SCALABLE LINEAR PREDICTIVE CODING OF TIME-CONSISTENT 3D MESH SEQUENCES

Nikolče Stefanoski, Xiaoliang Liu, Patrick Klie, Jörn Ostermann

Leibniz Universität Hannover, Appelstr. 9A, 30167 Hannover, Germany

ABSTRACT

We present a linear predictive compression approach for time-consistent 3D mesh sequences supporting and exploiting scalability. The algorithm decomposes each frame of a mesh sequence in layers employing patch-based mesh simplification techniques. This layered decomposition is consistent in time. Following the predictive coding paradigm, local temporal and spatial dependencies between layers and frames are exploited for compression. Prediction is performed vertex-wise from coarse to fine layers exploiting the motion of already encoded 1-ring neighbor vertices for prediction of the current vertex location. It is shown that a predictive exploitation of the proposed layered configuration of vertices can improve the compression performance upon other state-of-the-art approaches by up to 16% in domains relevant for applications.

Index Terms— Animation compression, dynamic 3D mesh coding, scalability, linear predictive coding, time-consistent mesh sequence.

1. INTRODUCTION

Multimedia hardware is getting evermore powerful and affordable. This development enables a permanent improvement of existing applications or even development of new applications based on time-varying 3D content, like 3D television, immersive telesurgery, or immersive computer games. Efficient compression of time-varying 3D content gets crucial importance in this context.

Due to an increasingly broadening range of access networks, like the Internet or local area networks, mobile networks, etc., the bit rate of compressed time-varying 3D content has to be adapted to network transfer rates and end-user devices. To meet this requirement we developped a scalable compression scheme. This technique enables to encode 3D content once only, while decoding can be performed on different devices with a quality adapted to the capacity of the network and the end-user device. This is achieved by creating structured bit streams that allow layer-wise decoding and successive reconstruction of 3D content with increasing quality.

Time-varying 3D content is usually represented by a sequence of 3D meshes called frames. Frames consist of two types of data: connectivity and 3D locations. In this paper we assume that we are dealing with frames that have constant connectivity throughout time, i.e. time-consistent 3D mesh sequences consisting of F frames and V vertices per frame. Each vertex v in frame f is associated with a location in 3D space denoted by p_v^f for $v \in \mathcal{V} := \{1, \dots, V\}$ and $f \in \mathcal{F} := \{1, \dots, F\}$. Furthermore, the set of all vertices $\mathcal{V}$ is decomposed into L layers, i.e. disjoint sets of vertices $\mathcal{V}_l$ for $1 \leq l \leq L$ with the property $\cup_{1 \leq l \leq L} \mathcal{V}_l = \mathcal{V}$ (see Fig. 1). Since connectivity does not change throughout the entire mesh sequence, it has to be encoded only once. We assume that connectivity is compressed in the beginning of the encoding process by one of the nearly optimal connectivity compression techniques [1, 2]. While connectivity does not vary over time, vertices change their location. Therefore, the major part of an encoded mesh sequence generally consists of vertex locations. For this reason, in this paper we concentrate on compression of vertex locations of time-consistent 3D mesh sequences.

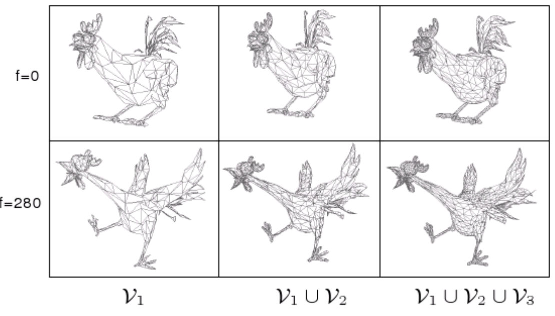

f=0
f=280
$\mathcal{V}_1$
$\mathcal{V}_1 \cup \mathcal{V}_2$
$\mathcal{V}_1 \cup \mathcal{V}_2 \cup \mathcal{V}_3$

Fig. 1. Illustration of a decomposition of two frames into three layers.

The rest of the paper is organized as follows. Section 2 gives an overview about recent developments in the area of compression of mesh sequences. An overview of the proposed scalable coder is given in Section 3, describing in detail the decomposition in layers, prediction, and entropy coding. In Section 4, compression results are evaluated and discussed. Finally we end with a conclusion in Section 5.

2. RELATED WORK

Several approaches for compression of dynamic 3D meshes have been presented recently. Karni and Gotsman [3] and Sattler et al. [4] transform dynamic meshes using principal component analysis (PCA) to reduce the amount of coded data. Guskov et al. [5] and Payan et al. [6] propose wavelet-based approaches for compression. While Guskov et al. apply the wavelet transform for each frame separately exploiting later the temporal coherence between wavelet coefficients, Payan et al. apply the wavelet transform in temporal direction on vertex trajectories and use a model-based entropy coder for entropy compression. A method for error resilient streaming of dynamic 3D meshes that minimizes the perceptual effect of data loss was introduced by Varakliotis et al. [7]. Recently Müller et al. [8] presented a rate-distortion optimized compression scheme which exploits the coherence between motion vectors by combining octree based motion vector clustering with an optimized selection of a compression mode for each cluster. Yang et al. [9] and Ibarria and Rossignac [10] presented vertex traversal based compression algorithms

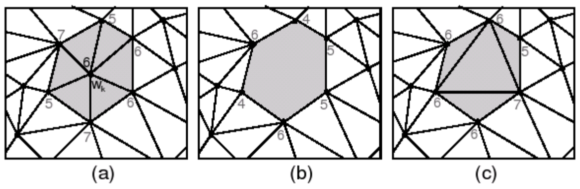


Fig. 2. Illustration of patch-based vertex removal: (a) a degree-6 patch with middle vertex w_k, (b) patch after removal of w_k, (c) re-triangulated patch. Numbers represent valences of corresponding vertices.

using linear predictors. In the first paper a parallelogram-like prediction rule is applied, while in the second paper motion vector averaging is employed to exploit local inter and intra frame coherence between vertex locations. Recently, Stefanoski and Ostermann [11] presented a non-linear predictor for vertex traversal based compression improving the prediction accuracy at high bit-rates compared to linear predictors. Mamou at al. [12] introduced a novel technique for compression of mesh animations based on a skinning animation technique. They employ vertex clustering and propose a weighted affine transform in order to exploit inter and intra cluster dependencies for prediction. The algorithm presented in this paper is related to the predictive approaches [11, 10, 9] introducing a novel configuration of vertices for scalable compression.

3. SCALABLE COMPRESSION

The coder presented in this paper follows the predictive coding paradigm and supports scalability. In this section we describe a spatial scalable linear predictive coder (SSLPC). A simple extension to spatio-temporal scalability is described in Section 4. The proposed coder encodes all frames in order $1, \ldots, f-1, f, \ldots, F$ encoding all vertex locations p_v^{f-1} with $v \in \mathcal{V}$ before encoding all p_v^f with $v \in \mathcal{V}$. Vertex locations p_v^f of an arbitrary frame f are encoded layer-wise in order

$$v \in \mathcal{V}_1, \ldots, v \in \mathcal{V}_l, \ldots, v \in \mathcal{V}_L,$$

starting with all vertex locations of the base layer $\mathcal{V}_1$ and ending with all vertex locations of the highest layer $\mathcal{V}_L$. Fig. 1 illustrates the organization in layers.

3.1. Layer Design

Layers, i.e. disjoint sets $\mathcal{V}_l$, are defined by employing mesh simplification techniques. A deterministic mesh simplification algorithm is applied exploiting only mesh connectivity. Hence, no additional side information is needed for describing layers $\mathcal{V}_l$, since connectivity in known at the decoder side in the beginning of the compression process. The basic simplification operation used in this algorithm is based on patches. A degree-d patch is set of triangles incident to a vertex of valence d. In Fig. 2(a) a gray shaded degree-6 patch is presented. The employed simplification algorithm consists of two major steps: *Patch Decomposition* and *Simplification*.

3.1.1. Decomposition into Patches

Mesh connectivity is first decomposed into patches using a *deterministic* patch-based region-grow traversal algorithm conquering only patches of degree ≤ 6. Decomposition starts with an arbitrarily selected initial seed patch of degree ≤ 6 by marking it as not conquered and enqueueing it into a FIFO. In the traversal loop, the first patch is dequeued from the FIFO. If this patch is already marked as conquered it is skipped an the loop starts from the beginning. If it is not yet conquered, it is marked as conquered, its middle vertex w_k is saved, and all neighboring patches of degree ≤ 6 are enqueued into the FIFO. Here neighboring patches are patches which have only one common edge with the current patch. This procedure is iterated until the FIFO is empty. As output we obtain a series of K patches, described by their middle vertices $w_1 \ldots, w_k, \ldots, w_K$. Subsequent simplification is performed only to these patches.

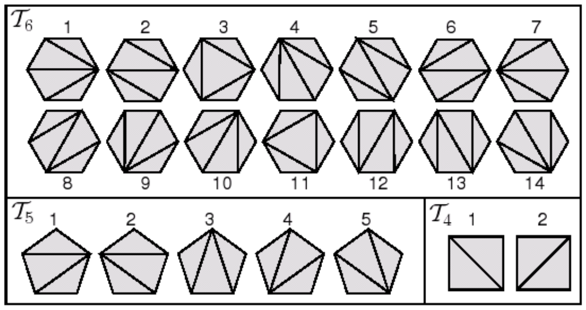


Fig. 3. Triangulations $\mathcal{T}_d$ for $d = 4, 5, 6$.

3.1.2. Patch Based Simplification

All patches obtained during patch decomposition are traversed again in the order they were conquered. The middle vertex w_k of each patch is removed and the remaining polygon is re-triangulated (see Fig. 2). The number of different triangulations T_d for a degree-d patch depends on d, i.e. there are $T_3 = 1, T_4 = 2, T_5 = 5, T_6 = 14$ triangulations [13]. We define the set of all triangulations of a degree-d patch as $\mathcal{T}_d := \{1, \ldots, T_d\}$ (see Fig. 3). Let $\mathcal{N}(w_k)$ denote the set of 1-ring vertices of the middle patch vertex w_k.

In order to select one triangulation $t \in \mathcal{T}_d$ for a degree-d patch after removing its middle vertex w_k we apply a measure $\mathsf{Dev}(w_k, t)$. It measures the average absolute deviation of all valences of vertices in $\mathcal{N}(w_k)$ from the valence 6:

$$\mathsf{Dev}(w_k, t) := \frac{1}{|\mathcal{N}(w_k)|} \sum_{v' \in \mathcal{N}(w_k)} |\mathsf{val}(v', t) - 6|.$$

Here $\mathsf{val}(v', t)$ denotes the valence of vertex v' after removing the middle vertex w_k and re-triangulating the remaining polygon using triangulation t. Note that different triangulations can lead to different deviations, since valences of vertices v' change depending on the triangulation t. We select that triangulation t for re-triangulating a patch which leads to the smallest deviation $\mathsf{Dev}(w_k, t)$. Overall, this kind of selection reduces the absolute deviation of valences from valence 6. Hence, it prevents the creation of vertices with large valences, which usually lead to long narrow triangles. Furthermore, large vertex valences would also lead to many valence 3 vertices, which can not be predicted accuratelly later on.

3.1.3. From Simplification to Layers

The set of all vertices $\mathcal{V}$ is decomposed in L disjoint subsets $\mathcal{V}_l$ by recursively applying the procedures described in Sections 3.1.1

and 3.1.2 to the simplified connectivity. First, patch decomposition and simplification is applied to the connectivity consisting of vertices in $\mathcal{V}$. Thus, a set of K_L vertices $\mathcal{V}_L = \{w_1^L, \dots, w_{K_L}^L\}$ is removed and a simplified connectivity consisting of vertices in $\mathcal{V} \setminus \mathcal{V}_L$ remains. Recursively applying this procedure to the simplified connectivity we obtain for each $l = L-1, \dots, 2$ a set of vertices $\mathcal{V}_l = \{w_1^l, \dots, w_{K_l}^l\}$ and a simplified connectivity consisting of vertices $\mathcal{V} \setminus \cup_{k=l}^{L} \mathcal{V}_k$. At last we define the base layer as the remaining set of vertices $\mathcal{V}_1 = \mathcal{V} \setminus \cup_{k=2}^{L} \mathcal{V}_k$.

Patch based simplification guarantees that the neighbors of each vertex are located one layer below, i.e. for each vertex $v \in \mathcal{V}_l$ for $l = 2, \dots, L$ we have $\mathcal{N}(v) \subset \mathcal{V}_{l-1}$. In the following it will be shown that a configuration of interleaving vertices like this allows a robust exploitation of inter layer dependencies of vertex locations p_v^f in space and time.

3.2. Prediction

Vertex locations of meshes representing real objects show correlations. In order to exploit this coherence in spatial and temporal direction predictive coding is applied. Vertex locations p_v^f which are part of the base layer, i.e. $v \in \mathcal{V}_1$, are encoded for $f = 1, \dots, F$ using a single resolution (flat) predictive coder presented in [11]. After having encoded all vertex locations of the base layer of a frame f, vertex locations of higher layers of the same frame are encoded subsequently. Vertex locations p_v^f part of a layer $l > 1$ are predicted using already encoded vertex locations. This prediction of p_v^f is performed by a linear predictor $\hat{p}_v^f$ based on motion vector averaging (see Fig. 4) [11, 9]:

$$\hat{p}_v^f = p_v^{f-1} + \frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} (p_u^f - p_u^{f-1}).$$

For $f = 1$ we define $p_v^{f-1} = p_u^{f-1} = (0,0,0)^T$. This allows a prediction of vertex locations of layers $l > 1$ also for frame $f = 1$ using predictor $\hat{p}_v^f$. Thus, encoded vertex locations of frame $f-1$, which are part of layers l and $l-1$, and of frame f, which are part of layer $l-1$, are used for prediction of p_v^f exploiting inter frame and inter layer dependencies.

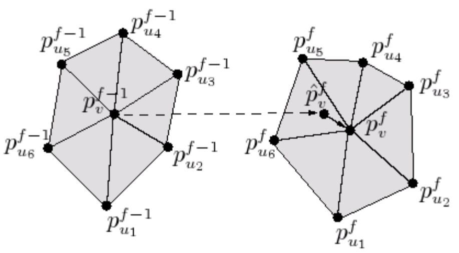


Fig. 4. Prediction of location p_v^f based on p_v^{f-1} and locations of neighboring vertices $\mathcal{N}(v) = \{u_1, \dots, u_6\}$ in frames $f-1$ and f.

3.3. Quantization and Entropy Coding

Prediction errors $\delta_v^f = p_v^f - \hat{p}_v^f$ are uniformly quantized and entropy coded in order to exploit statistical dependencies. We apply an adaptive order-0 arithmetic coder combined with Golomb codes [11] for entropy coding. Separate entropy coders are employed for encoding quantized predictions errors of each layer l, adapting arithmetic coders separately to statistical distributions of residuals in each layer.

Table 1. Number of vertices K_l per layer l and percentage of increased spatial resolution per new layer.

	Chicken		Cow	
l	K_l	$K_l / \sum_{k=1}^{l-1} K_k$	K_l	$K_l / \sum_{k=1}^{l-1} K_k$
1	318	∞	593	∞
2	103	32.4%	225	37.9%
3	166	39.4%	290	35.5%
4	221	37.6%	423	38.2%
5	299	37.0%	554	36.2%
6	442	39.9%	819	39.3%
7	600	38.7%	-	-
8	881	41.0%	-	-
Σ	3030		2904	

4. EVALUATION AND RESULTS

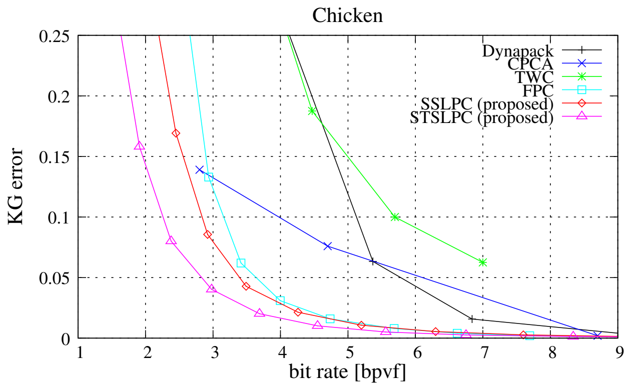
For experimental evaluation, we used the mesh sequences *Chicken* consisting of 400 frames and 3030 vertices per frame and *Cow* consisting of 204 frames and 2904 vertices per frame. Evaluation with other mesh sequences led to comparable results.

As result of the layered representation of connectivity we obtain time-consistent mesh sequences in different spatial resolutions (Fig. 1). The connectivities of the sequences *Chicken* and *Cow* were decomposed in 8 and 6 layers respectively (Table 1). Each time a new layer is added the number of vertices is increased by about 38%, i.e. spatial resolution increases with a nearly constant factor.

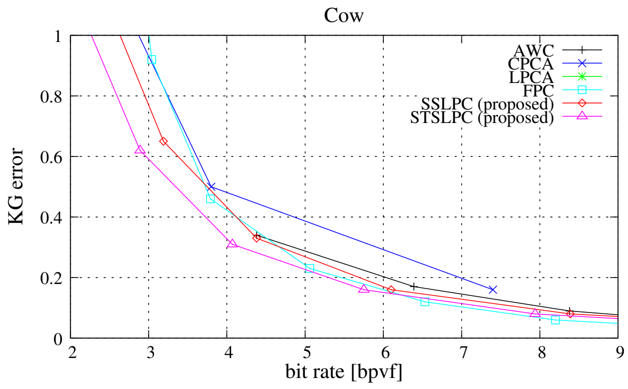
The quality of encoded vertex locations is measured relative to corresponding original vertex locations using a normalized vertex-wise L_2 norm [3]. We denote it here as KG error. Bit rate is measured in bits per vertex and frame (bpvf). In order to allow a comparison with other approaches using this measure, all vertex locations, i.e. all L layers per frame, are encoded.

Before entropy coding, prediction errors are quantized uniformly in each spatial direction using a predefined quantization bin size Δ [11]. Operational rate-distortion curves shown in Fig. 5 were produced by varying Δ. Besides the already in Section 3 introduced coder SSLPC, which supports only one-directional prediction based on a previous frame $f-1$, we evaluated also a spatio-temporal coder (STSLPC) by employing bi-directional prediction. It is realized by first encoding an odd frame f predictively based on an encoded odd frame $f-2$ and then encoding the even frame $f-1$ using bi-directional prediction. It is performed by calculating the average of two one-directional predictions, one based on frame f and the other one based on frame $f-2$.

In Fig. 5 the proposed coders SSLPC and STSLPC were evaluated against state-of-the-art compression algorithms. Due to the usage of different error measures we were not able to compare against all algorithms mentioned in Section 2. We compared against the flat predictive coder Dynapack [10], the flat predictive coder (FPC) of Stefanoski and Ostermann [11], the wavelet based approaches of Payan et al. (TWC) [6] and Guskov et al. (AWC) [5], and the PCA-based approach of Sattler et al. (CPCA) [4]. Both proposed coders outperform all other approaches except of FPC. FPC shows higher gains in the domain of very high bit rates because of its better exploitation of non-linear dependencies between vertex locations and a frame-wise adaption of the predictor. Note that an error of 0.02 for *Chicken* and 0.15 for *Cow* can be regarded as lossless with regard to visual quality. In this domain STSLPC and FPC show best per-



(a) Sequence *Chicken* encoded using l =8 layers.



(b) Sequence *Cow* encoded using l =6 layers.

Fig. 5. Evaluation results.

formance. Thus, STSLPC provides the feature of scalability without any overhead in bit rate. In domains of higher errors both, SSLPC and STSLPC, show significant gains. For instance for the *Chicken* sequence in the area of errors above 0.08, SSLPC achieves gains of over 10%, while STSLPC achieves even higher gains of over 16%. This gains are due to the configuration of interleaving vertices between neigboring layers. This kind of configuaration enables a prediction based on interpolation, which is robust against quantization noise. A bi-directional prediction increases the robustness even more. Obviously in domains relevant for applications scalability can lead to additional gains.

5. CONCLUSION

In this paper, we presented a scalable coder for time-consistent 3D mesh sequences. Layers were defined by employing patch-based mesh simplification techniques and a robust interpolation based prediction was applied in order to exploit spatio-temporal dependencies between layers and frames. We experimentally showed that a scalable configuration of vertices can improve the exploitation of spatio-temporal dependencies. The proposed algorithm outperforms state-of-the-art approaches in domains relevant for applications. Furthermore, it can be used for real-time compression due to its low computational cost (linear run-time in the number of vertices).

6. ACKNOWLEDGMENT

This work is partly supported by the EC within FP6 under Grant 511568 with the acronym 3DTV. We would like to thank Matthias Müller from ETH Zürich for providing the *Cow* sequence. The *Chicken* character was created by Andrew Glassner, Tom McClure, Scott Benza, and Mark Van Langeveld. This short sequence of connectivity and vertex position data is distributed solely for the purpose of comparison of geometry compression techniques.

7. REFERENCES

[1] P. Alliez and M. Desbrun, "Valence-driven connectivity encoding for 3d meshes.," *Comput. Graph. Forum*, vol. 20, no. 3, 2001.

[2] J. Rossignac, "Edgebreaker: Connectivity compression for triangle meshes.," *IEEE Trans. Vis. Comput. Graph.*, vol. 5, no. 1, pp. 47–61, 1999.

[3] Z. Karni and C. Gotsman, "Compression of soft-body animation sequences," *Computers & Graphics*, vol. 28, no. 1, pp. 25–34, 2004.

[4] M. Sattler, R. Sarlette, and R. Klein, "Simple and efficient compression of animation sequences," in *ACM/EG Symposium on Computer Animation*, NY, USA, 2005, pp. 209–217, ACM Press.

[5] I. Guskov and A. Khodakovsky, "Wavelet compression of parametrically coherent mesh sequences," in *Proceedings of ACM/EG Symposium on Computer Animation*, August 2004, pp. 183–192.

[6] F. Payan and M. Antonini, "Temporal wavelet-based geometry coder for 3d animations," *Computers & Graphics*, 2006.

[7] S. Varakliotis, S. Hailes, and J. Ostermann, "Optimally smooth error resilient streaming of 3d wireframe animations.," in *VCIP*, 2003, pp. 1009–1022.

[8] K. Muller, A. Smolic, M. Kautzner, P. Eisert, and T. Wiegand, "Predictive compression of dynamic 3D meshes," in *International Conference on Image Processing*, 2005, pp. I: 621–624.

[9] J.-H. Yang, C.-S. Kim, and S. U. Lee, "Compression of 3-d triangle mesh sequences based on vertex-wise motion vector prediction.," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 12, no. 12, pp. 1178–, 2002.

[10] L. Ibarria and J. Rossignac, "Dynapack: space-time compression of the 3d animations of triangle meshes with fixed connectivity," in *Proceedings of Eurographics '03*. 2003, pp. 126–135, Eurographics Association.

[11] N. Stefanoski and J. Ostermann, "Connectivity-guided predictive compression of dynamic 3d meshes," in *Proceedings of the International Conference on Image Processing, ICIP2006*, Atlanta, 2006.

[12] K. Mamou, T. Zaharia, and F. Prêteux, "A skinning approach for dynamic 3d mesh compression: Research articles," *Comput. Animat. Virtual Worlds*, vol. 17, no. 3-4, pp. 337–346, 2006.

[13] Robert Sedgewick and Philippe Flajolet, *An Introduction to the Analysis of Algorithms*, Addison-Wesley, 1996.