COLLUSION-SECURE FINGERPRINTING FOR THREE-DIMENSIONAL POLYGONAL MESHES

Yung Hur
Basic Technology Research Department
National Security Research Institute
161 Gajeong-dong, Yuseong-gu, Daejeon, 305-350, Korea

Yo-Sung Ho
Dept. of Information and Communications
Kwangju Institute of Science and Technology
1 Oryong-dong, Puk-gu, Kwangju, 500-712, Korea

Abstract

This paper presents a collusion-secure fingerprinting scheme which can embed fingerprints, customer's identification information, into a 3-D polygonal mesh to protect its copyright. In the proposed scheme, we generate fingerprints based on the finite projective geometry, partition a mesh related to the number of bits assigned to each fingerprint, and then embed a watermark, copyright information, into each submesh to be marked. Experimental results demonstrate that our scheme is robust to the additive random noise, 3-D mesh compression, geometrical transformations, and a simple collusion attack like average calculation by traitors. In addition, the number of bits assigned to each fingerprint has been optimized. Using the proposed scheme, we can find traitors responsible for the collusion attack causing unauthorized copy and distribution.

Key Words
Collusion-secure Fingerprinting, Digital Watermarking, Finite Projective Geometry, 3-D Polygonal Meshes

1. Introduction

As the network technology has grown quickly, digital contents, such as audio, still images, moving pictures, and three-dimensional (3-D) data, are distributed through the network easily and fast in recent years. While digital data provide a lot of advantages, many unauthorized copies can be distributed rapidly. This is a serious threat for copyright protection. Thus, digital watermarking has been introduced as a technology to prevent these illegal copies and their distribution. After Tanaka, et al. [1] introduced a watermarking algorithm for still images, various digital watermarking algorithms have been proposed since the mid 1990s.

In the digital watermarking schemes, we embed a watermark signal, i.e., copyright information, into digital contents in order not to be detected or removed by unauthorized users with ease. In addition, a copyright owner should be able to detect it after various attacks, such as filtering, noise, and compression. The technology has been used in many application areas: copyright protection, authentication, copy protection, broadcast monitoring, fingerprinting, etc. Among them, fingerprinting embeds a fingerprint like a serial number of software into digital data to trace illegal copies that are being distributed. However, it has a critical problem. Since users can detect all different places by comparing their own fingerprinted copies one another, they can attack the fingerprinted copies easily.

After Blackey, et al. [2] first discussed the collusion problem, Boneh and Shaw [3, 4] presented a solution against a larger collusion. They introduced c-secure codes to obtain collusion-secure fingerprinting, pointed out that collusions are the most significant problems in fingerprinting scheme, and provided a clear solution for collusion-secure fingerprinting. Dittmann first introduced its application to still images [5, 6]. She combined a robust watermarking algorithm and a collusion-secure fingerprinting algorithm based on the finite projective geometry to embed fingerprints into images. Later, Domingo-Ferrer and Herrera-Joancomartí presented a simple collusion-secure fingerprinting scheme for images using dual binary Hamming codes when the number of traitors is just two [7].

Collusion-secure fingerprinting schemes for still images are, however, not suitable for 3-D mesh models broadly used in CAD (computer aided design) and VR (virtual reality) recently. Thus, this paper presents a new collusion-secure fingerprinting scheme including a new watermarking algorithm to embed fingerprints into 3-D polygonal meshes efficiently.

2. Collusion-secure Fingerprinting for 3-D Polygonal Meshes

First of all, we are going to describe some important terminology [3, 4, 8]. A mark is a portion of an object and has a set of several possible states; a fingerprint is a collection of marks; a distributor is an authorized provider of fingerprinted objects to users; an authorized user is an individual authorized to gain access to a fingerprinted object; an attacker is an individual gaining unauthorized
access to fingerprinted objects, and a traitor is an authorized user distributing fingerprinted objects illegally. In this paper, a cover-mesh is an original mesh before a fingerprint is embedded into it and a stego-mesh is a fingerprinted mesh that may also be attacked by attackers.

Requirements of fingerprinting include collusion tolerance as well as all requirements of watermarking like imperceptibility and robustness [8]. In terms of collusion tolerance, attackers should not be able to find, generate, or delete the fingerprint by comparing the copies even if attackers have access to a certain number of copies. Particularly, the fingerprints must have a common intersection. In terms of imperceptibility, the marks must not decrease the usefulness or quality of the object significantly. In terms of robustness, the fingerprint should still be negotiable if an attacker tampers the object, unless there is so much noise that makes the object useless. Especially, the fingerprint should tolerate lossy data compression.

As a matter of fact, it is not quite easy to implement such a fingerprinting scheme satisfying all the requirements. In order to do that, the proposed fingerprinting scheme consists of fingerprint generation, marking position selection, watermarking, and traitor tracing algorithm. The fingerprint generation algorithm generates fingerprints embedded into a cover-mesh. When we select marking positions on the cover-mesh, the marking position selection algorithm partitions it by the number of bits assigned to each fingerprint and then selects a certain cover-submeshes to mark according to each fingerprint bit information. The watermarking algorithm embeds a watermark into those selected cover-submeshes to be marked. Finally, the traitor tracing algorithm detects traitors. When a modified stego-mesh is inserted, it outputs at least one fingerprint of traitors who contributed to constructing the stego-mesh.

3. Fingerprint Embedding Process

Fig. 1. Fingerprint Embedding Process

Fig. 1 presents a fingerprint embedding process. It consists of the fingerprint generation algorithm, the marking position selection algorithm, and the watermarking algorithm.

Fingerprint Generation Algorithm

$q+1$ fingerprints $FP_1, FP_2, \ldots, FP_{q+1}$ with $k (k=q^d+q^{d-1}+\ldots+q+1)$ bits are generated by applying $d$-detecting fingerprinting scheme based on $PG(d,q)$ by Dittmann's algorithm [5, 6]. The fingerprinting scheme is called $d$-detecting since it, for a maximal number $d$ of traitors, puts enough information in the intersection of up to $d$ fingerprints to uniquely identify all the traitors. In order to implement the scheme, finite projective geometry [9, 10] is used. The finite projective spaces $PG(d,q)$ is constructed from vector spaces over finite fields. When $GF(q)$ is a finite field which exists and is unique for all prime powers $q$ and $V=GF(q)^{d+1}$ is the $(d+1)$-dimensional vector space over $GF(q)$. In general, the $d$-dimensional subspaces of $PG(d,q)$ are the $(d+1)$-dimensional subspaces of $V$.

Since the number of bits assigned to each fingerprint has rapidly increased with input parameters $d$ and $q$, we have tried to reduce the bit number $k$ by finding redundant bits after comparing $q+1$ fingerprints. A redundant bit is defined as follows: If the $j$-th bit in all $q+1$ fingerprints is zero, then it is defined as a redundant bit. And then, all redundant bits are omitted from each fingerprint. Therefore, we can generate new $q+1$ fingerprints with $k'$ bits where $k'=k-\eta$. Here, $\eta$ is the number of omitted redundant bits.

For example, three fingerprints generated based on $PG(2,2)$ that are $1101000, 0011010$, and $1001101$ become $1101000, 0011101$, and $1000111$ since the 7-th bit in them are all zero. In this case, the omitted redundant bit is the 7-th bit. Even though we reduce the number of bits in the fingerprints, the new fingerprints still maintain $d$-detecting property.

Marking Position Selection Algorithm

Let $FP_i=\{f_1,f_2,\ldots,f_r\}$ be a $k$-bit fingerprint assigned to the $i$-th user. The total number of bits for each fingerprint becomes $k'x r$ by repeating each bit with redundancy $r$ in order to be robust to collusion attacks. The redundancy $r$ is computed up to the maximum $r_{\text{max}}$ by $r_{\text{max}}=\left\lfloor \frac{v}{(k'x2m-c)} \right\rfloor$ where $v$ represents the number of vertices in a cover-mesh, $m$ the number of bits in a watermark, $c$ chip rate, $\leq\leq$ an operator which takes only integer. Thus, $FP_i$ becomes a $k'x r$ bit fingerprint $FP_i=\{f_1,f_2,\ldots,f_r\}$. Then, the cover-mesh is partitioned into $s$ submeshes with McTis [11].

The number of submeshes is calculated up to the maximum $s_{\text{max}}$ by $s_{\text{max}}=\left\lfloor \frac{v}{(2m-c)} \right\rfloor$ satisfying $szk'x r$. After $k'x r$ submeshes among the total $s$ submeshes are selected according to PRNS (pseudo random number sequence) generated by stego-key, $k_i$, the one-to-one mapping is
applied to the relation between k×r bits in FP, and selected submeshes. Here, the stego-key prevents traitors recognizing the mapping. Thus, they are able to only know the positions of different vertex coordinates among stego-meshes by collusion attacks. If the i-th bit in FP, is 1, a submesh corresponding to the bit is watermarked. Otherwise, a watermark isn’t embedded into the submesh.

Watermarking Algorithm

By using Terdiman’s scheme [12], several triangle strips are generated from a cover-submesh M, corresponding to the i-th bit of FP, and then DCT transform is implemented on the geometric information of vertices included in the triangle strips. For the simplification of the expression, let’s consider the case of embedding the watermark into N vertices \( v = (x_i, y_i, z_i) \) (1 ≤ i ≤ N) containing in the j-th triangle strip \( ST_j \). To perform DCT transform, vertices \( ST_j = (x_j, y_j, z_j) \) are generated by separating each component of the N vertex coordinates according to x, y, and z coordinates. Then, each component of \( ST_{j,x} \) is DCT transformed by

\[
X_{j,x} = \frac{2}{N} \sum_{n=0}^{N-1} C_x n \cos \left( \frac{2n+1}{N} \pi \right) \times_{j,x} \cos \left( \frac{2n+1}{N} \pi \right), \quad k = 0, 1, \ldots, N - 1
\]

where \( C_x = 1/\sqrt{2} \) when \( k = 0 \), otherwise \( C_x = 1 \). Similarly to \( ST_{j,x} \), \( ST_{j,y} \), \( ST_{j,z} \), DCT transform is applied. As a result, DCT coefficient vectors \( ST_{j,x} = (X_{j,x}, X_{j,y}, X_{j,z}) \), \( ST_{j,y} = (Y_{j,x}, Y_{j,y}, Y_{j,z}) \), \( ST_{j,z} = (Z_{j,x}, Z_{j,y}, Z_{j,z}) \) are generated.

To increase imperceptibility and robustness of the proposed fingerprinting scheme, we select mid-frequency DCT coefficients to embed the watermark into \( ST_{j} \) by modulating amplitude of the DCT coefficients. To modify the coefficients, we employ a spread-spectrum approach similar to [13 - 15] for modulating the sequence of numbers obtained by using DCT transform. The data to be embedded into \( ST_{j} \) is an m-dimensional bit vector \( a = (a_1, a_2, \ldots, a_m) \), \( a_i \in \{0, 1\} \). Each bit \( a_i \) is duplicated by chip rate \( c \) to produce a watermark symbol vector \( b = (b_1, b_2, \ldots, b_m) \), \( b_i \in \{0, 1\} \) of length \( mc \leq N \):

\[
b_i = a_i, \quad jMc \leq (j + 1)c \leq jMc + c
\]

Embedding the same bit c times repeatedly increases robustness of the watermark against additive random noise. Averaging the detected signal by c times upon watermark detection reduces the effect of the additive random noise. The bit vector b is converted to another bit vector \( b' = (b_1', b_2', \ldots, b_m') \) by

\[
b'_i = \begin{cases} -1, & b = 0 \\ 1, & b = 1 \end{cases}
\]

Let us now consider modulating DCT coefficients of one of the DCT coefficient vectors \( ST_{j,x} \). Modulation processes for the other two DCT coefficient vectors \( ST_{j,y}, ST_{j,z} \) are identical. Let \( X_{j,x} \) be the i-th DCT coefficient in \( ST_{j,x}, p_i \in \{-1, 1\} \) be PRNS generated from a known stego-key \( k_2 \) and \( \alpha (\alpha > 0) \) be the modulation amplitude. The modulation amplitude \( \alpha \) is computed by the heuristic method considering imperceptibility and robustness of the scheme. The watermark that may be any company logo or character is used not only to detect fingerprint-bit information from a stego-mesh but also to prove the copyright ownership. The watermarked i-th DCT coefficient \( \tilde{X}_{j,x} \) is computed by

\[
\tilde{X}_{j,x} = X_{j,x} + p_i \cdot \alpha \cdot \alpha
\]

The extraction algorithm requires the same stego-key \( k_2 \), which is a seed for the PRNS used for the embedding, for extraction. Performing the same to \( Y_{j,y} \) and \( Z_{j,z} \) components of \( ST_{j,y} \), \( ST_{j,z} \) produces the watermarked i-th DCT coefficient \( \tilde{Y}_{j,y} \) and \( \tilde{Z}_{j,z} \). Thus, the watermarked DCT coefficient vectors \( ST_{j,x} = (\tilde{X}_{j,x}, \tilde{X}_{j,y}, \tilde{X}_{j,z}) \), \( ST_{j,y} = (\tilde{Y}_{j,x}, \tilde{Y}_{j,y}, \tilde{Y}_{j,z}) \), \( ST_{j,z} = (\tilde{Z}_{j,x}, \tilde{Z}_{j,y}, \tilde{Z}_{j,z}) \) are generated. Using Eq. (5) that is the inverse DCT transform, we produce vertex coordinates of the watermarked i-th triangle strip \( ST_{j,x} = (\tilde{x}_{j,x}, \tilde{x}_{j,y}, \ldots, \tilde{x}_{j,y}) \) from \( ST_{j,x} \):

\[
\tilde{x}_{j,x} = \frac{2}{N} \sum_{n=0}^{N-1} C_x n \cos \left( \frac{2n+1}{N} \pi \right) \times_{j,x} \cos \left( \frac{2n+1}{N} \pi \right), \quad n = 0, 1, \ldots, N - 1
\]

Similarly, \( ST_{j,y} = (\tilde{y}_{j,x}, \tilde{y}_{j,y}, \ldots, \tilde{y}_{j,y}) \), \( ST_{j,z} = (\tilde{z}_{j,x}, \tilde{z}_{j,y}, \ldots, \tilde{z}_{j,y}) \) are obtained. Performing the same to other triangle stripes produces the i-th stego-submesh \( \tilde{M}_i \), that the watermark is embedded into.

According to the bit information of FP, there are two kinds of submeshes; stego- and cover-submeshes in the mesh synthesis process. The stego-submeshes that the watermark is embedded into are generated when their corresponding bits are 1. On the other hand, the cover-submeshes are generated when their corresponding bits are 0. We should be considered to minimize the distortion of the watermark information embedded into submeshes when we make a stego-mesh by combining those cover-submeshes and stego-submeshes. In case that the stego-submeshes are adjacent each other, each value of the sharing vertex coordinates among them is computed by averaging those vertex coordinates. However, in case that the stego-submeshes are adjacent to the cover-submeshes, each value of the sharing vertex coordinates among them is changed into that of the vertex coordinates in the stego-submeshes.

4 Fingerprints Retrieval Process

Fig. 2 shows how to extract fingerprints and detect traitors. A fingerprint embedding process consists of the
marking position selection algorithm, the watermarking algorithm, and the traitor algorithm. The inputs are a cover-mesh, a stego-mesh or possibly attacked stego-mesh, and some parameters that are used in the fingerprint embedding process. Based on it.

**Traitor tracing algorithm**

When we compare \( a' \) obtained from the \( i \)-th stego-submesh with the original watermark \( a \) used for fingerprint embedding process, we set 1 to the fingerprint bit corresponding to the \( i \)-th submesh if \( a' \) and \( a \) are equal. Otherwise, we set 0 to the fingerprint bit. By applying the same procedure to all submeshes, the \( k \times r \) fingerprint bits are extracted. Using the stego-key \( k_i \), those fingerprint bits are rearranged. After checking out the each repeated bit of fingerprints with redundancy \( r \), we set 1 if is exist more than 0s among repeated \( r \) bits for a specific bit in the \( k \times r \) fingerprint bits. Otherwise, we set 0.

Thus the \( k \) bit fingerprint \( FP_{avg} \) is generated from the stego-mesh. By applying Dittmann's traitor detection algorithm [5, 6], we can find traitors up to the maximum number \( d \) with \( FP_{avg} \) obtained from the stego-mesh and \( q+1 \) users' fingerprint information \( FP_1, FP_2, ..., FP_{q+1} \). If redundancy \( r \) has increased, the possibility to detect the correct traitors has also increased.

**5. EXPERIMENTAL RESULTS**

We implemented the proposed collusion-secure fingerprinting scheme using Visual C++ and OpenGL graphics API. Fig. 3 shows 3-D polygonal mesh models for the experiment: Bunny model with 15095 vertices and 30019 faces and Horse model with 15002 vertices and 30000 faces. As a matter of convenience, we simplify original Bunny model obtained from Stanford University and original Horse model obtained from Cyberware Inc.

![Bunny and Horse models](image)

Fig. 3. Test models

For the experiment, we use three fingerprints which are generated based on PG(2,2). Those fingerprints are \( FP_1=110100 \), \( FP_2=001101 \), and \( FP_3=100011 \). As an example of input parameters for Bunny model, we use the following parameter values: \( \nu=15095 \), \( k=6 \), \( r=3 \), \( s=30 \), \( m=6 \), \( c=3 \), \( \alpha=0.0002 \).

In this paper, we employ MESH [16] proposed by EPFL to evaluate the imperceptibility of the proposed fingerprinting scheme by measuring distances between a cover-mesh and a stego-mesh. In MESH, the sampling step \( \delta \) plays a role in the precision of the measured distance. The measure is stable for values of \( \delta \) below.
0.5% or 0.4% of the bounding box diagonal. For the experiment, we choose 0.5 as the value of \( \delta \), and then calculate the symmetric version of the root-mean-square error \( d_{nme} \) by comparing the forward \( d_{nme} \) with the backward \( d_{nme} \). A graph of Fig. 4 shows the distortions of a submesh of Bunny model and Horse model with different values of modulation amplitude \( \alpha \).

\[ \text{Fig. 4. Distortion of a submesh with different } \alpha \]

It is desirable for us to select below 0.0002 as the \( d_{nme} \) value to be imperceptible after embedding the watermark into a cover-mesh. Thus, we reasonably chose \( \alpha = 0.0002 \) for the experiment. In addition, it is recognized that a submesh of Horse model is more sensible than that of Bunny model to \( \alpha \).

As attacks for the proposed scheme, additive random noise, 3-D mesh compression, geometrical transformations and average calculation by two traitors’ collusion are implemented into a fingerprinted stego-mesh. As the additive random noise attack, we added vertex coordinates of the stego-mesh with uniform random noise. The percentage of the additive random noise to the stego-mesh represents the ratio between the largest displacement and the largest size of the stego-mesh’s bounding box [17]. As the 3-D compression attack, we use the MPEG-4 SNHC standard [18]. As the geometrical transformations, we translated, scaled, rotated, or sheared the stego-mesh. As the collusion attack, we took the average of vertex coordinates of 2 users’ stego-meshes.

\[ \text{Fig. 5. Various Attacks to a test model} \]

Fig. 5 shows the attacked Bunny model using (a) the additive random noise, (b) the 3-D mesh compression, (c) the various geometric transformations, and (d) a combined attack with the additive random noise and the 3-D mesh compression.

Table 1 presents the experimental results against various attacks shown in Fig. 4. The fingerprint bit error rate \( BER_f \) is computed by dividing the total number of error bits in the extracted fingerprint information from the stego-mesh by the total number of bits assigned to each embedded fingerprint. Similarly, the watermark bit error rate \( BER_w \) is computed by dividing the total number of error bits in the extracted watermark information from the stego-mesh by the total number of bits embedded into a cover-mesh.

\[ \text{Table 1. Test Results for Bunny model and Horse model} \]

Table 2 presents the experimental results against a collusion attack. Even though Dittmann applied her scheme to still images, her test results are comparable with ours since the type of the collusion attack is both the average calculation.

\[ \text{Table 2. Test Results of a collusion attack} \]

Here, CCR indicates the correct customer recognition in \([5, 6]\). The result shows that we are able to find all traitors. This is due to using a watermarking algorithm robust to the average calculation unlike Dittmann’s watermarking algorithm.

Fig. 6 shows the number of bits assigned to the fingerprints generated based on \( PG(2,q) \) in Dittmann’s scheme and the proposed scheme. While the number of fingerprint bits \( k \) in Dittmann’s scheme increases by \( q^2 + q + 1 \), the number of fingerprint bits \( k' \) in the proposed scheme increases by \( \sum q^i \). Therefore, we can save more bits when \( q \) is much larger.
6. CONCLUSIONS AND FUTURE WORK

We presented a new collusion-secure fingerprinting scheme to embed fingerprints into 3-D polygonal meshes. After we generate the same number of fingerprints as the total number of customers based on the finite projective geometry, we partition the 3-D mesh model related to the total number of bits for each fingerprint and embed the watermark signal into each submesh to be marked. In the watermarking algorithm, copyright information giving a clue to gain suspicious traitors' fingerprints is embedded into mid-frequency DCT coefficients obtained by transforming the vertex coordinates in the triangle strips generated from the selected submeshes to be marked.

Our scheme is robust to additive random noise, MPEG-4 SNHC compression for 3-D mesh vertex coordinates, various geometrical transformations such as rotation, translation, uniform scaling, and shearing, and average calculation by two traitors' collusion. In addition, the number of bits assigned to each fingerprint has been optimized. Using this scheme, we are eventually able to find traitors responsible for the collusion attack causing unauthorized copy and distribution.

In the future, we would like to investigate other attacks including an operation which alters connectivity of meshes, mesh smoothing (i.e., low-pass filtering of 3-D meshes), and resection of a part of the mesh. We will also consider another collusion attack which might be employed in 3-D polygonal meshes effectively.

REFERENCES


