Segmentation and Compression Techniques for 3D Animation Models based on Motion Trajectory in the Spherical Coordinate System

Jeong-Hwan Ahn and Yo-Sung Ho
Kwangju Institute of Science and Technology
1 Oryong-Dong Puk-Gu, Kwangju, 500-712, Korea

ABSTRACT

In recent days, applications using 3D animation models are increasing. Since the 3D animation model contains a huge amount of information, data compression is needed for efficient storage or transmission. Although there have been various proposals for 3D model coding, most works have considered only static connectivity and geometry information. Only a few studies have been presented for 3D animation models. This paper presents a coding scheme for 3D animation models using a new 3D segmentation algorithm. For an accurate segmentation, we take advantage of temporal coherence in the generic animated 3D model. After the motion vector of each vertex is mapped onto the surface of the unit sphere in the spherical coordinate system, we partition the surface of the sphere equally to have the same area. We then reconstruct in-between 3D models using the reconstructed key frame and an affine motion model for each segmented unit.

Keywords: 3D Object Segmentation, 3D Model Coding, Dynamic Mesh Compression, Affine Motion Model

1. INTRODUCTION

Three-dimensional (3D) animation objects are popularly used in various multimedia applications, such as internet services, computer graphics, and synthetic imaging systems. A 3D static model is generally represented by triangular meshes, which can be defined by vertices and their associated edges[1]. A 3D animation model consists of consecutive frames having 3D static models. Figure 1 shows an example of the *CHICKEN CROSSING* animation model[2], which comprises 3030 vertices, 400 frames, and 5665 triangles.

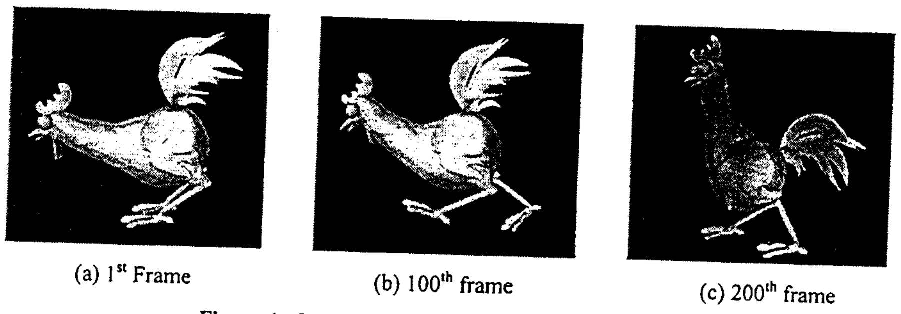

(a) 1st Frame (b) 100th frame (c) 200th frame

Figure 1. *CHICKEN CROSSING* animation model

The original *CHICKEN CROSSING* animation model requires 14,577,900 bytes (3030 vertices $\times$ 400 frame $\times$ 3 coords/vertex $\times$ 4 bytes/coord + 5665 triangles $\times$ 3 indices/triangle $\times$ 2 bytes/index). In order to represent this animation model efficiently, we need to compress dynamic mesh information as well as static mesh data[4-8]. However, despite of its widespread use, not much work for efficient compression of streaming 3D animation has been done[2,3]. In this paper, we describe 3D animation coding based on a new segmentation algorithm.
A block diagram of the proposed scheme is shown in Figure 2. The encoder consists of three functional blocks: coding of the static model, geometric transformation, and compression of residual data. Once we select key frames[9] from a 3D animation model, the 3D static model corresponding to each key frame is encoded by conventional geometry/connectivity compression schemes or Level-of-Detail (LOD)[10] algorithms. We then partition the key frame according to the (θ,ϕ) distribution of the motion trajectory in the spherical coordinate system by grouping parts of similar movement. Finally, we represent the motion vector for each partition using an affine motion model of 12 parameters. In-between models are reconstructed from the key frame and the representative motion vectors using a frame interpolation method.

Correspondence: E-mail: {jhahn,hoyo}@kjist.ac.kr; Telephone: +82-62-970-2258; Fax: +82-62-970-2204

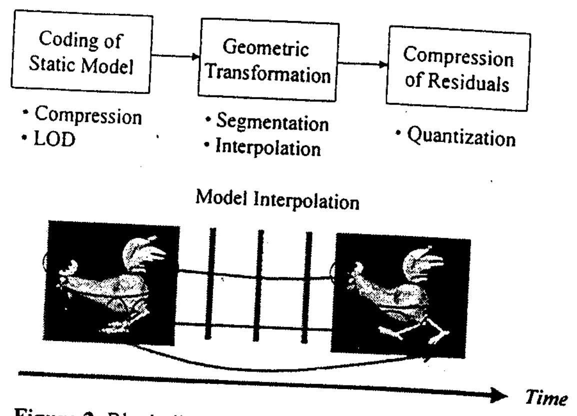


Figure 2. Block diagram of the proposed encoding scheme

2. SEGMENTATION OF 3D STATIC MODELS

In order to encode the animation model efficiently, we partition the given 3D model into semantic regions, each of them can be dealt as a separate object. In this section, we describe three different algorithms for 3D model segmentation: spatial clustering, segmentation based on the topology, and segmentation based on the motion trajectory.

2.1. Spatial Clustering

In the spatial clustering scheme, we partition all triangular faces of the 3D model into N clusters. This clustering algorithm is similar to vector quantization or pattern recongnition. Once a set of seed triangles are initially chosen, we compare distances from the centroid of each triangle to the centers of N clusters, and group the triangle into the nearest cluster, as illustrated in Figure 3.

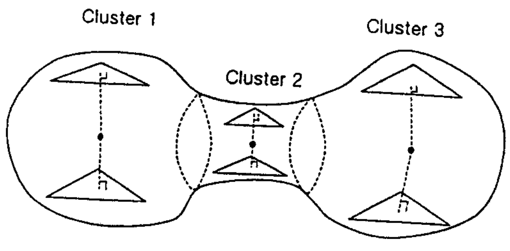


Figure 3. Spatial clustering

2.2. Segmentation based on the Topology

We randomly select one triangle of the 3D static model and traverse triangles in the clockwise direction, as shown in Figure 4. This process resembles the process of orange peeling[5]. This segmentation algorithm produces several independent parts according to the topology of the 3D model. We consider each independent part as one segmented unit.

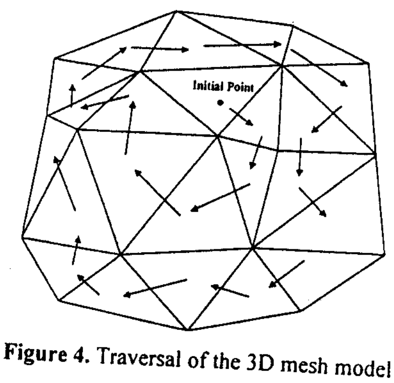


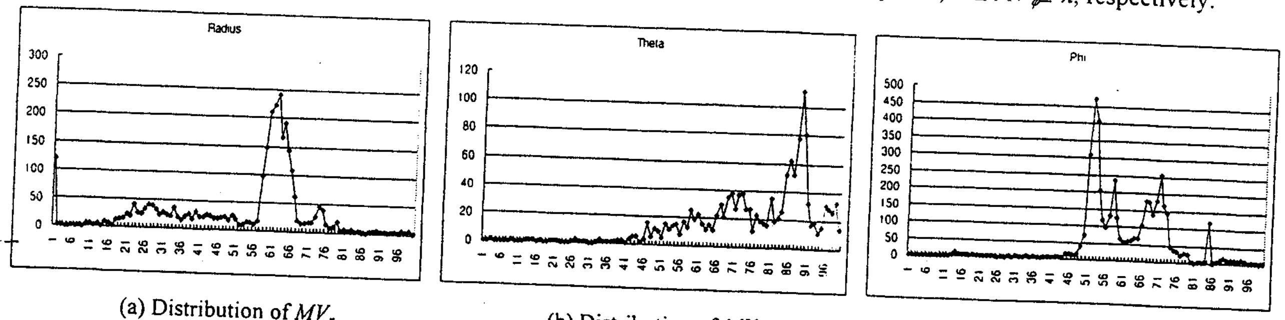
Figure 4. Traversal of the 3D mesh model

2.3. Segmentation based on the Motion Trajectory

In general, consecutive frames in the temporal direction have coherence of moving objects in the animated 3D model. Therefore, we can segment the 3D model by taking advantage of the motion coherence property. A 3D motion vector (MV_x, MV_y, MV_z) for each vertex is calculated between two consecutive key frames, and transformed into $(MV_r, MV_\theta, MV_\phi)$ in the spherical coordinate system by Eq. (1).

$$MV_r = \sqrt{MV_x^{\,2} + MV_y^{\,2} + MV_z^{\,2}}, \quad MV_\theta = \tan^{-1}(MV_y / MV_x), \quad MV_\phi = \cos^{-1}(MV_z / MV_r) \tag{1}$$

Figure 5 shows the distributions of MV_r, MV_θ and MV_ϕ, where the ranges are $0 \le MV_\theta \le \pi/2$, $-\pi \le MV_\phi \le \pi$, respectively.



(a) Distribution of MV_r (b) Distribution of MV_θ (c) Distribution of MV_ϕ

Figure 5. Distribution of motion vectors $(MV_r, MV_\theta, MV_\phi)$ between 1st and 10th frames

Let's consider the rotational movement of a model in the 3D space, which is depicted in Figure 6.

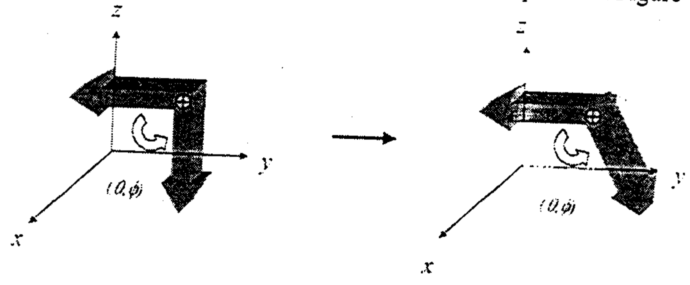


Figure 6. Moving object in the 3D space

When we consider motion vectors (MV_x, MV_y, MV_z) in the spherical coordinate system, normalized motion vectors of similar movement are clustered in a small solid angle of the sphere, as illustraed in Figure 7. Therefore, we can easily segment the 3D animation model by grouping each clustered vectors as one segmented unit.

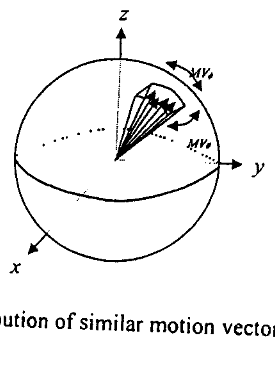
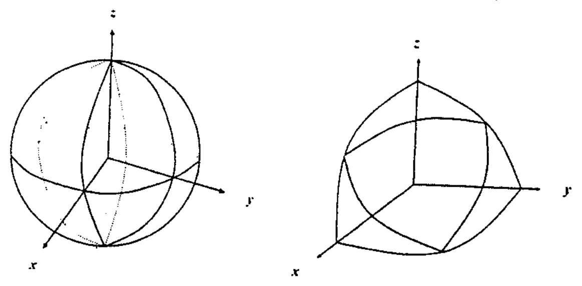


Figure 7. Distribution of similar motion vectors (MV_θ, MV_ϕ)

In order to make a more systematic analysis, we can partition the surface of each sphere equally to have the same area. We can consider two subdivision schemes: 8-4 subdivision[7] and 6-4 subdivision[8].

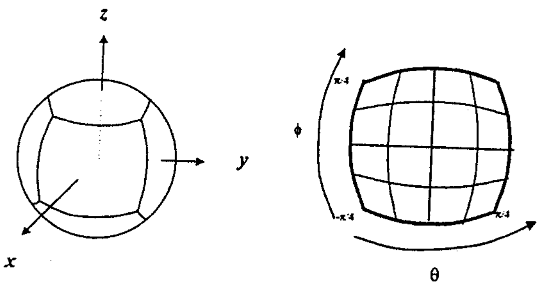
The 8-4 subdivision scheme, which is drawn in Figure 8, divides the surface of the sphere into octants. Since we have octants of the sphere, three bits are used to indicate each octant. Since each octant is symmetrical, we only consider the first octant ($x \geq 0, y \geq 0, z \geq 0$) without loss of generality. Each octant is subdivided further into four triangular-shaped patches of equal size. Each patch is then subdivided recursively into the triangular-shaped smaller patches. This partitioning process produces triangular patches of approximately equal area.



(a) Octant subdivision (b) Four subdivision

Figure 8. 8-4 Subdivision scheme

The 6-4 subdivision scheme, which is shown in Figure 9, is similar to the 8-4 subdivision scheme. However, the surface of the sphere is initially divided into six parts. Each sextant is divided into four rectangular-shaped patches of similar size.



(a) Sextant Subdivision (b) Four subdivision

Figure 9. 6-4 Subdivision scheme

3. MOTION REPRESENTATION AND MODEL INTERPOLATION

In order to represent the motion of each segmented object, we utilize an affine motion model having 12 parameters. We first estimate the 12 parameters using the least square method[11,12]. Once motion parameters are obtained for all segmented units, we can reconstruct in-between models from the key frame and the affine motion parameters by an interpolation technique.

Let *(x,y,z)* be a vertex position in the previous key frame and *(x',y',z')* be the vertex location in the current key frame. Then, we calculate the motion vector (v_X, v_Y, v_Z) for each vertex by Eq. (2).

$$\begin{pmatrix} v_X(x,y,z) \\ v_Y(x,y,z) \\ v_Z(x,y,z) \end{pmatrix} = \begin{pmatrix} x'-x \\ y'-y \\ z'-z \end{pmatrix} \tag{2}$$

Since we use the affine motion model of 12 parameters, the motion vector can be expressed as follows:

$$\begin{pmatrix} \hat{v}_X(x,y,z) \\ \hat{v}_Y(x,y,z) \\ \hat{v}_Z(x,y,z) \end{pmatrix} = \begin{pmatrix} a_1 & a_2 & a_3 \\ a_5 & a_6 & a_7 \\ a_9 & a_{10} & a_{11} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} a_4 \\ a_8 \\ a_{12} \end{pmatrix} \tag{3}$$

In order to estimate 12 affine motion parameters, we define an error function to be minimized:

$$E(a)=\sum_{i=1}^{N}\{[v_X(x_i,y_i,z_i)-\hat{v}_X(x_i,y_i,z_i)]^2+[v_Y(x_i,y_i,z_i)-\hat{v}_Y(x_i,y_i,z_i)]^2+[v_Z(x_i,y_i,z_i)-\hat{v}_Z(x_i,y_i,z_i)]^2\} \tag{4}$$

where N is the number of motion vectors in the same segmented unit.

By substituting Eq. (3) into Eq. (4), we have

$$E(a)=\sum_{i=1}^{N}\{[v_X(x_i,y_iz_i)-(a_1x+a_2y+a_3z+a_4)]^2+[v_Y(x_i,y_i,z_i)-(a_5x+a_6y+a_7z+a_8)]^2$$
$$+[v_Z(x_i,y_i,z_i)-(a_9x+a_{10}y+a_{11}z+a_{12})]^2\} \tag{5}$$

The optimal values of the 12 parameters are estimated by the least square method. The resulting equation is represented in Eq. (6).

$$\sum_{i=1}^{N}\begin{bmatrix} x_i^2 & x_iy_i & x_iz_i & x_i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ x_iy_i & y_i^2 & y_iz_i & y_i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ x_iz_i & y_iz_i & z_i^2 & z_i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ x_i & y_i & z_i & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_i^2 & x_iy_i & x_iz_i & x_i & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_iy_i & y_i^2 & y_iz_i & y_i & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_iz_i & y_iz_i & z_i^2 & z_i & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x_i & y_i & z_i & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_i^2 & x_iy_i & x_iz_i & x_i \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_iy_i & y_i^2 & y_iz_i & y_i \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_iz_i & y_iz_i & z_i^2 & z_i \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_i & y_i & z_i & 1 \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \\ a_{11} \\ a_{12} \end{bmatrix}=\sum_{i=1}^{N}\begin{bmatrix} v_X(x_i,y_i,z_i)\cdot x \\ v_X(x_i,y_i,z_i)\cdot y \\ v_X(x_i,y_i,z_i)\cdot z \\ v_X(x_i,y_i,z_i) \\ v_Y(x_i,y_i,z_i)\cdot x \\ v_Y(x_i,y_i,z_i)\cdot y \\ v_Y(x_i,y_i,z_i)\cdot z \\ v_Y(x_i,y_i,z_i) \\ v_Z(x_i,y_i,z_i)\cdot x \\ v_Z(x_i,y_i,z_i)\cdot y \\ v_Z(x_i,y_i,z_i)\cdot z \\ v_Z(x_i,y_i,z_i) \end{bmatrix} \tag{6}$$

4. EXPERIMENTAL RESULTS

We have evaluated the performance of the segmentation and the model interpolation algorithms with the *CHICKEN CROSSING* model, which is shown in Figure 1. The original *CHICKEN CROSSING* animation model requires 14,577,900 bytes. In our experiment, we have investigated the first 10 frames of the animation model.

4.1. Distortion Measure

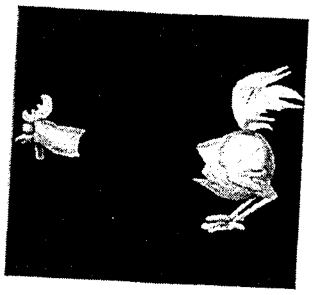
In order to evaluate the distortion between the original 3D model A and the reconstructed one B, we define an objective error metric $E_{dist}(A,B)$ by Eq. (7).

$$E_{dist}=\frac{1}{2}\left(\frac{\sum_{v\in A}d^2(v,B)}{N_A}+\frac{\sum_{v\in B}d^2(v,A)}{N_B}\right) \tag{7}$$

where the distance $d(v,M)=\min_{p\in M}\|v-p\|$ is the minimum distance from v to the closest vertex of M, and $\|\cdot\|$ represents the usual Euclidean vector length operator. N_A and N_B are number of vertices of model A and model B, respectively. Since the distortion metric is the average value of the two asymmetric values, the measure becomes symmetric with respect to the original and the reconstructed models.

4.2. 3D Object Segmentation

Figure 10 shows experimental results of the spatial clustering segmentation algorithm. In this experiment, we partition the model into two and four parts. As shown in Figure 10(a) and Figure 10(b), neighboring triangles are grouped into the same segmentation unit. However, we cannot define an accurate motion vector for each segmented unit because different motion vectors can exist in one segmented unit. Therefore, the decoder cannot reconstruct in-between models with those motion vectors.


(a) Two

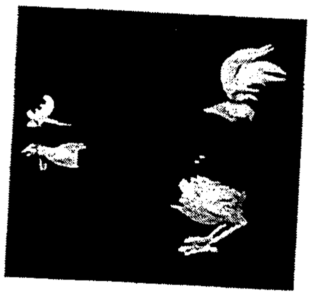

(b) Four

Figure 10. Results of spatial clustering segmentation

Figure 11 shows the experimental result of the segmentation algorithm based on the topology. For the *CHICKEN CROSSING* model, we have 31 segmented parts. This algorithm also produces an inaccurate motion vector for the segmented unit as in the spatial clustering, and we cannot control the number of segmented objects. In addition, since it produces segmented parts by traversing the triangular faces, it is difficult to modify segmented parts intentionally.

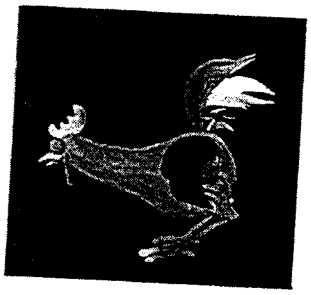
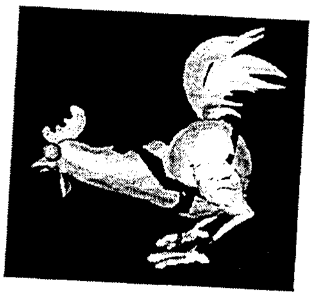


Figure 11. Result of segmentation based on the topology

Figure 12 shows two experimental results of the proposed segmentation algorithm based on the motion trajectory. In this experiment, we partition the original model into four and eight segmented parts. In the first 10 frames of the *CHICKEN CROSSING* model, the head of the chicken moves downward and the foot moves forward. As shown in Figure 12, the segmented parts are the head, the body, the foot and the tail of the model. The results show that the proposed scheme can decompose the 3D animation model into semantic parts having similar motions. Therefore, we can define accurate motion vectors and reconstruct better in-between models.


(a) Four

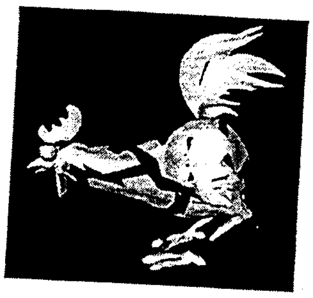
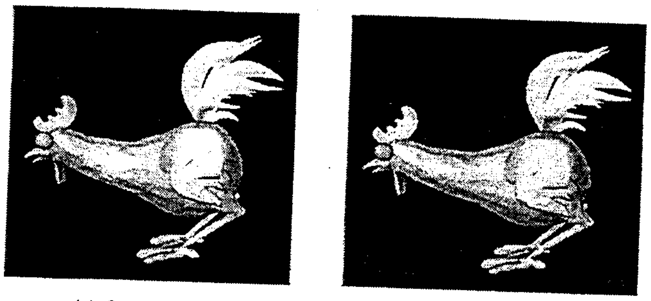

(b) Eight

Figure 12. Results of proposed scheme

4.3. Interpolation of In-Between 3D Models

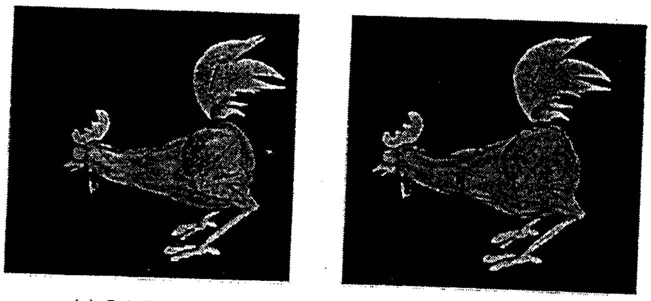
In order to interpolate in-between models, we segment the 0th frame of the given model into 32 partitions using the 8-4 subdivision scheme and estimate affine motion parameters for each partition. We then reconstruct 10 in-between models by a linear interpolation algorithm.

Figure 13 shows the original model at the 5th frame and its interpolated model. Figure 14 shows the original model at the 9th frame and its interpolated model. As we can observe in Figure 13 and Figure 14, visual quality of the reconstructed models by the proposed interpolation scheme is acceptable. Table 1 represents the average distortion E_{dist} of all interpolated frames.



(a) Original model (b) Reconstructed model

Figure 13. 3D model interpolation at the 5th frame



(a) Original model (b) Reconstructed model

Figure 14. 3D model interpolation at the 9th frame

Table 1. Average distortions of interpolated frames

Frame Number	E_{dist}
1	0.259724
2	0.410400
3	0.476859
4	0.501918
5	0.502291
6	0.485860
7	0.455056
8	0.407002
9	0.321586

5. CONCLUSIONS

In this paper, we introduce a new coding scheme for 3D animation models and propose a 3D segmentation algorithm using successive subdivision in the spherical coordinate system. Based on the distribution of motion vectors, we partition the animation model accurately. We use a 3D affine motion model for each segmented object and estimate 12 parameters by the least square method. The proposed scheme demonstrates good performance and reconstructs visually acceptable in-between models.

ACKNOWLEDGMENTS

This work was supported in part by the Korea Science and Engineering Foundation (KOSEF) through the Ultra-Fast Fiber-Optic Networks (UFON) Research Center at Kwangju Institute of Science and Technology (K-JIST), and in part by the Ministry of Education (MOE) through the Brain Korea 21 (BK21) project.

We also want to thank J.E. Lengyel for making the chicken sequence public. The chicken character was created by A. Glassner, T. McClure, S. Benza and M. Van Langeveld. The short sequence of connectivity and position data is distributed solely for the purpose of comparison of geometry compression techniques.

REFERENCES

1. J.D. Foley, *Computer Geaphics: Princeplles and Practice*, Addiosn-Wesley, 1995.
2. J.E. Lengyel, "Compression of Time-Dependent Geometry," *ACM Symposium on Interactive 3D Graphics*, Atlanta, 1999.
3. M. Alexa and W. Muller, "Representing Animations by Principal Components," *Proc. of EUROGRAPHICS*, 2000.
4. M. Deering, "Geometry Compression," *Proc. of SIGGRAPH*, pp.13-20, July 1995.
5. G. Taubin, W.P. Horn, F. Lazarus and J. Rossignac, "Geometry Coding and VRML," *Proc. IEEE*, vol. 86, pp.1228-1243, June 1998.
6. J.H. Ahn and Y.S. Ho, "Geometry Compression of 3D Models using Adaptive Quantization for Prediction Errors," *Picture Coding Symposium*, pp. 193-197, April 1999.
7. J.H. Ahn and Y.S. Ho, "An Efficient Geometry Compression Method for 3D Objects in The Spherical Coordinate System," *IEEE Int'l Conference on Image Processing (ICIP)*, pp. (II)482-486, Oct. 1999.
8. J. Li, C.C. Kuo, and H. Chen, "Embedded Coding of Mesh Geometry," ISO/IEC JTC1/SC29/WG11 MPEG 98/3325, March 1998.
9. J. Vince, *Virtual Reality Systems*, Addiosn-Wesley, 1995.
10. H. Hoppe, "Progressive Meshes," *Proc. of SIGGRAPH*, pp. 99-108, Aug. 1996.
11. M.C. Kim, J.G. Choi, D.H. Kim, H. Lee, M.H. Lee, C.T. Ahn and Y.S. Ho, "A VOP Generation Tool: Automatic Segmetnation of Moving Objects in Image Sequences Based on Spatio-Temporal Information," *IEEE Trans. Circuits Syst. Video Tech.*, vol.9, pp.1216-1226, Dec. 1999.
12. G. Strang, *Linear Algebra And Its Application*, Saunders College, 1988.