design. At full speed (40 MHz), and maximum array dynamic range (six decades), the power consumption will be 71 mW. Normal operation produces events at a maximum of 4 MHz, for a dynamic range of five decades, while consuming 10 mW. A full VGA with three decades of dynamic range operates at 2.46 MHz and consumes only 7.3 mW. Fig. 4 shows how the output rate per pixel, dynamic range, power and array size scales.

The main drawback to this approach is the complexity of the digital frame grabber required (pixel size, fixed pattern noise and power consumption can be reduced with better circuit designs). To make every spike count, a high-resolution timer ($\simeq$ 24 bits) and a large frame buffer are required ($\simeq$ 15 MB for VGA). The timer indexes each event and compares it with the last time an event at that pixel was recorded. The difference is inversely proportional to the light intensity. The buffer must hold the latest pixel time index and the intensity value. Fig. 5 shows example images recorded with the array. Note that features in the shadows can be seen using a log plot.

16 September 2001
Electronics Letters Online No: 20010969
DOI: 10.1049/el:20010969

E. Culurciello and R. Etienne-Cummings (*Department of Electrical and Computer Engineering, Johns Hopkins University, Baltimore, MD 21218, USA*)

E-mail: euge@jhu.edu

K. Boahen (*Bioengineering Department, University of Pennsylvania, Philadelphia, PA 19104, USA*)

References

1 BOAHEN, K.: 'Point-to-point connectivity between neuromorphic chips using address events', *IEEE TCAS-II*, 2000, **47**, (5), pp. 416–434

2 BRAJOVIC, V., *et al.*: 'Sensor computing', *Proc. SPIE*, 2000, **4109**

3 YANG, W.: 'A wide-dynamic range, low-power photosensor array'. Dig. Tech. Papers of ISSCC94, San Francisco, USA, 1994, pp. 230–231

4 CHO, K.-B., KRYMSKI, A., and FOSSUM, E.R.: 'A 1.2 V micropower CMOS active pixel image sensor for portable applications'. Dig. Tech. Papers of ISSCC2000, San Francisco, USA, 2000, pp. 114–115

5 STEVANOVIC, N., HILLEBRAND, M., HOSTICKA, B.J., and TEUNER, A.: 'A CMOS image sensor for high-speed imaging'. Dig. Tech. Papers of ISSCC2000, San Francisco, USA, 2000, pp. 104–105

Motion-compensated compression of 3D animation models

Jeong-Hwan Ahn, Chang-Su Kim, C.-C. Jay Kuo and Yo-Sung Ho

A new algorithm is proposed to compress animated 3D mesh models. First, an input mesh model is partitioned into segments, and each segment is motion compensated from that of the previous time instance. Then, the motion residuals are effectively compressed by using a transform coding method. It is shown that the proposed algorithm yields a much higher compression ratio than the MPEG-4 codec.

Introduction: 3D animation models are popularly used in various multimedia applications, such as Internet services, computer graphics, and synthetic imaging systems. The MPEG-4 SNHC 3DMC [1] group developed a graphic codec to compress the triangle connectivity and vertex positions of 3D static meshes. One can straightforwardly apply the codec to compress a sequence of 3D static models to achieve the coding task of animated meshes. However, with this approach, the temporal correlation between adjacent mesh frames cannot be exploited. Lengyel [2] and Ahn [3] proposed mesh sequence coding algorithms to exploit the temporal correlation. These algorithms cannot provide sufficient quality reconstruction at low bit rates and, as a consequence, it is difficult to distribute 3D animation contents in real-time over a network of relatively low bandwidth.

A more efficient compression algorithm to encode animated mesh models is proposed in this Letter. First, a graphic model is partitioned into several segments, and each segment is treated as a minimum unit for coding. With the aid of the segmentation process, we can borrow the well-developed concept in video coding techniques by considering a segment as the counterpart of a macroblock. More specifically, an input mesh sequence is classified into two types of meshes: intra-coded meshes (intra-meshes) and inter-coded meshes (inter-meshes). Then, intra-meshes are compressed without reference to other meshes, while the motion-compensated prediction is employed for the coding of inter-meshes to exploit the temporal correlation.

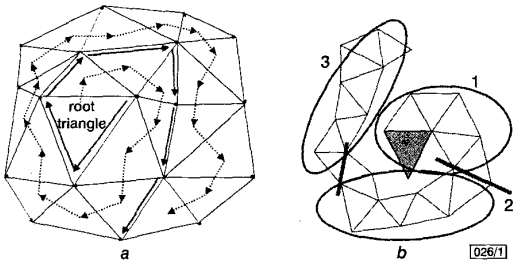

Fig. 1 *Triangle strip representation and segmentation*

a Representation
b Segmentation

Segmentation: In general, a typical 2D image is defined on a regular rectangular grid, but a 3D mesh has arbitrary connectivity in the 3D space. Therefore, it is not easy to design a universal coding scheme for 3D models based on a specific data distribution. Here, we convert the triangular mesh structure into the triangular linear strip form [4] in order to represent the connectivity in a regular way as shown in Fig. 1*a*. Then, the triangle strip is partitioned into distinct segmented blocks by grouping the same number of vertices as shown in Fig. 1*b*. Since this decomposition can group adjacent vertices in the spatial domain, we can take advantage of the spatial coherence during the encoding process.

Intra-mesh coding: We encode each mesh model in the intra-frame (called the I-mesh) based on the DPCM structure [3]. Instead of encoding the coordinate values of each vertex point directly, we adopt a differential coding approach to exploit the strong correlation of the coordinate values among adjacent vertex points along the traversal order shown in Fig. 1*a*. Note that any effective 3D static mesh compression algorithm can be used here.

Inter-mesh coding: In general, vertex positions in a mesh can be efficiently predicted from those in the previous time instance, and the entropy of the prediction error is relatively small. We assume that the topology structure is preserved for the whole sequence of the animation model so that the motion trajectory of any vertex can be exactly determined. We define the representative motion vector $\overline{MV}_t(k)$ of the kth segment at the tth frame to be the average of motion vectors of vertices in that segment,

$$\overline{MV}_t(k) = \frac{1}{N}\sum_{n=0}^{N-1} MV_t(n) = \frac{1}{N}\sum_{n=0}^{N-1}(v_t(n) - v_{t-1}(n)) \quad k = 0, 1, ..., M-1 \quad (1)$$

where $v_t(n)$ denotes the position of the nth vertex in the segment at the tth frame, N is the number of vertices in the segment, and M is the number of segments in the mesh. These motion vectors are quantised, and encoded by a DPCM technique in the space domain.

After defining the motion vector for each segment, we obtain residual signals by subtracting the real motion vector $MV_t(n)$ of each vertex from the representative motion vector $\overline{MV}_t(k)$, given by

$$e_t(n) = \overline{MV}_t(k) - MV_t(n)$$

Even after the segment-based motion compensation, the residual errors $e_t(n)$ exhibit high correlation between adjacent vertices. This high spatial correlation can be effectively exploited by a

transform coding. In our work, we transform the residual errors along x-, y- and z-coordinates using 1D DCT. After the DCT transform, we apply a bounding box quantiser for transform coefficients. Then, quantised DC and AC coefficients are entropy-encoded by using a QM coder.

If a mesh is predicted by using only the information in the previous mesh by eqn. 1, it is called the P-mesh. However, we can further increase the prediction gain for the current mesh by using the bi-directional linear interpolation of vertex positions in the previous and subsequent time instances. This mesh is called the B-mesh. More specifically, the vertex position $v_t(n)$ in the B-mesh is predicted from the weighted sum of the previous vertex position and the subsequent vertex position, given by

$$\hat{v}_t(n) = \frac{m}{m+l} v_{t-l}(n) + \frac{l}{m+l} v_{t+m}(n)$$

As in the P-mesh, the prediction errors $v_t(n) - \hat{v}_t(n)$ are also transformed and entropy-encoded.

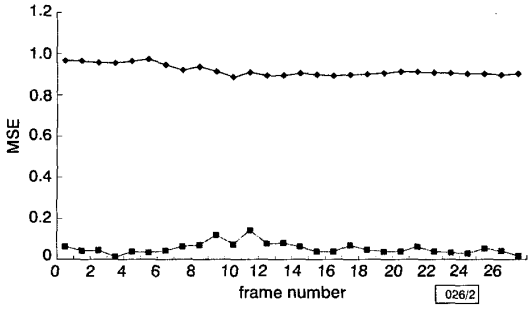

Fig. 2 *Performance comparison*

10:1 compression ratio
—●— MPEG-4 SNHC
—■— proposed

Experimental results: We evaluated the performance of the proposed algorithm with the 'chicken crossing' model [2], which consists of 400 frames, 3030 vertices and 5665 triangles. It requires 14,577,900 bytes of storage space before compression. Our target is 3000 bytes per mesh, which corresponds to a compression ratio of about 10:1. For the MPEG-4 encoder, the bits required for the connectivity are counted just once at the first frame. For the proposed scheme, the temporal frame structure is set as IBBPBBP ...PBBP, the segmentation block size is 16, and the quantisation bits are allocated as follows; 5 bits for MV, and 11, 5, and 2 bits for vertex coordinates in I-, P- and B-meshes, respectively. Fig. 2 compares the performance of the proposed scheme with that of the MPEG-4 in terms of MSE. Note that the proposed algorithm provides 10 times lower MSE than MPEG-4 at the same bit rate.

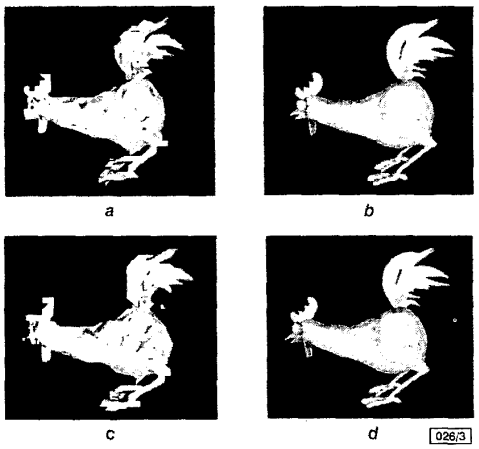

Fig. 3 *Reconstructed models*

a MPEG-4 SNHC
b Proposed scheme at 9th frame (B-mesh)
c MPEG-4 SNHC
d Proposed scheme at 11th frame (P-mesh)

Fig. 3 shows reconstructed models at the 9th and 11th frames for visual quality comparison. It can be seen that the MPEG-4 codec yields severe distortion with sharp discontinuities. In contrast, the proposed algorithm reconstructs these models very faithfully without noticeable artifacts, since it efficiently exploits the temporal and spatial correlations in the mesh sequence by using motion-compensated prediction and DCT transform coding of motion residuals.

Conclusion: In this Letter, we have proposed a new compression algorithm applicable to 3D animation models based on mesh segmentation, motion-compensated prediction, and transform coding. It has been demonstrated by experimental results that the proposed algorithm outperforms the MPEG-4 codec both in reconstructed visual quality and in the MSE distortion measure, by exploiting the spatio-temporal correlation in the animated model efficiently.

6 August 2001
Electronics Letters Online No: 20010993
DOI: 10.1049/el:20010993

Jeong-Hwan Ahn and Yo-Sung Ho (*K-JIST, 1 Oryong-dong, Puk-gu, Kwangju, 500-712, Korea*)

E-mail: jhahn@kjist.ac.kr

Chang-Su Kim and C.-C. Jay Kuo (*University of Southern California, Los Angeles, CA 90089-2564, USA*)

References

1 'Description of core experiments on 3D model coding' ISO/IEC JTC1/SC29/WG11 MPEG98/N244rev1, Atlantic City, October 1998
2 LENGYEL, J.E.: 'Compression of time-dependent geometry'. ACM Symp. Interactive 3D Graphics, Atlanta, 1999
3 JEONG-HWAN AHN, and YO-SUNG HO: 'Segmentation and compression techniques for 3D animation models based on motion trajectory in the spherical coordinate system'. SPIE VCIP 2001, Jan. 2001
4 TAUBIN, G., and ROSSIGNAC, J.: 'Geometry compression through topological surgery', *ACM Trans. Graph.*, 1998, pp. 84–115

Motion parameter estimation by using time-frequency representations

S. Stanković and I. Djurović

The estimation of motion parameters of moving objects by using variable μ-propagation and time-frequency representations is proposed. The spectrogram and the Wigner distribution, two basic time-frequency distributions, are used. Both the velocity and the initial position can be accurately estimated by this approach.

Introduction and motion model: The subspace-based line detection (SLIDE) algorithm for high resolution estimation of the line parameters, as well as its numerous interesting applications, has been proposed in [1]. Recently, the SLIDE algorithm has been used for estimation of velocity of moving objects in video sequences [2]. Motion estimation is a very important topic in video signal processing (video signal compression, road tracking, aerial photograph processing, tomography, traffic control, meteorology, etc.). The velocity determination problem, by using the SLIDE algorithm and constant μ-propagation, is reduced to the frequency estimation in the Fourier domain.

In this Letter, variable μ-propagation is introduced and velocity determination is performed based on the instantaneous frequency estimation. Time-frequency representations for the instantaneous frequency estimation are used [3, 4]. The variable μ-propagation provides estimation of initial position and velocity in a single step.

An image containing a moving object can be represented as $i(x, y, t) = f(x, y) + s(x - x_0 - v_x t, y - y_0 - v_y t)$ where $s(x, y)$ is moving object, $f(x, y)$ is contrast background, while t is the considered frame. The parameters of the moving object are: initial position (x_0, y_0) and velocity (v_x, v_y). The estimation of the motion param-