MOTION-BASED GEOMETRY COMPENSATION FOR DWT COMPRESSION OF 3D MESH SEQUENCES

Yasmine Boulfani-Cuisinaud and Marc Antonini, member, IEEE

Laboratoire I3S - UMR 6070 CNRS, Université de Nice - Sophia Antipolis
2000, Route des Lucioles - 06903 Sophia Antipolis FRANCE
phone: +33 4 92 94 27 85, fax: +33 4 92 94 28 98, email: {boulfani,am}@i3s.unice.fr

ABSTRACT

In this paper, We propose an efficient compression method to encode the geometry of 3D mesh sequences of objects sharing the same connectivity. Our approach is based on the clustering of the input mesh geometry into groups of vertices following the same affine motion. The proposed algorithm uses a scan-based temporal wavelet filtering geometrically compensated. The wavelet coefficients are encoded by an efficient coding scheme that includes a bit allocation process, whereas the displacement vectors are lossless entropy encoded. Simulation results provides good compression performances compared to some state of the art coders.

Index Terms— Scan-based compression, mesh sequences, motion-based clustering, geometry compensation, temporal wavelet, lifting scheme.

1. INTRODUCTION

3D animated objects are used in a variety of fields, like computer games, multimedia, medical imaging,... They are often represented by a sequence, stored and transmitted as series of consecutive triangular meshes generally called frames, where each frame is defined by the location of the vertices (geometry) and by triangles (connectivity). In general, the meshes are irregular, and the connectivity of the meshes may also evolve with time. In this paper, we restrict our attention to a class of animation meshes sharing the same connectivity at any frame of the sequence.

Most of the compression methods of the literature, proposed for 3D animations, exploit the affine transformations of different segments of the meshes [1, 2, 3]. Also, some approaches proposed to predict the vertex displacements along the sequence and then to encode the residual errors [4, 5]. Recently, several more complex prediction techniques have been proposed. In [6, 7] for instance, the authors exploited the temporal coherence by clustering vertices with similar affine transforms between successive frames. In [8], the authors used the mesh connectivity to determine the order of compression of vertex locations. In parallel, Alexa and Müller [9] proposed a coding scheme based on the principal component analysis (PCA) to represent a mesh sequence 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 predictive coding called LPC [10]. The method of [9] has been also improved in [11], where the authors proposed to cluster vertices before using PCA. Briceno *et al.* presented an original approach in [12]. The technique is to project each frame onto a 2D image, and then to encode the resulting sequence of "2D images" with some well-known video techniques. Besides, several methods based on wavelets have been proposed. In [13], the authors proposed to exploit the temporal coherence of the geometry components by applying a B-spline wavelet transform on the successive vertex positions. Recently, a coder based on temporal wavelet filtering implemented in lifting scheme has been proposed [14]. In [15], Guskov and Khodakovsky proposed to exploit the parametric coherence of some specific animations by combining a spatial multiresolution analysis and 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 algorithm, but for sequences of irregular meshes with changing connectivity [16]. Finally, in [17] Mamou and $al.$ recently proposed a novel approach for 3D mesh compression, based on a skinning animation technique. Their method is based on a piecewise affine predictor coupled with a skinning model and a DCT representation of the residuals errors. In this paper, we propose a compression scheme based on a temporal scan-based discrete wavelet transform (DWT). It includes motion-based clustering of the mesh sequence and a geometrically compensated DWT (see figure 1). Similarly to several techniques, we propose an efficient way to compress animated sequences by considering the sequence as geometric deformations of the geometry of a key frame.

The rest of this paper is organized as follows. Section 2 deals with the motion-based mesh clustering, the scan-based processing and the geometry compensation approach. Section 3 presents the proposed geometry compensated DWT. In section 4 experimental results are given and compared to some state of the art methods. We finally conclude and propose future works in section 5.

2. MOTION-BASED MESH CLUSTERING AND GEOMETRY COMPENSATION

2.1. Motion-based mesh clustering

In this paper, we propose an approach to construct a partition for each connex component of the 3D mesh of the sequence, based on the motion between consecutive frames. This approach consists in constructing a partition of the vertices according to the motion. In other words, vertices with same motion belong to the same cluster of the partition. The number of clusters in the partition depends completely on the motion nature between the two frames. Let us introduce some notations.
Let $\mathcal{F}$ be the set composed by the T frames of the mesh sequence, $\mathcal{F}$ is defined by:

$$\mathcal{F} = \{f_1, f_2, ..., f_t, ...f_T\}. \quad (1)$$

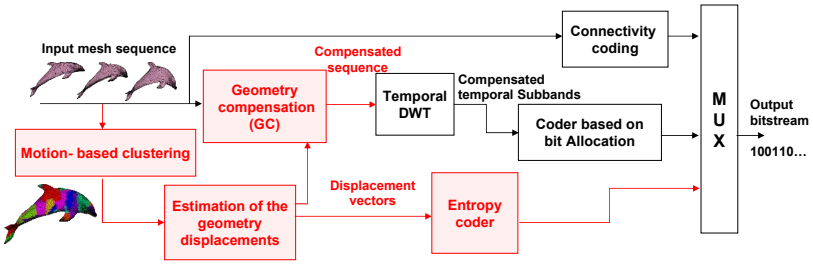

Fig. 1. General structure of the proposed compression algorithm.

Let us also defined $\mathcal{V}_t$ the set of V vertices of each frame t of the sequence by[1]:

$$\mathcal{V}_t = \{v_t^1, v_t^2, ..., v_t^i, ..., v_t^V\}, \ \forall t, \tag{2}$$

and $\overline{\mathcal{V}}^i$ the set of neighbor vertices of v_t^i (which belong to the same connex component). The set $\overline{\mathcal{V}}^i$ is identical for all frame t of the sequence (with fixed connectivity) and is written as:

$$\overline{\mathcal{V}}^i = \{(\overline{v}_t^i)^k\}_{k=1,...,K}, \ \forall t, \tag{3}$$

with K the number of the neighbor vertices of v_t^i, such as $K \leq V$. Finally, the partition for each frame t is denoted by:

$$\mathcal{C}_t = \{C_t^1, C_t^2, ..., C_t^n, ..., C_t^N\}, \tag{4}$$

with N being the number of clusters in the frame t of the sequence, which depends on the motion between the frame t and the frame $t-1$. Let us suppose now that the vertices i at times t and $t-1$ are linked by the following relation:

$$v^i{}_t = M_t^{v^i} \times v^i{}_{t-1}, \ \forall \, v^i{}_{t-1} \in \mathcal{V}_{t-1} \text{ and } v^i{}_t \in \mathcal{V}_t, \tag{5}$$

where $M_t^{v^i}$ is an affine transform matrix which transforms the vertex v^i, given in homogeneous coordinates[2], from the frame $t-1$ to the frame t. It is of size (4 x 4) and it contains 12 coefficients representing the motion (rotation, translation and scaling):

$$M_t^{v_i} = \begin{pmatrix} & [R,S]_{3\times3} & & [Tr]_{3\times1} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6}$$

where the matrix $[R,S]_{3\times3}$ contains the rotation (R) and scaling (S) parameters, and $[Tr]_{3\times1}$ is the translation vector. In order to find the matrix $M_t^{v_i}$, we need 4 vertices. For this purpose, we choose the vertex $v_t^i \in \mathcal{V}_t$ and 3 vertices $(\overline{v}_t^i)^q$ chosen randomly among the set of the first order neighborhood vertices of v_t^i. We choose the first order neighborhood in order to have a precise estimation of the motion.

Let us denote by $\mathcal{Q}_t^{v^i}$ the 4×4 matrix containing the 4 neighbor vertices given in homogeneous coordinates. This matrix can be written as:

$$\mathcal{Q}_t^{v^i} = \begin{pmatrix} v_t^i & (\overline{v}_t^i)^1 & (\overline{v}_t^i)^2 & (\overline{v}_t^i)^3 \end{pmatrix}$$

where v_t^i and $(\overline{v}_t^i)^q$ are to be column vectors of size (4×1) (homogeneous coordinates). In that way, the equation (5) can be rewritten as follows:

$$\mathcal{Q}_t^{v^i} = M_t^{v_i} \times \mathcal{Q}_{t-1}^{v^i} \tag{7}$$

[1]Since the connectivity remains the same for all the frames of the sequence, the number V is constant whatever t.

[2]A vertex v given in homogeneous coordinates is written as $v^t = (x, y, z, 1)$.

Or equivalently to:

$$M_t^{v_i} = \mathcal{Q}_t^{v^i} \times (\mathcal{Q}_{t-1}^{v^i})^{-1} \tag{8}$$

In order to solve the equation (7), and find an estimation of the matrix $M_t^{v_i}$, we introduce the matrix $\mathcal{P}_t^{v^i}$ of size $4 \times (Q+1)$, where Q is the number of the first order neighborhood vertices of v_t^i belonging to $\overline{\mathcal{V}}^i$, such that $Q \leq K \leq V$. So, $\mathcal{P}_t^{v^i}$ contains the vertex v_t^i and all its first order neighborhood vertices. It is given by (with vertex expressed in homogeneous coordinates):

$$\mathcal{P}_t^{v^i} = \begin{pmatrix} v_t^i & (\overline{v}_t^i)^1 & ... & (\overline{v}_t^i)^q & ... & (\overline{v}_t^i)^Q \end{pmatrix} \tag{9}$$

where v_t^i and $(\overline{v}_t^i)^q$ are column vectors of size (4×1).
The transformation which is solution of the equation (7), corresponds to the matrix $M_t^{v_i} = M_t^{v_i *}$ minimizing among all the possible neighbors $(\overline{v}_t^i)^q$, q from 1 to Q, chosen randomly among the first order neighborhood vertices of v_t^i, the following formula:

$$\|\mathcal{P}_t^{v^i} - M_t^{v_i} \cdot \mathcal{P}_{t-1}^{v^i}\|^2 \tag{10}$$

Once the affine transform $M_t^{v_i^*}$ associated to the vertex v_t^i is found, the cluster $C_t^n \in \mathcal{C}_t$ of the frame t, represented by the vertex v_t^i, will contain the vertex v_t^i and all of its neighbor vertices $(\overline{v}_t^i)^k$ among $\overline{\mathcal{V}}^i$ which verify the following equation:

$$\|M_t^{v_i^*} \cdot (\overline{v}_{t-1}^i)^k - (\overline{v}_t^i)^k\|^2 < \varepsilon, \tag{11}$$

with ε a threshold (chosen empirically in our experiments). A cluster C_t^n can be written as:

$$C_t^n = \{(v_t^j)^n\}_{j=1,...,V'} \tag{12}$$

where the quantity V' ($V' \leq K \leq V$) is the number of vertices in the cluster C_t^n which verify the equation (11). The partition verifies the following properties:

$$\cup_n C_t^n = f_t \ \text{ and } C_t^n \cap C_t^m = \emptyset, \ \forall m, n \tag{13}$$

This process is repeated for all the vertex v_t^l of the frame f_t ($l \neq i$ and $v_t^l \notin C_t^n$).

2.2. Scan-based processing

In the proposed approach, the sequence is processed on the fly (also called scan-based processing). To this purpose, the sequence is decomposed into groups of frames (GOF) where each GOF is processed and computed independently. In the rest of the paper, we consider that a GOF is composed by T frames. To simplify, the clustering process is done for the first two frames of each GOF of the sequence and is maintained constant in the whole GOF. Then, the number N of clusters per frame remains the same in a current GOF. As an example, the clustering of the first frame of the SNAKE sequence is shown in the figure 2, for $N = 21$ clusters.

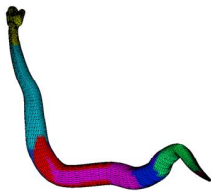

Fig. 2. Clustering of the SNAKE sequence, $N = 21$ clusters.

2.3. Geometry compensation

The geometry compensation approach, denoted in the rest of paper by GC, consists in displacing all the frames of a given GOF on the key frame (here first frame) of the GOF, while keeping this key frame unchanged. Once the set of clusters $(C_t^n)_{n\in\{1....N\}}$ is computed, the proposed GC approach is given as follows.

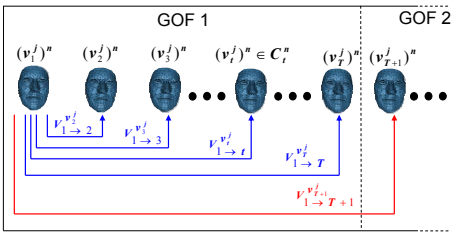

Fig. 3. Estimation of the geometric displacement vectors. To compute the scan-based wavelet transform (Section 3) and avoid boundary effects, we need to use the first frame of the next GOF, which is also displaced on the first frame of the current GOF.

The proposed idea consists to compute one average displacement vector per cluster $\mathcal{C}_t^n$, as it is shown in the figure 3. An average displacement vector corresponds to the translation motion computed for each frame and for each cluster according to the first frame of the GOF. It can be written as:

$$\bar{V}_{1\to t}^{\mathcal{C}_t^n} = \frac{1}{V'} \sum_{v_t^j \in \mathcal{C}_t^n} V_{1\to t}^{v_t^j}, \text{ where } V_{1\to t}^{v_t^j} = v_t^j - v_1^j, \qquad (14)$$

for $j = 1, 2, ..., V'$, $n = 1, 2, ..., N$ and $t = 2, 3, ..., T$.
Once the suitable displacement vectors of the geometry are computed for each cluster $\mathcal{C}_t^n$ of each frame t, we can apply the GC on this GOF. Let us denote the new compensated cluster by:

$$\hat{C}_t^n = \{(\hat{v}_t^j)^n\}_{j=1,...,V'} \qquad (15)$$

Each compensated vertex $\hat{v}_t^j \in \hat{C}_t^n$ is computed as follows:

$$\hat{v}_t^j = v_t^j - \bar{V}_{1\to t}^{\mathcal{C}_t^n}, \qquad (16)$$

for $j = 1, 2, ..., V'$, $n = 1, 2, ..., N$ and $t = 2, 3, ..., T$.
The temporal wavelet transform is then done on the compensated sequence as explained in the following section 3.

3. GEOMETRY COMPENSATED DWT

A practical problem in temporal wavelet transform implementation is related to memory requirements of time filters. Usually, wavelet transform is implemented by loading all the data in memory and then performing filtering. In the case of temporal filtering of mesh sequences, this would require a huge memory size and moreover could imply an encoding delay as long as the sequence duration itself. As said in section 2.2, a simple solution to the temporal filtering problem is to crop the input sequence in several short subsequences called GOF. Once each GOF of the original sequence is geometry compensated, we can apply the scan-based wavelet transform on the compensated sequence. The figure 4 shows the principle of the scan-based filtering on the FACE sequence for a GOF of 4 frames, by using the lifting scheme $(2, 0)$ as proposed in [14]. As we mentioned in section 2.3, in order to solve the boundary problem, we need to consider the first frame of the next GOF to compute the filtering of the last frame of the current GOF.

The hight frequency (HF) details and the low frequency (LF) sequence of each GOF are encoded with an efficient bit allocation-based coder [14]. As the connectivity remains the same for all frames of the mesh sequence, we simply encode the connectivity of the first frame with the efficient coder of Touma and Gotsman [18].

Sequence	F	V	T	CC	N
DOLPHIN	64	6179	12337	1	89
CHICKEN	384	2916	5454	37	37

Table 1. Considered features of the used sequences.

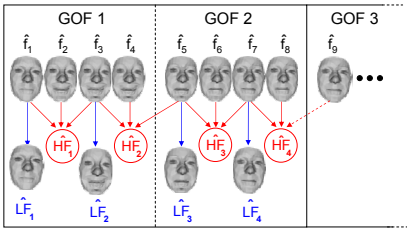

Fig. 4. Example of a scan-based wavelet transform decomposition (lifting scheme $(2, 0)$ - 1 level decomposition)

4. SIMULATION RESULTS

In order to evaluate the efficiency of the proposed compression scheme, we have tested its performances on different sequences. Here, we present two animation sequences: DOLPHIN and CHICKEN sequences, with different features presented in Table 1. In this table, F corresponds to the number of frames, V to the number of vertices per frame, T to the number of triangles per frame, CC to the number of the connex components of the original mesh, and N to the average number of clusters obtained for all the GOFs of the sequence. To evaluate the quality between the original sequences and the reconstructed ones, we use the metric error called *KG error*, introduced by Karni and Gotsman in [10].
Figure 5 and 6 show the curves *KG Error/bitrate* for the two different sequences, using the proposed coder for different GOF sizes. The results are compared to the Wavelet-based coder proposed in [14] (without clustering and without GC). Also, we compare the coding performances of the proposed coder with the PCA-based coder of [9], the CPCA- based coder of [11] and the Skinning-based coder of [17]. We choose different GOF sizes of 8 frames, 64 frames (DOLPHIN) and 384 frames (CHICKEN). The bitrate is given in bits per vertex per frame.
For both of the figures 5 and 6, we observe that the best results

are given by the Skinning-based coder of [17]. However, the three cases of the proposed coder provides best results than the PCA-based coder. Also, the proposed coder using the lifting scheme $(4, 2)$ is better than the CPCA-based coder of [11] (at high bitrates) and the Wavelet- based coder of [14].
Nevertheless, the proposed coder using scan-based processing for GOF of size 8 frames is interesting when huge sequences must be processed. Indeed, a large sequence requires a huge memory and moreover could imply an encoding delay as long as the sequence duration itself.

5. CONCLUSIONS AND FUTURE WORKS

In this paper, we have introduced motion-based clustering for mesh sequences with fixed connectivity. This compression scheme uses a scan-based wavelet transform geometrically compensated which requires low memory. Experimentally, we have shown that using motion estimation/compensation before the lifting steps, reduces the energy of the wavelet coefficients and improves the efficiency of the coder when small GOF size are used. First experimental results are promising. In our current works, we are improving the GC and the clustering approach.

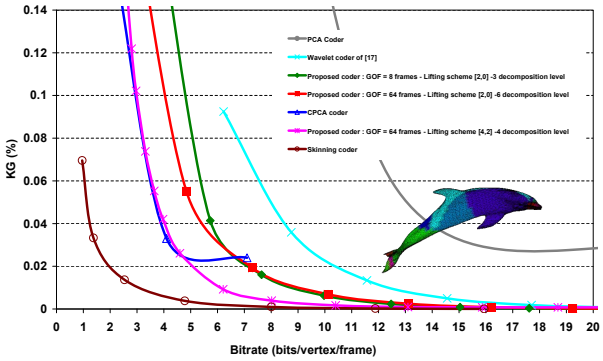

Fig. 5. *KG Error/bitrate* for DOLPHIN relative to different compression methods.

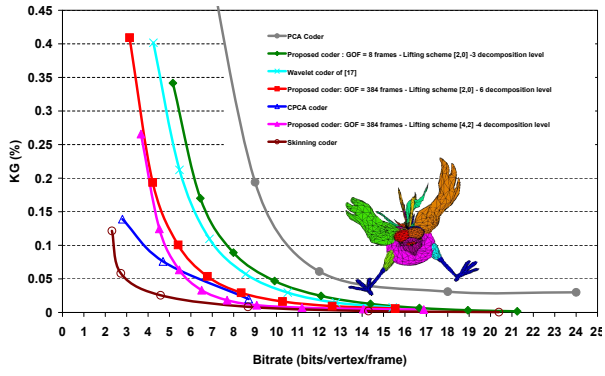

Fig. 6. *KG Error/bitrate* for CHICKEN relative to different compression methods.

6. REFERENCES

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

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

[3] S.Gupta, K.Sengupta, and A. Kassim, "Registration and partitioning-based compression of 3d dynamic data," *IEEE Trans. on Circuits Syst. Video Techn.*, vol. 13, no. 11, pp. 1144–1155, 2003.

[4] J.H. Yang, C.S. 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.

[5] 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.

[6] Karsten Muller, Aljoscha Smolic, Matthias Kautzner, and Thomas Wiegand, "Rate-distortion optimization in dynamic mesh compression," in *ICIP*, 2006.

[7] J. Zhang and C.B. Owen, "Hybrid coding for animated polygonal meshes: Combining delta and octree," in *International Conference on Information Technology: Coding and Computing*, 2005, pp. 68–73.

[8] Nikolce Stefanoski and Joern Ostermann, "Connectivity-guided predictive compression of dynamic 3d meshes," in *Proc. of ICIP*, october 2006.

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

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

[11] Mirko Sattler, Ralf Sarlette, and Reinhard Klein, "Simple and efficient compression of animation sequences," in *Proc. of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 2005, pp. 209–217, ACM Press.

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

[13] A.C. Lopes and M.N. Gamito, "Wavelet compression and transmission of deformable surfaces over networks," in *Proc. of the 10th Portuguese Computer Graphics Meeting*, 2001, pp. 107–114.

[14] F. Payan and M. Antonini, "Temporal wavelet-based geometry coder for 3d animations," *Computer & Graphics, In Press, Available Online*, 2006.

[15] I. Guskov and A. Khodakovsky, "Wavelet compression of parametrically coherent mesh sequences," in *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, August 2004.

[16] J-H. Yang, C-S. Kim, and S-U. Lee, "Progressive compression of 3D dynamic sequences," in *Proc. ICIP 2004*, Oct. 2004.

[17] K. Mamou, T. Zaharia, and F. Prêteux, "A skinning approach for dynamic 3d mesh compression," *Computer Animation and Virtual Worlds*, vol. 17, no. 3-4, pp. 337–346, July 2006.

[18] C. Touma and C. Gotsman, "Triangle mesh compression," *Graphics Interface'98*, pp. 26–34, 1998.