Watermarking 3D Polygonal Meshes Using the Singular Spectrum Analysis

MUROTANI Kohei and SUGIHARA Kokichi

Department of Mathematical Informatics Graduate School of Information Science and Technology University of Tokyo 7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-0033, Japan
muro@simplex.t.u-tokyo.ac.jp

Abstract. Watermarking is to embed a structure called a watermark into the target data such as images. The watermark can be used, for example, in order to secure the copyright and detect tampering. This paper presents a new robust watermarking method that adds a watermark into a 3D polygonal mesh in the spectral domain. In this algorithm, a shape of a 3D polygonal model is regarded as a sequence of vertices called a vertex series. The spectrum of the vertex series is computed using the singular spectrum analysis (SSA) for the trajectory matrix derived from the vertex series. Watermarks embedded by this method are resistant to similarity transformations and random noises.

1 Introduction

Digital watermarking is a technique for adding secret information called a watermark to various target objects data. The watermark must not interfere with the intended purpose of the target object data (e.g., the watermark should not decrease the geniality of a 2D image), and the watermark should not be separable from the target object data. Embedded watermarks can be used to secure copyright, to add comments, to detect tampering and to identify authorized purchasers of the object.

In general, watermarks are classified into private watermarks and public ones. Private watermarks are retrieved by both of the original data and the watermarked data, while public watermarks are retrieved by only the watermarked data. A lot of papers on watermarking these data objects have been published [13]. However most of the previous research on watermarking have been concentrating on watermarking "classical" object data types, such as texts, 2D still images, 2D movies, and audio data. Recently, on the other hand, 3D objects data, such as 3D polygonal meshes and various 3D geometric CAD data, become more and more popular and important, and hence techniques to watermark 3D models also become more important [1] [4]-[12].

This paper presents an algorithm that embeds watermark data into 3D polygonal meshes. The watermark embedded by the algorithm is robust against similarity transformation (i.e., rotation, translation, and uniform scaling). It is also resistant against such random noises added to vertex coordinates.

In many techniques for watermarking, secret data (watermarks) are hidden in the spectral coefficients. Therefore, in those techniques of watermarking, some kind of spectrum decomposition is required. The eigenvalue decomposition of a Laplacian matrix derived only from connectivity of the mesh in [9] and the wavelet decomposition for only meshes of certain classes in [4] are computed. Since both methods depend on the connectivity of the mesh, the methods are not robust for altering the connectivity of the mesh. While our method is robust for altering the connectivity of the mesh, since our method do not depend on the connectivity of the mesh.

In section 2, we will review the singular spectrum analysis, which is a basic tool for our algorithm. In section 3, the algorithm of embedding and extracting watermarks will be described. In section 4, we will present experimental results, and in section 5 we conclude this paper with summary and future work.

2 Basic SSA

The singular-spectrum analysis (SSA) [2][3] is a novel technique for analyzing time series incorporating the elements of classical time series, multivariate statistics, multivariate geometry, dynamical systems and signal processing. Recently, SSA is one of the popular statistical methods for signal detection in climatology and meteorology.

In many techniques for watermarking, secret data (watermarks) are hidden in the spectral coefficients. Therefore, in those techniques of watermarking, some kind of spectrum decomposition is required. In this paper, SSA is applied to watermarking, since SSA performs a spectrum decomposition of time series.

In this section, we describe the basic algorithm of SSA. In the next section, a generalized version of SSA to multivariate series is applied to our purpose.

2.1 Algorithm of the Basic SSA

Let $N > 2$. Consider a real-value time series $F = (f_0, f_1, \ldots, f_{N-1})$ of length N. Assume that F is a nonzero series; that is, there exist at least one i such that $f_i \geq 0$. The basic SSA consists of two complementary stages: the decomposition stage and the reconstruction stage.

Decomposition stage The decomposition stage consists of the next two steps.

1st step: Embedding In the first step, the original time series is mapped to a sequence of lagged vectors in the following way. Let L be an integer (window length) such that $1 < L < N$. We define $K = N - L + 1$ lagged vectors X_i by

$$X_i = (f_{i-1}, \ldots, f_{i+L-2})^T, \qquad 1 \leq i \leq K. \tag{1}$$

We shall call X_i's L-lagged vectors. The L-trajectory matrix (or simply trajectory matrix) of the series F is defined by

$$X = [X_0 : \ldots : X_K], \tag{2}$$

whose columns are the L-lagged vectors. In other words, the trajectory matrix is

$$\boldsymbol{X} = (x_{ij})_{i,j=1}^{L,M} = \begin{pmatrix} f_0 & f_1 & f_2 & \cdots & f_{K-1} \\ f_1 & f_2 & f_3 & \cdots & f_K \\ f_2 & f_3 & f_4 & \cdots & f_{K+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{L-1} & f_L & f_{L+1} & \cdots & f_{N-1} \end{pmatrix} \tag{3}$$

Obviously $x_{ij} = f_{i+j-2}$ and the matrix $\boldsymbol{X}$ has equal elements on the 'diagonal' i + j = const. Thus, the trajectory matrix is a Hankel matrix. Certainly if N and L are fixed, then there is a one-to-one correspondence between the trajectory matrix and the time series.

2nd step: Singular value decomposition In the second step, the singular value decomposition (SVD) is applied to the trajectory matrix. Let $\boldsymbol{S} = \boldsymbol{X}\boldsymbol{X^T}$. Denote by $\lambda_1, \ldots, \lambda_L$ the eigenvalues of $\boldsymbol{S}$ taken in the decreasing order of magnitude ($\lambda_1 \geq \ldots \geq \lambda_L \geq 0$), and by $U_1, \ldots, U_L$ the orthonormal system of the eigenvectors of the matrix $\boldsymbol{S}$ corresponding to these eigenvalues. Let $d = Max\{i$, such that $\lambda_i \geq 0\}$. We defines $V_i = \boldsymbol{X^T} U_i/\sqrt{\lambda_i}$ and $\boldsymbol{X_i^T} = \sqrt{\lambda_i} U_i V_i^T$ ($i = 1, \ldots, d$). Then the SVD of the trajectory matrix $\boldsymbol{X}$ can be written as

$$\boldsymbol{X} = \boldsymbol{X_1} + \boldsymbol{X_2} + \ldots + \boldsymbol{X_d}. \tag{4}$$

The matrix $\boldsymbol{X_i}$ have rank 1; therefore they are elementary-matrices. The collection (λ_i, U_i, V_i) is called ith eigentriple of the SVD (4).

Reconstruction stage

3rd step: Diagonal averaging In the last step of the basic SSA, each matrix in the decomposition (4) is transformed into a new series of length N; this step is called the diagonal averaging. Let The matrix $\boldsymbol{Y}$ be an $L \times K$ matrix with elements $y_{ij}, 1 \leq i \leq L, 1 \leq j \leq K$.

Diagonal averaging transfers the matrix $\boldsymbol{Y}$ to the series $(g_0, \ldots, g_{N-1})$ by the formula:

$$g_k = \begin{cases} \frac{1}{k+1} \sum_{m=1}^{k+1} y_{m,k-m+2} & for \quad 0 \leq k < L-1, \\ \frac{1}{L} \sum_{m=1}^{L} y_{m,k-m+2} & for \quad L-1 \leq k < K, \\ \frac{1}{N-k} \sum_{m=k-K+2}^{N-K+1} y_{m,k-m+2} & for \quad K \leq k < N. \end{cases} \tag{5}$$

The expression (5) corresponds to averaging of the matrix elements over the 'diagonal' $i + j = k + 2$: for $k = 0$ we have $g_0 = y_{11}$, for $k = 1$ we have $g_1 = (y_{12} + y_{12})/2$, and so on. Note that if $\boldsymbol{Y}$ is the trajectory matrix of some series $(h_0, \ldots, h_{N-1})$ (in other word, if $\boldsymbol{Y}$ is the Hankel matrix), then $g_i = h_i$ for all i. Diagonal averaging (5) applied to the decomposition matrix $\boldsymbol{X_k}$ produces the series $\tilde{F}^{(k)} = (\tilde{f}_0^{(k)}, \ldots, \tilde{f}_{N-1}^{(k)})$ and therefore the initial series $F = (f_0, \ldots, f_{N-1})$ is obtained by the sum of d series:

$$f_n = \sum_{k=1}^{d} \tilde{f}_n^{(k)}. \tag{6}$$

2.2 Optimality of SVD and the Hankel matrix

Here, we describe two optimal its features in the process of SSA. The first optimality is related to SVD.

Proposition 2.1. Let $\boldsymbol{X} = [X_0 : \ldots : X_K]$ be the matrix define by equation (2), and let $\boldsymbol{X_i}$ be the matrices define by equation (4). Then the following two statements hold.

1. The vector $Q = U_1$ is the solution of the problem

$$\nu_1 := \sum_{i=1}^{K} (X_i, Q) = Max_P \sum_{i=1}^{K} (X_i, P), \tag{7}$$

where the maximum on the right hand side of (7) is taken over all $P \in \boldsymbol{R}^L$ with $||P|| = 1$, and also $\nu_1 = \lambda_1$ holds.

2. Let Q be the solution of the following optimization problem

$$\nu_k := \sum_{i=1}^{K} (X_i, Q) = Max_P^{(k)} \sum_{i=1}^{K} (X_i, P), \tag{8}$$

where the maximum on the right hand side of (8) is taken over all $P \in \boldsymbol{R}^L$ such that $||P|| = 1$ and $(P, U_i) = 0$ for $1 \leq i < k$. If $k \leq d$, then the $Q = U_k$ and $\nu_k = \lambda_k$. If $k > d$, then $\nu_k = 0$.

Proposition 2.1 enables us to call the vector U_i the ith principal vector of collection $X_0, \ldots, X_K$. The second optimality is related to diagonal averaging. When a general matrix is transformed to the Hankel matrix, diagonal averaging have the optimality as stated in the following proposition. Proposition 2.2. Assume that $\boldsymbol{Z} = \psi\ (\boldsymbol{Y})$ is a Hankel matrix of the same dimension as $\boldsymbol{Y}$ such that the difference $\boldsymbol{Y} - \boldsymbol{Z}$ has the minimal Frobenius norm. Then the element $\tilde{y}_{ij}$ of the matrix $\psi\ (\boldsymbol{Y})$ is given by

$$\tilde{y}_{ij} = \begin{cases} \frac{1}{s-1} \sum_{l=1}^{s-1} y_{l,s-l} & for \quad 2 \leq k \leq L-1, \\ \frac{1}{L} \sum_{l=1}^{L} y_{l,s-l} & for \quad L \leq k \leq K+1, \\ \frac{1}{N-s+2} \sum_{l=s-K}^{L} y_{l,s-l} & for\ K+2 \leq k \leq N+1. \end{cases} \tag{9}$$

The linear operator ψ is called the Hankelization operator.

3 Algorithm for Watermarking in the Spectral Domain

The watermarking algorithm in this paper inserts a watermark to a given 3D polygonal mesh. Previous methods using connectivity of the mesh are not robust against modifications of the connectivity of the mesh. In our method, on the other hand, connectivity of the mesh is not used. Instead, a 3D polygonal model is regarded as a sequence of vertices called a vertex series. Therefore, our method is robust against modifications of the connectivity of the mesh. Next, the spectra of the vertex series are computed using SSA for the trajectory matrix derived from the vertex series. The watermarks are added in the spectra domain in such a way that their singular values are modified. To recover the watermarks, the watermarked matrix is converted into the vertex series by the diagonal averaging. This method is for private watermarking, meaning that the watermark extraction requires both the watermarked vertex series and the original non-watermarked vertex series. The watermark can be extracted by comparing singular values of the watermarked data and the original data in the spectral domain.

3.1 Spectral Decomposition of the Vertex Series

Though a scalar-value time series $F = (f_0, f_1, \ldots, f_{N-1})$ is considered in the basic SSA, we expand the basic SSA into a multivariate version in order to use tri-value time series $\boldsymbol{F} = (F_0, \ldots, F_{N-1})$ with $F_i = (f_{i,x}, f_{i,y}, f_{i,z})^T$. The trajectory matrix (2) is expanded into a $3L \times K$ matrix:

$$\boldsymbol{X} = \begin{pmatrix} F_0 & F_1 & F_2 & \cdots & F_{K-1} \\ F_1 & F_2 & F_3 & \cdots & F_K \\ F_2 & F_3 & F_4 & \cdots & F_{K+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ F_{L-1} & F_L & F_{L+1} & \cdots & F_{N-1} \end{pmatrix} \tag{10}$$

and we perform the singular value decomposition for the trajectory matrix (10). The SVD produces a sequence of singular values and a corresponding sequence of elementary-matrices.

Approximately, large singular values correspond to lower spatial frequencies, and small singular values correspond to higher spatial frequencies. Elementary-matrices associated with higher singular values represent global shape features, while elementary-matrices associated with lower singular values represent local or detail shape features.

3.2 Embedding Watermark

Suppose that we want to embed an m-dimensional bit vector $\boldsymbol{a} = (a_1, a_2, \ldots, a_m)$ where each bit takes value 0 or 1. Each bit a_i is duplicated by chip rate c to produce a watermark symbol vector $\boldsymbol{b} = (b_1, b_2, \ldots, b_{mc})$, $b_i \in \{0, 1\}$ of length $m \times c$;

$$b_i = a_j, \qquad jc < i \leq (j+1)c \tag{11}$$

Embedding the same bit c time increases resistance of the watermark against additive random noises, because averaging the detected signal c times reduces the effect of the additive random noises. Let $b' = (b'_1, b'_2, \ldots, b'_{mc})$, $b'_i \in \{-1, 1\}$, be the vector defined by the following simple mapping;

$$b'_k = \begin{cases} -1 & for \;\; b_i = 0, \\ 1 & for \;\; b_i = 1, \end{cases} \tag{12}$$

For $i = 1, 2, \ldots, mc$, let us choose $p_i \in \{-1, 1\}$ in an appropriate way described later. Moreover let α be a positive constant, called the watermark embedding amplitude. The i-th singular value is converted by the following formula:

$$r_i = \sqrt{\lambda_i} + b'_i p_i \alpha. \tag{13}$$

Using r_i, $i = 1, 2, \ldots, d$, we construct the trajectory matrix by

$$\boldsymbol{X'} = \sum_{i=1}^{d} r_i U_i V_i^T = \sum_{i=1}^{d} \sqrt{\lambda_i} U_i V_i^T + \sum_{i=1}^{d} b'_i p_i \alpha U_i V_i^T. \tag{14}$$

From this matrix, the vertex coordinates $\boldsymbol{F'} = (F'_0, \ldots, F'_{N-1})$ with $F'_i = (f'_{i,x}, f'_{i,y}, f'_{i,z})^T$ are computed by using the formula (4). As a result, the vertices of the original polyhedral mesh are converted into watermarked vertices, which are slightly altered from the original positions.

Finally, if $p_i \in \{-1, 1\}$ is chosen randomly, the both centers of gravity $\phi(\boldsymbol{F})$ and $\phi(\boldsymbol{F'})$ are quite different where $\phi(\boldsymbol{F})$ is the center of gravity of vertex series $\boldsymbol{F}$. The center of gravity is a very important invariant for recovering the watermark, and hence it is desirable select p_i so that $\phi(\boldsymbol{F}) = \phi(\boldsymbol{F'})$. Since we can not satisfy $\phi(\boldsymbol{F}) = \phi(\boldsymbol{F'})$ accurately, we try to minimize the absolute value of the second term in the right-hand side of equation (14), i.e., we obtain the optimal p_i by solving the optimal problem:

$$Min || \sum_{i=1}^{d} b'_i p_i \phi \circ \psi(U_i V_i^T) ||_2 \qquad s.t. \quad p_i \in \{-1, 1\}. \tag{15}$$

3.3 Extracting Watermark

In this method, extraction of the watermark requires both of the original vertex series and a watermarked vertex series. The extraction stars with fitting the original vertex series and the watermarked vertex series by translation, rotation and scaling. First, the data are translated so that the center of gravity of the watermarked vertex series considers with the center of gravity of the original vertex series. Next, coarse approximations of their vertex series are reconstructed from the first (highest-frequency) 15 singular values and the corresponding elementary-matrices. (Though number 15 is reasonably determined by our experiences, this number need not always be 15.) Then, each set of eigenvectors (i. e., three principal axis for each vertex series) is computed from a 3×3 covariance matrix derived from each reconstructed shape. Then the data rotated so that the directions of two sets of eigenvectors coincide with each other.

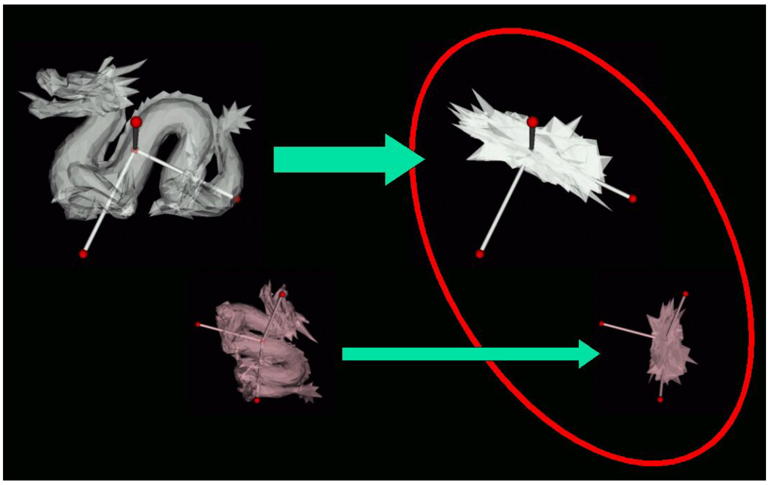


Fig.2.1. The perception of similarity transformation.

Next, SVD is performed for the original vertex series $\boldsymbol{F}$ to produce the singular values $\sqrt{\lambda_i}$ and the associated elementary-matrices. For the watermarked vertex series $\boldsymbol{F'}$, we do not apply SVD, but use the orthogonal matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ for $\boldsymbol{F}$ and compute $\sqrt{\lambda'_i}$ by the equations (7) and (8). Here, if we multiply the difference $(\sqrt{\lambda'_i} - \sqrt{\lambda_i})$ with p_i and sum the result over c times, then we obtain q_j:

$$q_j = \sum_{i \in I_j} (\sqrt{\lambda'_i} - \sqrt{\lambda_i}) p_i \approx \sum_{i \in I_j} b'_i p_i^2 \alpha$$
$$I_j = \{j, j+m, j+2m, \ldots, j+(c-1)m\}. \tag{16}$$

As the p_i for the embedding and the extraction are synchronize, and if disturbances applied to the vertex coordinates are negligible,

$$q_j = b'_i \alpha c. \tag{17}$$

where q_j takes one of the two values $\{-ac, ac\}$. Since α and c are always positive, simply testing the signs of q_j recovers the original message bit sequence a_j:

$$a_j = \{sign(q_j) + 1\}/2. \tag{18}$$

As a result, we can extract the embedded watermark.

3.4 Vertex Series Partitioning

When we watermark a large mesh, we partition the vertex series into smaller vertex sub-series of treatable size as Color Plate 3, so that the calculations times are decreased and the accuracies are increased. The embedding watermarks and extracting watermarks are performed for individual vertex sub-series separately. In the experiments in this paper, as seen in Color Plate 3, the vertex series is partitioned into 5.

4 Experiments and Results

4.1 Parameters

Chip rate *c* In case of watermarking 2D still images and 2D movies, the chip rate c can be quite large, since there are at least a few ten thousands of numbers to be manipulated for watermarking. But, in case of 3D polygonal meshes, the numbers to be manipulated for watermarking is often not as much as the 2D data. In this paper, we use two popular mesh models, the "bunny" model (1494 vertices, 2915 faces) and the "dragon" model (1257 vertices, 2730 faces), both of which have a little more then a thousand vertices. We chose the chip rates as 15 and 12 respectively, since we embed a 55 bit data and a 50 bit data respectively. (In this case, the maximum chip rates are respectively 20 and 18.) If a mesh is fixed, a higher chip rate means a lower data capacity and more robustness.

The watermark embedding amplitude $\boldsymbol{\alpha}$ The watermark embedding amplitudes α is defined as a function of the largest length l of edge of the axis-aligned bounding box of the target mesh. In this paper, we define as $\alpha = \beta \times l$, in the Color Plate, the appearances for $\beta = 0.01, 0.1, 1$ are presented. If the a is larger, the watermark withstand against much disturbances, (for example, adding uniform random noises and mesh smoothing) and the appearance is not preserved.

4.2 Appearances of the watermarked meshes

Color plates 1 and 2 show appearances of the watermarked meshes. In color Plate 1, (1-1) shows the original mesh, while (1-2) (1-3) (1-4) show the watermarked meshes for $\beta = 0.01$, 0.1 and 1, respectively. Color Plate 2 shows another mesh in the same manner. The appearances of (1-2) (1-3) or (2-2) (2-3) can hardly be distinguished from the appearance of (1-1) or (2-1); thus they are watermarked successfully. On the other hand, the appearances of the original meshes are not preserved in (1-4) or (2-4); thus the watermarks are too large in those cases.

4.3 Robustness

We experimentally evaluate the robustness of our watermarks against the similarity transformation and random noises.

Similarity Transformation Watermarks embedded by our method are robust against similarity transformation, for we insert the watermarks in such a way that the center of gravity of all the vertices moves as small as possible, and coequally the transformation can be identified and inverted by the method described in section 3.3.

In case of the dragon model mesh, if the p_i was randomly selected, the center of gravity moved by about $0.01 \times l$ in the 2-norm, so that we obtained false values of the 20 bit data out of the 50 bit data. On the other hand, when the optimal p_i is selected by (15), the center of gravity moved by about $10^{-5} \times l$ in the 2-norm, so that we reconstructed most of the bits correctly.

Added Uniform Random Noises Color Plate 4 shows the appearances of the mesh whose vertex coordinates are disturbed with uniform random noises with amplitude $\alpha \times \gamma$. In the case of $\beta = 0.1$ and $\gamma = 0.001, 0.01, 0.1, 1$, we investigate the number of the values answered correctly out of 50 bits. We repeated the experiment 100 times, and shown in Fig. 4.1.

In Fig 4.1, each vertical line represents one experiment; the red bars show the number of the values answered correctly for $\gamma = 1$, the yellow bars show for $\gamma = 0.1$, the green bars show for $\gamma = 0.01$, the blue bars show for $\gamma = 0.001$, and their bars are superposed in order. If we average the number of the values answered correctly, $\{29.59, 48.02, 49.19, 50.\}$ are gotten for $\gamma = \{1, 0.1, 0.01, 0.001\}$ respectively. From this experiment we can see that the watermark can withstand against uniform noises for about $\gamma \leq 0.1$.

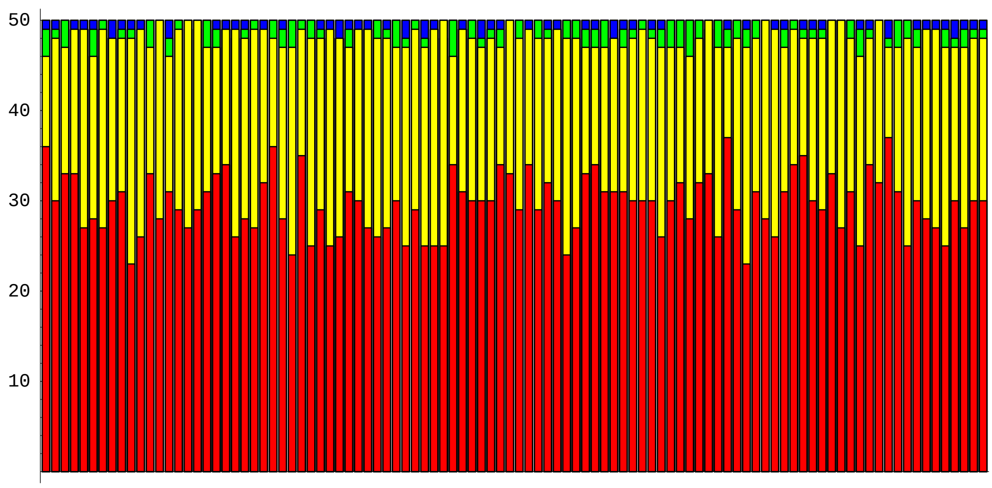


Fig. 4.1. The number of the values answered correctly out of 50 bit 100 times.

5 Summary and Future work

We present a new watermarking algorithm that embeds data into 3D polygonal meshes. The algorithm employs the singular values of the trajectory matrix as the feature to be modified for watermarking. The trajectory matrix is produced by the vertex series of the 3D polygonal mesh. Since our method is a private watermark, we require both the original mesh and the watermarked mesh for extracting the watermark.

The watermark embedded by our method is robust against similarly transformation and moderate uniform noises added to vertex coordinates. This method has a relatively high information density; we require a small mesh having only the $\lfloor 3(m+1)/4 \rfloor$ vertices to embed m bit data without duplicating the watermarks. If we want to watermark a large mesh, the vertex series is partitioned into smaller vertex sub-series with a treatable size, so that the calculations times are decreased and the accuracies are increased. In the future, we would like to investigate the relations between the geometric features of the polygonal meshes and the performance of the watermarking. First, we should modify the vertex series. In this paper, we use random sequences of vertices so that the sizes of the singular values become uniform and we can enlarge α (or β). The watermarks whose a are large are robust, but the appearance of the watermarked meshes is not relatively preserved. Second, the various parameters in our method should be adapted for several purposes. We would like to investigate the features of those parameters.

Acknowledgement

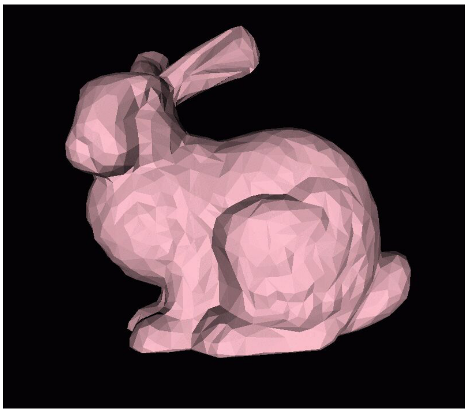
This work is partly supported by the 21st Century CEO Program on Information Science Strategic case of the Japanese Ministry of Education, Science, Sports and Culture.

References

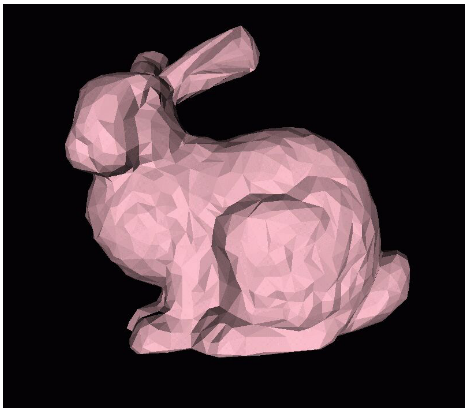
1. Benedens, O., Geometry-Based Watermarking of 3D Models, *IEEE CG&A*, pp. 46-55, January/February 1999.
2. Galka, A., *Topics in Nonlinear Time Series Analysis*, World Scientific, pp. 49-71, 2001.
3. Golyandina, N., Nekrutkin, V., and Zhigljavsky, A., *Analysis of Time Series Structure–SSA and Related Techniques*, Chapman & Hall/CRC, 2001.
4. Kanai, S., Date, H., and Kishinami,T., Digital Watermarking for 3D Polygons Using Multiresolution Wavelet Decomposition, *Proceedings of the Sixth IFIP WG 5.2 International Workshop on Geometric Modeling: Fundamentals and Applications (GEO-6)*, pp. 296-307, Tokyo, Japan, December 1998.
5. Ohbuchi, R., Masuda, H., and Aono, M., Watermarking Three-Dimensional Polygonal Models, *Proceedings of the ACM International Conference on Multimedia '97*, pp. 261-272, Seattle, USA., November, 1997.
6. Ohbuchi, R., Masuda, H., and Aono, M., Watermarking Three-Dimensional Polygonal Models Through Geometric and Topological Modifications, *IEEE Journal on Selected Areas in Communication*, Vol. 16, No. 4, pp. 551-560, May, 1998.
7. Ohbuchi, R., Masuda, H., and Aono, M., Targeting Geometrical and Non-Geometrical Components for Data Embedding in Three-Dimensional Polygonal Models, *Computer Communications*, Vol. 21, pp. 1344-1354, October, 1998.
8. Ohbuchi, R., Masuda, H., and Aono, M., A Shape-Preserving Data Embedding Algorithm for NURBS Curves and Surfaces, *Proceedings of the Computer Graphics International'99*, pp. 180-177, Canmore, Canada, June 7-11, 1999.
9. Ohbuchi, O., Takahashi, S., Miyazawa, T., and Mukaiyama, A., Watermarking 3D Polygonal Meshes in the Mesh Spectral Domain, *Proceedings of the Graphics Interface 2001*, pp. 9-17, Ontario, Canada, June 2001.

10. Praun, E., Hoppe, H., Finkelstein, A., Robust Mesh Watermarking, *ACM SIGGRAPH 1999*, pp. 69-76, 1999.
11. Wagner, M. G., Robust Watermarking of Polygonal Meshes, *Proceedings of Geometric Modeling & Processing 2000*, pp. 201-208, Hong Kong, April 10-12, 2000.
12. Yeo, B-L. and Yeung, M. M., Watermarking 3D Objects for Verification, *IEEE CG&A*, pp. 36-45, January/February 1999.
13. Matsui, K., *Basic of watermarks* (in Japanese), Morikita Shuppan Publishers, Tokyo, 1998.

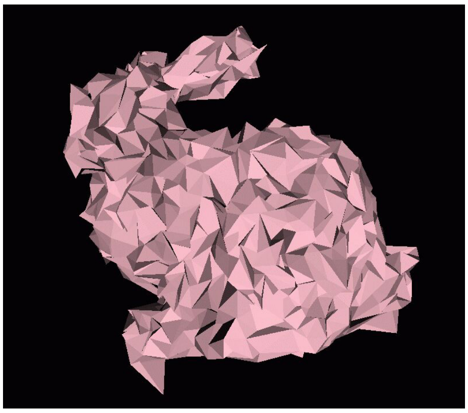
(1-1)



(1-2)



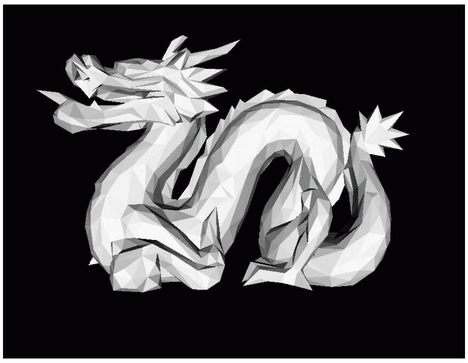
(1-3)



(1-4)

Color Plate 1: Appearance of the "bunny" model (1494 vertices, 2915faces) watermarked with the watermark embedding amplitudes α and the chip rate $c = 15$. The embedded data is 55 bit.

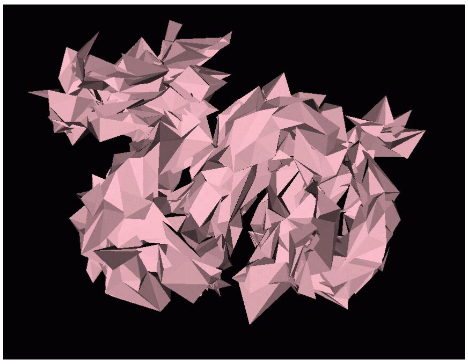
(1-1) The 3D polygonal mesh without watermarks of the bunny model.
The original mesh.
(1-2) The watermarked mesh with $\beta = 0.01$.
(1-3) The watermarked mesh with $\beta = 0.1$.
(1-4) The watermarked mesh with $\beta = 1$.



(2-1)

(2-2)

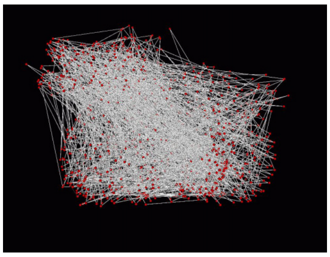
(2-3)



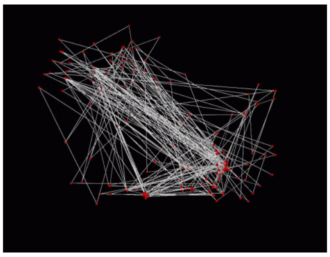
(2-4)

Color Plate 2: Appearance of the "dragon" model (1257 vertices, 2730faces) watermarked with the watermark embedding amplitudes α and the chip rate c = 15. The embedded data is 50 bit.

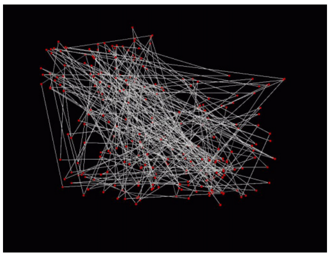
(2-1) The 3D polygonal mesh without watermarks of the dragon model.
The original mesh.
(2-2) The watermarked mesh with $\beta = 0.01$.
(2-3) The watermarked mesh with $\beta = 0.1$.
(2-4) The watermarked mesh with $\beta = 1$.



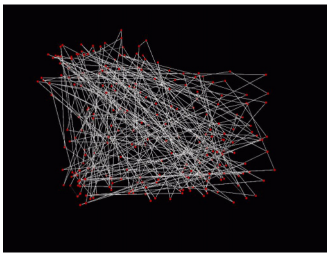
(3-1)



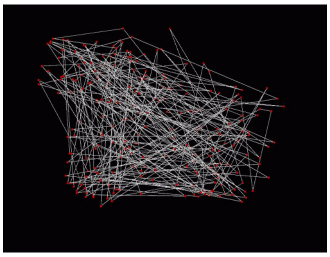
(3-2)



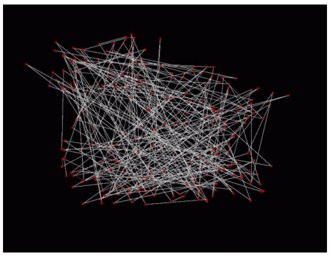
(3-3)



(3-4)



(3-5)



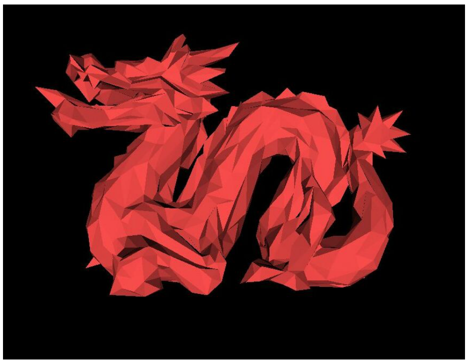
(3-6)

Color Plate 3: The vertex series obtained from the dragon mesh model are partitioned into 5 vertex sub-series.

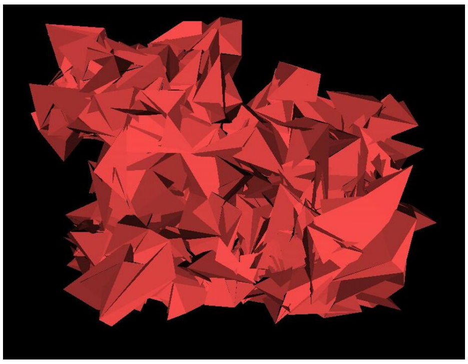
(3-1) The vertex series obtained from the dragon mesh model.
The original vertex series.
(3-2) (3-3) (3-4) (3-5) (3-6) five vertex sub-series of the vertex series (3-1).

(4-1)

(4-2)



(4-3)



(4-4)

Color Plate 4: Appearance of the dragon model added uniform random noises with amplitude $\alpha \times \gamma$. ($\beta = 0.1$)

(4-1) The dragon model added white noises with $\gamma = 0.001$.
(4-2) The dragon model added white noises with $\gamma = 0.01$.
(4-3) The dragon model added white noises with $\gamma = 0.1$.
(4-4) The dragon model added white noises with $\gamma = 1$.