other hand, we observe that the proposed algorithm gives better coding performances than *AWC* for Face, but not for Cow. This is because *AWC* exploits a spatial multiresolution analysis for each frame (contrary to the proposed method which exploits a temporal multiresolution analysis). Such a spatial multiresolution analysis is efficient only for highly detailed meshes, *i.e.*, with a high number of vertices [9].




(a) FACE (7 decomposition levels).




(b) CHICKEN (4 decomposition levels).




(c) COW (4 decomposition levels).

Fig. 11. Curves *KG Error*/bitrate relative to different compression methods.

## VII. CONCLUSIONS AND FUTURE WORKS

In this paper we introduce a temporal lifting scheme for the geometry of animated mesh sequences (with a fixed connectivity), the final objective being an efficient wavelet-based compression algorithm for such data. The advantage of this lifting scheme is to strongly reduce the information needed to represent the geometry of a mesh sequence by exploiting its temporal coherence. We show experimentally that the filter $[4, 2]$ is the filter offering the best trade-off between coding performances and computational cost. Furthermore, we show that the proposed temporal lifting scheme associated to our optimal coder (that includes a model-based bit allocation optimizing the quantization process) provides better coding performances than several state of the art coders.

In addition, the proposed lifting scheme has the advantage to be simpler and faster than the other analysis tools (used by the state of the art coders we studied) during decompression but also during compression.

There are a lot of perspectives for such a coding scheme. The main one is about the encoding of the *LF* sequence. In this paper the multiresolution analysis is only processed along the time axis thanks to the proposed temporal lifting scheme. So, the spatial coherence of the *LF* sequence is not fully exploited. An additional spatial multiresolution analysis of the *LF* sequence should improve the coding performances of the coder proposed in this paper.

## VIII. 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 [9]. 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 [4] or provided by Igor Guskov.

## REFERENCES

[1] J. E. Lengyel, “Compression of time-dependent geometry,” in *ACM Symposium on Interactive 3D Graphics*, 1999, pp. 89–96.

[2] A. Shamir and V. Pascucci, “Temporal and spatial level of details for dynamic meshes,” in *Proceedings of Virtual Reality Systems and Techniques*, 2001, pp. 423–430.

[3] M. Alexa and W. Muller, “Representing animations by principal components,” *Comput. Graph. Forum 19,*, vol. 3, 2000.

[4] Z. Karni and C. Gotsman, “Compression of soft-body animation sequences,” *Computers and Graphics*, vol. 28, pp. 25–34, 2004.

[5] J. Ahn, C. Kim, C. Kuo, and Y. Ho, “Motion compensated compression of 3D animation models,” *IEEE Electronics Letters*, vol. 37, no. 24, pp. 1445–1446, Nov 2001.

[6] J. Yang, C. Kim, and S.-U. Lee, “Compression of 3D triangle mesh sequences based on vertex-wise motion vector prediction,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp. 1178–1184, Dec. 2002.

[7] H. Briceno, P. Sander, L. McMillan, S. Gotler, and H. Hoppe, “Geometry videos: a new representation for 3D animations,” *ACM Symp. Computer Animation*, pp. 136–146, 2003.

[8] L. Ibarria and J. Rossignac, "Dynapack: space-time compression of the 3D animations of triangle meshes with fixed connectivity," *ACM Symp. Computer Animation*, pp. 126–135, 2003.
[9] I. Guskov and A. Khodakovsky, "Wavelet compression of parametrically coherent mesh sequences," in *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, August 2004.
[10] Z. Karni and C. Gotsman, "3D mesh compression using fixed spectral basis," *Graphics interface Conference Proceedings*, pp. 1–8, 2001.
[11] M. Cagnazzo, T. André, M. Antonini, and M. Barlaud, "A model-based motion compensated video coder with jpeg2000 compatibility," in *Proc. IEEE International Workshop on Multimedia Signal Processing*, Singapore, Oct. 2004.
[12] F. Payan and 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*, vol. 22, pp. 466–486, July 2005.
[13] ——, "3D multiresolution context-based coding for geometry compression," in *Proceedings of IEEE International Conference in Image Processing (ICIP)*, Barcelona, Spain, sept. 2003.
[14] C. Touma and C. Gotsman, "Triangle mesh compression," *Graphics Interface'98*, pp. 26–34, 1998.
[15] W. Sweldens, "The lifting scheme: A construction of second generation wavelets," *SIAM Journal on Mathematical Analysis*, vol. 29, no. 2, pp. 511–546, 1998. [Online]. Available: citeseer.nj.nec.com/sweldens98lifting.html
[16] A. Calderbank, I. Daubechies, W. Sweldens, and B. Yeo, "Wavelet transforms that map integers to integrers," *Applied and Computational Harmonic Analysis*, vol. 5, no. 3, pp. 332–369, 1998.
[17] F. Payan and M. Antonini, "MSE approximation for model-based compression of multiresolution semiregular meshes," in *Proceedings of IEEE EUSIPCO 2005 (13th European Conference on Signal Processing)*, Antalya, Turkey, september 2005.
[18] ——, "Multiresolution 3D mesh compression," in *Proceedings of IEEE International Conference in Image Processing (ICIP)*, Rochester, USA, september 2002.