Predictive Compression of Dynamic 3D Meshes

K. Müller, A. Smolic, M. Kautzner, P. Eisert, and T. Wiegand
Fraunhofer Institute for Telecommunications, Heinrich-Hertz-Institut, Image Processing Department
Einsteinufer 37, 10587 Berlin, Germany
{kmueller/smolic/kautzner/eisert/wiegand}@hhi.de

***Abstract*—An efficient algorithm for compression of dynamic time-consistent 3D meshes is presented. Such a sequence of meshes contains a large degree of temporal statistical dependencies that can be exploited for compression using DPCM. The vertex positions are predicted at the encoder from a previously decoded mesh. The difference vectors are further clustered in an octree approach. Only a representative for a cluster of difference vectors is further processed providing a significant reduction of data rate. The representatives are scaled and quantized and finally entropy coded using CABAC, the arithmetic coding technique used in H.264/MPEG4-AVC. The mesh is then reconstructed at the encoder for prediction of the next mesh. In our experiments we compare the efficiency of the proposed algorithm in terms of bit-rate and quality compared to static mesh coding and interpolator compression indicating a significant improvement in compression efficiency.**

Keywords: 3D Mesh, Dynamic Meshes, Animation Coding, DPCM, Spatial Clustering

I. INTRODUCTION

In the MPEG-4 Visual standard, the coding of dynamic meshes is specified by using 3D mesh coding (3DMC) for the first mesh and Interpolator Compression (AFX-IC) for the animation part. For some specific meshes, i.e. head or body models, facial and body animation parameters have been introduced that also represent dynamic time-consistent 3D meshes. We consider generic time-consistent 3D meshes, for which we have developed a new coding scheme for efficient transmission. The algorithm is not only usable for our application involving 3D Video Objects (3DVOs) but for other computer graphics applications as well. Our coding algorithm compresses dynamic meshes in a new and highly efficient way by temporal prediction, spatial clustering, and context-based adaptive binary arithmetic coding.

A number of contributions to 3D mesh coding have been published in the past. Polygon meshes as a 3D surface representation with 3D points and connectivity have initially been considered for compression in a static way. The contributions were focused on compression of vertex positions and connectivity [14][4]. Static mesh coding exploiting spatial dependencies of adjacent polygons is also currently part of MPEG-4 [8]. With the creation of animated meshes and mesh sequences with changing vertex positions over time, dynamic approaches have been introduced. In [12] a decomposition of the vertex position matrix is suggested, allowing a better decorrelation of different types of mesh deformation over time. The deformation is described by special animation parameters, representing affine motion or free-form deformation. The residual between real mesh deformation and estimated animation parameter deformation is then coded. The temporal deformation is separated into low and high frequency motion - an approach that was later used in [15] to introduce a multiresolution approach for dynamic mesh coding. Dynapack [6] analyzes spatial and temporal dependencies using a predictor to exploit similarities of neighboring points. Besides this prediction, 3D points are compressed directly and not represented by substitutes, an approach that is taken in [1]. Here, principal component analyses (PCA) of the geometry covariance matrix is carried out to reduce spatial correlation. By applying linear predictive coding to PCA components, a fast algorithm was introduced in [11].

Finally, a spatial clustering algorithm for motion vectors was introduced in [15], predicting motion vectors within the object's bounding cube by tri-linear interpolation of the cube's corner vectors. This approach was adopted for spatial motion vector clustering which is one component of the algorithm that is described in this paper.

II. CODING OF DYNAMIC 3D MESHES

MPEG-4 Visual currently provides an efficient tool for compression of static 3D meshes [8]. In the dynamic case, the 3DMC algorithm codes each mesh of a sequence separately (corresponding to intra only coding in video). We assume time-consistent meshes that contain a large degree of temporal statistical dependencies that can be exploited for further compression. We have therefore developed a novel compression scheme for such time-consistent dynamic 3D meshes (Differential 3DMC (D3DMC)).

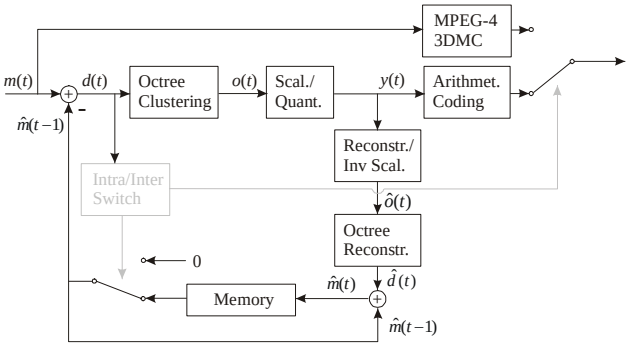

Figure 1. Block diagram of the D3DMC encoder

Figure 1 shows a block diagram of the encoder. It contains MPEG-4 3DMC as fallback mode that is enabled through the Intra/Inter switch that is fixed to either one for each 3D mesh of a sequence. This Intra mode is used for instance when the first mesh (I mesh) of a Group of Meshes (GOM) is encoded, i.e., when no prediction from previously decoded meshes is used. Additionally, I meshes can be used by the encoder in any

other case, e.g., when the prediction error in D3DMC becomes too large. This mode provides backward compatibility to 3DMC and ensures that D3DMC can never be worse than 3DMC. The newly introduced predictive mode for mesh coding (P meshes) consists of the following steps:

1. The previously decoded mesh is subtracted from the current mesh to be encoded. This step can only be done if time-consistent meshes with a common connectivity are available, and therefore we have constrained the mesh extraction process as described above. Only the difference signal between original and prediction is further processed which are called difference vectors.
2. The spatial clustering algorithm by Zhang and Owen [15] is applied to the difference vectors, in order to compute only very few representatives for a number of vectors. This algorithm uses an octree structure with large cell sizes in spatial regions of homogeneous motion and small sizes for outliers. The result is a number of substitute vectors representing the motion vectors and octree structure information.
3. The substitute vectors are passed to an arithmetic coder using context-adaptive binary arithmetic coding (CABAC) [13] to efficiently adapt to the signal statistics. Analysis of the probability density function (pdf) of the data has shown a Laplacian distribution that is superimposed by small peaks at varying positions due to the clustering algorithm. For CABAC, the binarization uses unary/kth-order Exp-Golomb codes to not only assign small code words to most frequent symbols but also fit to certain outliers that frequently occur. Furthermore, the unary as well as the Exp-Golomb part sizes are calculated from the data to minimize overall code length. Thus the algorithm can adapt to changing statistics, since 3D meshes exhibit a variety of global and local motions that significantly modify the corner or substitute vector distribution An important feature of CABAC is its usage of multiple probability models to better fit the input signal statistics i.e. that for each element of the unary part of the unary/kth-order Exp-Golomb codeword a different probability model is applied. These probability models adapt efficiently the most frequent codewords. Two additional probability models are used to encode the resulting octree structure. One probability model adapts to bits representing nodes and leaves of the octree the second probability model adapts to bits which give evidence if a node or leaf of the octree contains data to be encoded.

III. Experiments

The coding experiments consist of two parts.

1. The introduced predictive coding algorithm is compared to the 3DMC static mesh coding from MPEG-4 [8]
2. The compression of the P meshes or animation information is compared to the state-of-the-art MPEG-4 AFX interpolator compression (AFX-IC)

The rate distortion curves measure bit-rate in kbit/s against geometry distortion between reconstructed and original mesh using the Hausdorff distance [5] providing the maximum Euclidean distance between two wireframes. To analyze the quality, the average root mean squared error (AVGRMSE) is used for mesh-to-mesh comparison, as it represents a common measure for mesh evaluation, as in "Mesh" [3] and "Metro" [2]; two tools that automatically calculate distances between 3D meshes. "Metro" additionally provides a visual comparison, which was used to generate visual results in this paper. For the dynamic mesh comparison with AFX-IC, the average distortion error (D_A) is used as specified in [7]. Here a trapezoidal 2D displacement area is calculated between consecutive meshes, including the spatial mesh error and the temporal distance between consecutive P meshes. The temporal distance has to be considered in case of varying temporal distances between consecutive meshes, as described for the coordinate interpolation syntax in the following subsection. The experiments were performed with the "Chicken Crossing" sequence, which consists of 400 time-consistent meshes with 3030 vertices produced by animation of a Chicken model as well as the "Humanoid" sequence, with different resolutions at 498, 1940 and 7646 vertices and 153 time-consistent meshes.

A. 3DMC Coding Results

In Figure 2 the results for D3DMC are compared with 3DMC. For D3DMC results for a GOM size of 10 (D3DMC GOM10) and 400 (D3DMC GOM400) are shown. A GOM size of 400 means that only the first mesh of the sequence is coded with 3DMC and the rest with D3DMC. A significant increase of compression performance is shown compared to 3DMC.

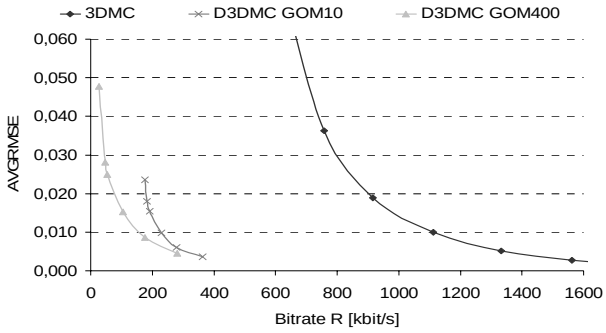

Figure 2. Average RMSE over bit-rate for 3DMC, D3DMC with GOM10 and GOM400, Chicken Crossing sequence

3DMC requires for the same distortion an order of magnitude more bit-rate compared to D3DMC. When reducing the amount of 3DMC-coded meshes for D3DMC by moving from GOM10 to GOM400, another factor of 2 in bit-rate reduction can be achieved.

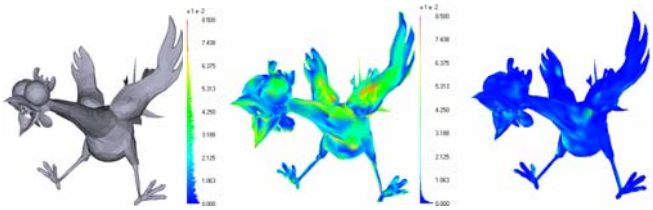

Figure 3. Original mesh (left) and reconstruction error and error distribution using 3DMC at 915 kBit/s (middle) and D3DMC GOM10 at 364 kBit/s (right), Chicken sequence

Figure 3 visualizes the reconstruction error. The original mesh is shown on the left, 3DMC in the middle and D3DMC on the right. The colors (see histogram to the left of the error images, where blue represents a small error, red a large error) indicate the amount of reconstruction error. Both color codes have a

maximum value of 0.085 to allow better comparison. The color histograms show, how the reconstruction quality has improved for D3DMC in comparison to 3DMC. The middle picture of Figure 3 shows an error distribution with large errors. In comparison to that, error distribution for D3DMC on the right is limited to very small values not exceeding the value 0.001.

B. Animation Coding Results

For state-of-the-art coding of dynamic meshes or mesh animations, usually MPEG-4 AFX Interpolator Compression (AFX-IC) is used for compression of dynamic meshes. Therefore D3DMC is compared to AFX-IC as shown in Figure 4.

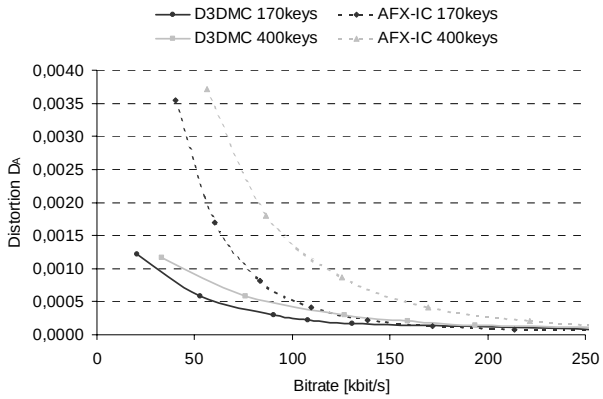

Figure 4. Distortion over bit-rate for D3DMC and AFX-IC with 170 and 400 keyframes (GOM400), Chicken Crossing sequence

First the mesh sequence was coded as GOM400 using both methods (gray curves). Here D3DMC performs better than AFX-IC especially for lower bit-rates. As an example, the bit-rate is reduced by 50% at a distortion measure of 0,001.

Additionally, the number of meshes within the GOM was reduced to a subset of 170 key-meshes out of the original 400 P meshes before coding (black curves). This approach is similar to video coding, where a number of successive B frames is interpolated from the leading and trailing P frame. Here, the P frames represent the key-meshes, while the B frames are neglected. Thus, only the I mesh of a GOM and all key-meshes are coded and transmitted. For the shown mesh sequence, the temporal distribution of key-meshes was obtained from the motion within the sequence: In case of linear motion over a larger number of meshes only few key-meshes were selected, while in areas of abrupt motion changes key-meshes are selected rather dense. After decoding, the neglected meshes have to be linearly interpolated for each single 3D point. If a 3D point exhibits non-linear motion between two key-meshes, interpolation errors are introduced for the points of the intermediate meshes. In the case of lossy compression, the interpolation error is added to the reconstruction error. In Figure 4 the results for the reduced sequence (black curves) are similar to the non-reduced sequence (gray curves). Comparing the reduced and non-reduced sequence for D3DMC (solid curves), a decrease of 30% in terms of bit-rate can be achieved by reducing the sequence from 400 to 170 key-meshes before coding and linearly interpolating the full sequence after decoding.

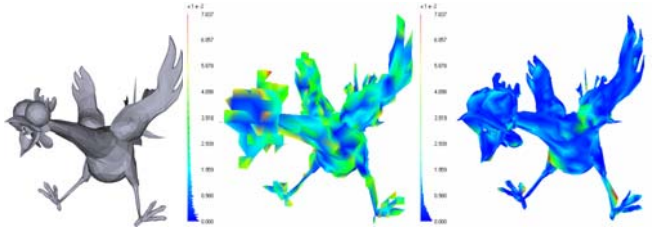

Figure 5. Original mesh (left) and reconstruction error and error distribution using AFX-IC at 125 kBit/s (middle) and D3DMC at 126 kBit/s (right), Chicken sequence

Figure 5 visualizes the reconstruction error. The original mesh is shown on the left, AFX-IC in the middle and D3DMC on the right. Both color codes have a maximum value of 0.078. The color histograms show, how the reconstruction quality has improved for D3DMC in comparison to AFX-IC at the same bit-rate.

For the Humanoid sequence, where different resolutions are given, the results are shown in Figure 6.

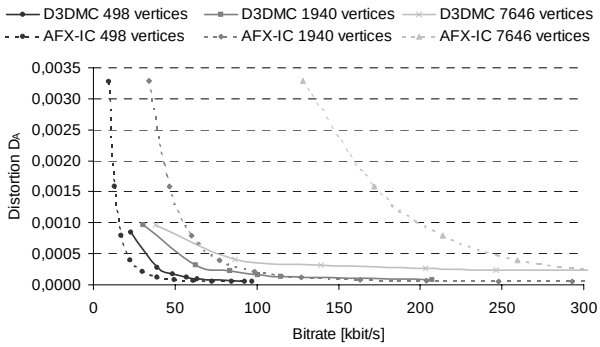

Figure 6. Distortion over bit-rate for D3DMC and AFX-IC with 498, 1920 and 7646 vertices, GOM153, Humanoid sequence

For the lowest resolution (black curves), D3DMC is slightly worse than AFX-IC. For the medium resolution of 1940 vertices, D3DMC performs better than AFX-IC (dark gray curves). For the highest resolution of 7646 vertices D3DMC clearly outperforms AFX-IC (light gray curves). Here the strength of the spatial clustering in D3DMC (solid curves) causes only slightly higher distortions at identical bit-rates for higher resolutions, while the AFX-IC bit-rate (dashed curves) depends directly on the mesh resolution.

This behavior can also be identified in the visual comparisons of Figure 7. Here, the first row shows the reconstruction results for the lowest resolution with 498 vertices, which shows a higher error distribution for D3DMC. In the other resolution cases, D3DMC performs better and in the bottom row of Figure 7, blocking artifacts are visible for AFX-IC, while the results of D3DMC still show smooth reconstruction results even at the lower bit-rate of 86,4 kBit/s in comparison to 128,1 kBit/s for AFX-IC.

IV. Summary and Futur Work

We have presented a coding scheme for dynamic time-consistent 3D meshes. The mesh coding algorithm processes the mesh sequences using of Groups of Meshes (GOMs) with I and P meshes similar to video coding. For I mesh coding, 3D

mesh compression (3DMC) for static meshes is used. For efficient P mesh coding we subtract two consecutive meshes or keyframes in case of a coordinate interpolator representation. One is a decoded mesh available at encoder and decoder and the other mesh is the currently to be coded one. The residual from this subtraction is passed to a clustering algorithm obtaining substitute differential motion vectors to exploit spatial statistical dependencies. Finally, these substitute motion vectors are coded arithmetically, using CABAC for efficiency.

In the experiments we have shown, how the proposed predictive coder performs in comparison to 3DMC, yielding coded sequences of only 10-20% with the same quality compared to 3DMC. A larger GOM-size decreases the bit-rate further, as presented for the synthetic Chicken sequence. For compression of the P meshes, we have shown comparisons to state-of-the-art AFX-IC for different keyframe distances and mesh resolutions. Here, D3DMC shows improvements especially for lower bit-rates. For different mesh resolutions, D3DMC guarantees higher compression efficiency for higher mesh resolutions compared to AFX-IC, due to spatial clustering.

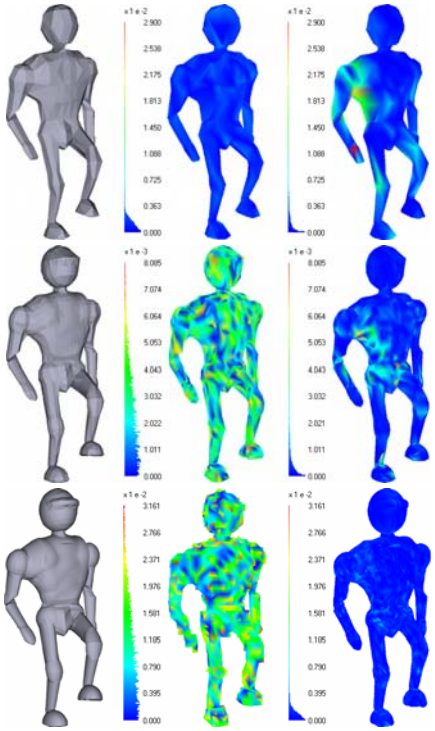

Figure 7. Original mesh (left) and reconstruction error and error distribution using AFX-IC (middle) and D3DMC (right), Humanoid sequence
Top row: 498 vertices, AFX-IC: 22,7 kBit/s, D3DMC: 23 kBit/s
Middle row: 1940 vertices, AFX-IC: 60,1 kBit/s, D3DMC: 62,7 kBit/s
Bottom row: 7646 vertices, AFX-IC: 128,1 kBit/s, D3DMC: 86,4 kBit/s

One drawback of the proposed coding scheme is the use of heuristic thresholds for the clustering step, which is planned to be replaced by a rate-distortion-optimized version. Moreover, additional predictions within the octrees and between octrees as well as additional external contexts for CABAC could further improve performance. Finally, it should be noted that the P meshes in our work can be extended towards B meshes in analogy to H.264/MPEG4-AVC [10] and that the AFX-IC scheme could be seen as a strict subset of such B meshes featuring only the temporal direct motion vector prediction mode. Hence, a combination of AFX-IC with our approach and a further extension will be subject to future work.

ACKNOWLEDGMENT

We would like to thank Microsoft for providing the Chicken Crossing sequence (© Copyright 1996, Microsoft Corporation. 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.).

We would also like to thank Alexandru Salomie from Vrije Universiteit Brussel for providing the Humanoid sequence.

REFERENCES

[1] M. Alexa and W. Müller, "Representing Animations by Principal Components", *Proc. of EUROGRAPHICS 2000*, vol. 19, no. 3, pp. 411-418, 2000.

[2] N. Aspert, D. Santa-Cruz, T. Ebrahimi, "MESH: Measuring errors between surfaces using the Hausdorff distance", *Proc. of the IEEE International Conferende on Multimedia and Expo, ICME2002*, vol. I, pp. 705-708, 2002.

[3] P. Cignoni, C. Rocchini and R. Scopigno, "Metro: Measuring Error on Simplified Surfaces", *Computer Graphics Forum*, vol. 17, no. 2, pp. 167–174, 1998.

[4] S. Gumbold, W. Strasser, "Real Time Compression of Triangle Mesh Connectivity", *Computer Graphics Proceedings, Annual Conference Series, (Proc. of ACM SIGGRAPH 98)*, pp. 133-140, 1998.

[5] D. Huttenlocher, G. Klanderman and W. Rucklidge, „Comparing Images Using the Hausdorff Distance", *IEEE Journal of Pattern Analysis and Machine Inteligence*, vol. 15, no. 9, pp. 850-863, 1993.

[6] L. Ibarria and J. Rossignac, "Dynapack: space-time compression of the 3D animations of triangle meshes with fixed connectivity", *In: Proceedings of the 2003 ACM SIGGRAPH/ Eurographics Symposium on Computer Animation*, pp. 126–135. Eurographics Association, 2003.

[7] ISO/IEC JTC1/SC29/WG11 "Animation Framework eXtension Core Experiments Description", Doc. N6988, Hong Kong, China, 2005.

[8] ISO/IEC JTC1/SC29/WG11 "Information Technology - Coding of Audio-Visual Objects. Part 2: Visual; 2001 Edition", Doc. N4350, Sydney, Australia, 2001.

[9] ISO/IEC JTC1/SC29/WG11, "ISO/IEC 14496-16/PDAM1", Doc. N6544, Redmond, WA, USA, 2004.

[10] ITU-T Recommendation H.264 & ISO/IEC 14496-10 MPEG4-AVC, Advanced Video Coding for Generic Audio-Visual Services, 2003.

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

[12] J. Lengyel, "Compression of Time Dependent Geometry", *Symposium on Interactive 3D Graphics*, pp. 89-95, 1999.

[13] D. Marpe, H. Schwarz and T. Wiegand, "Context-Based Adaptive Binary Arithmetic Coding in the H.264/MPEG4-AVC Video Compression Standard", *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 620-636, 2003.

[14] J. Rossignac, "Edgebreaker: Connectivity Compression for Triangle Meshes", *IEEE Trans. on Visualization and Computer Graphics*, vol. 5, no. 1, pp. 47-61, 1999.

[15] A. Shamir, V. Pascucci, "Temporal and Spatial Level of Details for Dynamic Meshes", *Proc. Of Virtual Reality Systems and Techniques*, pp. 423-430, 2001.

[16] J. Zhang and C. B. Owen, "Octree-based Animated Geometry Compression", *DCC'04, Data Compression Conference*, Snowbird, Utah, USA, pp. 508-517., 2004.