Temporal DCT-Based Compression of 3D Dynamic Meshes

KHALED MAMOU, TITUS ZAHARIA, and FRANCOISE PRETEUX

Groupe des Ecoles des Télécommunications
Institut National des Télécommunications / ARTEMIS Department
9, rue Charles Fourier 91011 Evry, France
http://www-artemis.int-evry.fr/

Abstract: - This paper introduces a new compression scheme for 3D dynamic meshes with constant connectivity and time-varying geometry. The proposed approach, referred to as Temporal-DCT encoder (TDCT), combines a piecewise affine prediction scheme with a temporal DCT-based compression of the prediction errors. Experiments show that TDCT achieves up to 70% and 54% lower compression distortions than the GV and MCGV approaches, while outperforming (47% to 95% lower distortions) RT, MPEG-4/AFX-IC, D3DMC and Dynapack techniques.

Key-Words: Dynamic 3D mesh compression, DCT, piecewise affine prediction, motion-based segmentation.

1. Introduction

Extensively used in physical simulations, virtual and augmented reality systems, CGI films and video games, 3D animations are becoming omnipresent in today's industrial and general public applications.

Such a dynamic content is produced by using a wide range of techniques depending on the considered application. Scientists and engineers use hardware-based motion acquisition [18] approaches and physically-based simulations. Multimedia content creators are more accustomed to morphing and skinning methods. Such techniques are fully supported by the majority of the existing 3D authoring environments (3DS MAX, Maya). Moreover, the MPEG-4/AFX (Animation Framework eXtension) standard [3] has adopted such a skinning approach, so-called bone-based animation (BBA).

However, for transmission and visualization purposes the 3D animation industry adopts a key-frame representation, which offers the advantages of:

- generality (ability to represent 3D animations independently of the underlying 3D modelling technique used for content creation),
- interoperability (multi-platform content rendering), and
- ability to ensure content protection [11].

In this case, the dynamic mesh is stored as a sequence of consecutive 3D meshes $(M_i)_{i\in\{0,\ldots,F-1\}}$ with constant connectivity and time-varying geometry, representing key-frames. F stands for the number of key-frames. The intermediate frames are derived by applying interpolation procedures.

However, the key-frame representation leads to a huge amount of data, since even short sequences of a few minutes require several thousands of meshes. In order to ensure the efficient storage, transmission and visualization of a such representation, compression techniques need to be elaborated.

The emerging research field of dynamic 3D mesh compression has gained rapidly the interest of the scientific community as testifies the important number of recently developed works reported in the literature (see [11] for an overview).

The clustering-based approaches [13], [5] express the animation in terms of rigid transforms (RT). In [14], the so-called D3DMC (Dynamic 3D Mesh Coder) technique represents the motion field of each frame as an octree structure and a set of associated motion vectors. The major limitations of the clustering approaches are related to the segmentation procedure involved, which is computationally complex and may cause disgraceful discontinuities at low bit rates.

The Interpolation Compression (IC) scheme [8], recently adopted by the MPEG-4/AFX standard, and the Dynapack approach [7] exploit local spatio-temporal predictors to encode the dynamic mesh geometry. Their simplicity and low computational cost make them well-suited for real-time decoding applications. However, these approaches do not support more advanced functionalities such as progressive transmission and scalable rendering.

The compression schemes introduced in [1] and [10] compress 3D animations by applying a principal component analysis (PCA) of the mesh deformation field. Such PCA-based approaches are specifically adapted for long repetitive animation sequences of small meshes with a number of frames much greater than the number of mesh vertices. However, they suffer from a high computational complexity, since a singular value decomposition is required.

The Geometry Video (GV) [4] compression scheme exploits a mesh cut and a stretch minimizing parameterization [16] over a 2D square domain. Here, the initial mesh connectivity is completely discarded and replaced by a regular one obtained by uniformly sampling the parametric domain. The so-obtained sequence of *geometry images* is then compressed by using traditional video encoding techniques. A global motion compensation procedure is first applied. Each frame is predicted from the previous one, by using a global affine motion model. The resulting prediction errors are then compressed by using a wavelet-based encoding scheme. The main drawbacks of the GV approach are related to the remeshing procedure involved, which may lead to a loss of surface details, and to tangent plane discontinuities at the level of the cut.

To overcome the above mentioned limitations, Mamou and *al.* recently introduced the *Multi-Chart Geometry Video (MCGV)* [12] representation which extends the GV approach. Here, the initial mesh connectivity is preserved, which makes it possible to avoid the remeshing related problems. Authors also improve the motion compensation stage by applying a piecewise affine predictor instead of a global one. In addition, parameterization distortions are limited by exploiting a low-distortion atlas of parameterization [19]. Finally, the predictions errors are stored as 2D Multi-Chart Geometry Images (MCGIMs) [17] and compressed using DCT-based image or video encoders (*e.g.* JPEG, MPEG-4).

This paper proposes a novel compression scheme for 3D dynamic meshes, referred to as Temporal-DCT encoder (TDCT). As in the case of MCGV, the TDCT compression scheme adopts a piecewise affine predictor for the motion compensation stage. However, the residual prediction errors are here encoded by applying a temporal-DCT-based compression. With respect to MCGV, the TDCT approach offers the following advantages:

- It avoids the mesh parameterization and thus the inherent parameterization distortions.
- It better captures the temporal correlations of the animation signal by globally processing the trajectory of each vertex over time.

The paper is structured as follows. Section 2 describes the proposed TDCT compression scheme and details its main stages: motion segmentation, piecewise affine prediction and temporal-DCT compression. Section 3 analyzes how the proposed approach responds to more advanced functionalities, such-as progressive transmission and scalable rendering. TDCT's performances are objectively evaluated and discussed in Section 4. Finally, Section 5 concludes the paper and opens perspectives of future work.

2. Temporal DCT-based Compression

Figure 1 illustrates the TDCT encoding scheme. As in [12], the mesh vertices are partitioned into groups exhibiting the same affine motion with respect to the first frame. This partition is then exploited in order to compute a piecewise affine predictor which minimizes, in the mean squares error sense, the prediction errors. Finally, the so-obtained prediction residuals are compressed by applying a temporal DCT encoding scheme.

The first mesh of the sequence is statically encoded and transmitted as side information. Any static compression approach can be used for encoding the first mesh. In this work, we have considered the classic Touma & Gotsman (TG) [15] monoresolution technique and the multiresolution Progressive Mesh [6] approach.

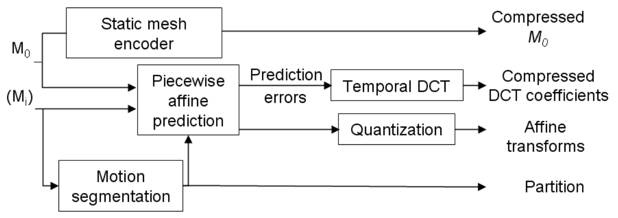


Figure 1: The TDCT compression scheme.

Let us now detail the main blocks of the TDCT encoding scheme, starting with the motion-based segmentation stage.

2.1 Motion-based segmentation

The objective of the segmentation stage is to obtain a partition $\pi = (\pi_k)_{k\in\{1,...,K\}}$ of the mesh vertices into K clusters whose motion can be accurately described by 3D affine transforms. The proposed approach, detailed here-below, takes into account the motion of the mesh vertices over the whole set of frames in the sequence.

First, an affine transform A_i^v describing the affine motion of the local neighbourhood of each vertex $v \in \{1,...,V\}$ at frame $i \in \{0,...,F-1\}$ with respect to first frame is computed as follows:

$$A_i^v = \arg\min_A \left(\sum_{v\in v^*} \|A\chi_0^v - \chi_i^v\|^2 \right), \qquad (1)$$

where A is 4×4 matrix representing an affine transform, v^* is the third order neighbourhood of the vertex v (*i.e.*, vertices connected to v by a path consisting of at most three edges), and χ_i^v is a 4D vector representing the homogeneous coordinates of the vertex v at frame i.

Then, the set of $(A_i^v)_{i\in\{0,...,F-1\}}$ is stored as a single vector $\alpha^v \in \mathbb{R}^{12\times F}$ (since an affine transform is completely defined by 12 real-valued coefficients). Finally, the partition π is obtained by applying the k-means clustering algorithm [9] to the vector set $(\alpha^v)_{v\in\{1,...,V\}}$.

2.2 Piecewise affine prediction

Once the partition of the dynamic mesh determined, for each frame i, the motion field is modeled as a set of K affine transforms denoted by $(H_i^k)_{k\in\{1,...,K\}}$. With the same notations as above, the affine transform H_i^k associated with patch k at frame i is defined as:

$$H_i^k = \arg\min_A \left(\sum_{v\in\pi_k} \|A\chi_0^v - \chi_i^v\|^2 \right). \qquad (2)$$

The set $(H_i^k)_{k\in\{1,...,K\}}$, together with the partition π provide a piecewise affine predictor of the frame i from the frame 0 defined as :

$$\forall\, v \in \{1,...,V\}, \quad \widehat{\chi}_i^v = H_i^{k(v)}\chi_0^v, \qquad (3)$$

where $k(v)$ denotes the cluster to which the vertex v belongs.

The prediction errors $e_i^v = (e_i^{v,x}, e_i^{v,y}, e_i^{v,z}, 0)^t$ are defined as:

$$\forall\, v \in \{1,...,V\}, \quad e_i^v = \chi_i^v - \widehat{\chi}_i^v. \qquad (4)$$

The residuals $(e_i^{v,x}, e_i^{v,y}, e_i^{v,z})$ are finally compressed by using a temporal DCT-based compression, as described in the next section.

2.3 Temporal DCT-based compression

For each vertex v of the dynamic mesh, we consider the associated prediction errors $(e_i^{v,x})_i$, $(e_i^{v,y})_i$, and $(e_i^{v,z})_i$ ($i \in \{1,,F\}$) as three temporal sequences. The spectra $(s_i^{v,x})_i$, $(s_i^{v,y})_i$, and $(s_i^{v,z})_i$ of each error sequence is then determined by applying a monodimensional DCT over the time domain.

This procedure is applied to all the mesh vertices. The obtained spectral coefficients are then multiplexed into a single vector S of dimension $3\times V\times F$ defined as follows:

$$S = \coprod_{i\in\{1,...,F\}} \coprod_{v\in\{0,...,V-1\}} (s_i^{v,x}, s_i^{v,y}, s_i^{v,z})^t, \qquad (5)$$

where $\coprod$ represents the concatenation operator.

Since the DCT transform concentrates the energy of a smooth signal within the low frequency band, the majority of the DCT coefficients will have a small magnitude, which makes them well-suited for arithmetic encoding and thus ensures high compression rates.

Let us analyse now how the proposed approach responds to more advanced functionalities such as progressive transmission and scalable rendering.

3 Progressive transmission and scalable rendering

The progressive transmission functionality concerns the bitstream adaptation to different, fixed or mobile communication networks with various bandwidths. Here, the decoder can start displaying a coarse approximation of the animation sequence when some baseline, minimal information is received. A refinement process is then applied to this coarse representation in order to gradually improve the visual quality of the decoded animation sequence.

The concatenation process, described in Equation (5), makes possible the progressive transmission of the DCT coefficients from low to high frequencies, and thus the reconstruction of an approximated animation sequence at any stage of the decoding process.

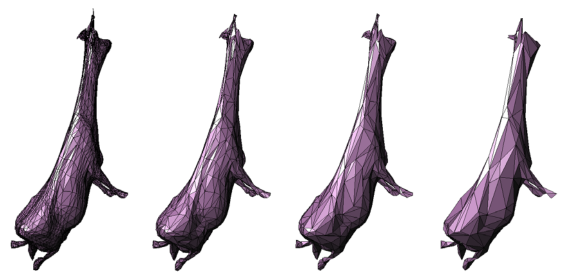
The scalable rendering functionality concerns the bitstream adaptation to terminals (*e.g.*, desktop computers, laptops, PDAs, mobilephones...) of various complexities with different memory and computational capacities, under real-time visualization constraints.

In order to support the scalable rendering functionality, the TDCT representation should be associated with a progressive encoding of the first mesh. In this work, we have adopted the Progressive Mesh (PM) approach introduced in [6].

The principle consist of simplifying the mesh topology by applying a sequence of half-edge collapse operations, denoted by $(hecol_j)_j$. Each half-edge collapse operation merges two mesh vertices into a single one. The encoder decimates the mesh vertices until a coarse base mesh is reached. Conversely, starting from the base mesh, the decoder can progressively reconstruct the mesh connectivity at increasing finer Levels Of Detail (LOD), by successively applying a sequence of vertex split operations, denoted by $(vsplit_j)_j$. A vertex split represents the inverse of a half-edge collapse operation and consists of splitting a vertex into two different ones.

The TDCT approach naturally exploits the PM hierarchy for recovering different LODs of the dynamic mesh. The connectivity information, corresponding to the targeted LOD is decoded only once for the entire sequence. Then, the associated geometry information is recovered for each frame. The bitstream includes the set of DCT spectra of all vertices of the initial mesh. The decoder selects the subset of spectra corresponding to the mesh vertices at the considered LOD. The inverse DCT is then applied solely to the selected spectra for recovering the geometry signal.

Let us note that the above-described procedure can be applied due to the half-edge collapse decimation operator considered, which ensures that the set of mesh vertices at a given LOD is included within the set of mesh vertices of any finer LOD. In addition, such an approach leads to savings in computational time since the inverse DCT, which is determinant for the decoding complexity, is applied only to a reduced set of spectra. Figure 2 illustrates the frame 37 of the "Cow" animation sequence at different LODs.



(a) V = 2904 (b) V = 1002 (c) V = 502 (d) V = 252

Figure 2: Frame 37 of the "Cow" sequence at different LODs.

4 Experimental results

4.1 Evalution corpus

In order to be able to perform objective comparisons, we have considered a data set animation sequences, used by the majority of the works reported in the literature, and so-called "Dance", "Chicken" and "Cow". Table 1 summarizes their properties, expressed in terms of numbers of vertices (V), frames (F), and connected components (CC).

Animation sequence	V	F	CC
"Cow"	2904	204	1
"Chicken"	3030	400	41
"Dance"	7061	201	1

Table 1: Properties of the animated mesh sequences considered in the evaluation.

4.2 Objective evaluation criteria

In order to compare compression performances for mesh sequences with various number of frames and vertices, compression rates are expressed in terms of bits per frame per vertex (bpfv).

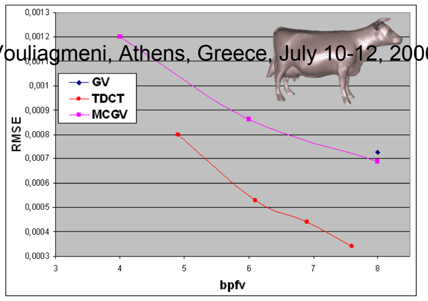
The compression distortions are measured by using the RMSE error [2] between initial and reconstructed (decoded) meshes. The RMSE error between two mesh sequences is defined as the mean value of the frame to frame RMSE, calculated over all the frames in the sequence.

4.3 Compression results

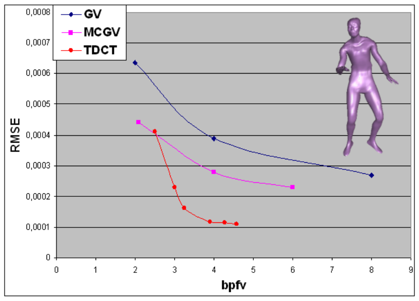
In our experiments the number of clusters K was set to 20 for all the sequences. The real coefficients describing the affine transforms were quantized on 17 bits. The first frame of the sequence was encoded by using the Touma and Gotsman approach [15], with the geometry being quantized on 12 bits. The predictions residuals were quantized on 8 bits.

Figure 3 presents a comparison between the proposed TDCT approach, the MCGV compression scheme and the GV encoder.

For the "Cow" sequence, TDCT shows 50% lower compression distortions than GV at 8 bpfv and 22% lower distortions than MCGV at 5 bpfv. For the "Dance" sequence, TDCT achieves up to 70% and 54% lower distortions than GV and MCGV respectively. The low compression performances of MCGV and GV when compared to the TDCT approach, can be explained by: the parameterization distortions inherent to these encoders, and their non-optimal handling of the temporal correlations of these sequences.



(a) "Cow"



(b) "Dance"

Figure 3: TDCT compression scheme *versus* GV and MCGV encoders.

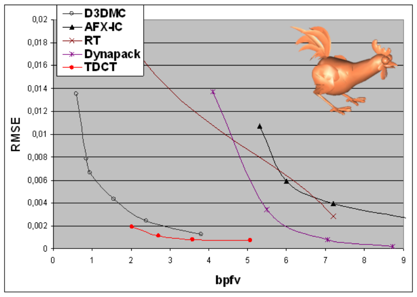


Figure 4: TDCT outperforms AFX-IC, D3DMC, RT and Dynapack for "Chicken" sequence.

Figure 4 compares the performances of TDCT to those of D3DMC, AFX-IC, RT, and Dynapack for the "Chicken" sequence. Here, D3DMC and TDCT outperform all the other encoders for low bitrates (less than 4 bpfv). At 4.1 bpfv, TDCT compression scheme provides 95% lower compression distortions than Dynapack and AFX-IC. At 2.2 bpfv, TDCT achieves up to 90%, 47% lower compression distortions than RT and D3DMC respectively. This can be explained by the non-optimality of the rigid and the octree decomposition of the motion considered by D3DMC and RT.

Experiments establish that the proposed TDCT compression scheme provides a compact representation of 3D animations which outperforms the other state-of-the-art approaches. In particular, the proposed approach proves to be highly efficient for articulated dynamic meshes such as "Dance" and "Chicken", where the piecewise affine prediction stage is fully exploited.

5. Summary and Conclusions

This paper introduced a new compression scheme for animated 3D meshes, which combines a piecewise affine motion compensation strategy with temporal DCT-based encoding of the prediction residuals.

Experimental results show that the proposed approach outperforms GV, MCGV, D3DMC, RT, Dynapack, and AFX-IC dynamic mesh compression techniques. The TDCT approach reveals to be particularly efficient when applied to articulated dynamic meshes.

Future work, will address the extension of this encoder to dynamic meshes with properties such as mesh normals, texture and colours.

References

[1] M. Alexa, W. Müller, "Representing animations by principal components", *Proc. Computer Graphics Forum*, Vol. 19, pp. 411-8, 2000.

[2] N. Aspert, D. Santa-Cruz, T. Ebrahimi, "MESH: Measuring Errors between Surfaces using the Hausdorff distance", *Proc. IEEE Internat. Conference in Multimedia and Expo (ICME)*, Vol. 1, pp. 705-708, 2002.

[3] M. Bourges-Sevenier, E.S. Jang, "An introduction to the MPEG-4 Animation Framework extension", *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 14, pp. 928-936, 2004.

[4] H. M. Briceno, P. V. Sander, L. McMillan, S. Gortler, H. Hoppe, "Geometry videos: a new representation for 3D animations", *Proc. ACM Siggraph Symposium on Computer Animation*, pp. 136-146, 2003.

[5] G. Collins, A. Hilton, "A rigid transform basis for animation compression and level of detail", *Proc. ACM Symposium on Interactive 3D Graphics*, pp. 21-29, 2005.

[6] H. Hoppe, "Progressive meshes", *Proc. ACM SIGGRAPH Conference*, pp. 99-108, 1996.

[7] L. Ibarria, J. Rossignac, "Dynapack: space-time compression of the 3D animations of triangle meshes with fixed connectivity", *Proc. ACM Siggraph Symposium on Computer Animation*, pp. 126-13, 2003.

[8] E. S. Jang, J.D.K. Kim, S. Y. Jung, M.J. Han, S. O. Woo, S. J. Lee, "Interpolator Data Compression for MPEG-4 Animation", *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 14, pp. 989-1008, 2004.

[9] T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 24, pp. 881-892, 2002.

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

[11] K. Mamou, T. Zaharia, F. Prêteux, "A preliminary evaluation of 3D mesh animation coding techniques", *Proc. SPIE Conference on Mathematical Methods in Pattern and Image Analysis,* San Diego, CA, Vol. 5916, pp. 44-55, 2005.

[12] K. Mamou, T. Zaharia, F. Prêteux, "Multi-chart geometry video: a compact representation for 3D animations", *Proc. Internat. Symposium on 3D Data Processing, Visualization and Transmission (3DPVT), 2006.*

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

[14] K. Müller, A. Smolic, M. Kautzner, P. Eisert, T. Wiegand, "Predictive compression of dynamic 3D meshes", *Proc. Internat. Conference on Image Processing*, pp. 621-624, 2005.

[15] C. Touma, C. Gotsman, "Triangle mesh compression", *Proc. Graphics Interface,* pp.26-34, 1998.

[16] P. Sander, P. Gortler, J. Snyder, H. Hoppe, "Signal-Specialized Parameterization", *Microsoft Research technical report*, MSR-TR-2002-27, 2002.

[17] P. Sander, Z. Wood, S. Gortler, J. Snyder, H. Hoppe, "Multi-chart geometry images", *Eurographics Symposium on Geometry Processing*, pp. 146-155, 2003.

[18] P. Sand, L. McMillan, and J. Popovic, "Continuous Capture of Skin Deformation", *ACM Trans. on Graphics*, 22(3), pp. 578-586, 2003.

[19] K. Zhou, J. Snyder, B. Guo, H. Y. Shum, "Iso-charts: Stretch-driven Mesh Parameterization Using Spectral Analysis", *Proc. ACM Symposium on Geometry Processing*, pp. 45-54, 2004.