

# A Content-Based Semi-Fragile Mesh Watermarking Scheme

Shwu-Huey Yen, Yu-Ying Chen, and Hwei-Jen Lin

*PRIA Lab., CSIE Department, Tamkang University*

105390@mail.tku.edu.tw, 695410331@s95.tku.edu.tw, hjlin@cs.tku.edu.tw

**Abstract-** *A semi-fragile watermarking scheme based on vertex geometry is presented for 3D model authentication. Watermark is generated from 3D model and random number generator by a seed number. To embed and extract watermark, it only needs the seed number with some simple arithmetic operations. Our method is robust to general 3D model processes, such as uniform scaling, translation, and vertex reordering. The watermark embedding has little impact to the original 3D models that PSNR values can be up to nineties. Thus the proposed method is very suitable for precision instruments or medical science models. Experiments also demonstrate that the method is time and performance satisfactory.*

**Keywords:** Semi-fragile, Watermark, 3D model

## 1. Introduction

In recent years the applications involving digital 3D data, such like 3D games, 3D tourism maps, 3D medical imaging, etc., are rapidly increasing in number and many efforts to obtain these models by direct acquiring or modeling have been deployed. One of reasons that make 3D models popular is that users can zoom in/out, enlarge and/or translate models to study the interested details. Due to the fact of easy duplication and modification of digital contents, it is necessary to develop a variety of watermarking techniques for various protection purposes such as ownership claiming and authentication. In this study, a watermarking scheme is proposed that it not only can ensure the integrity of 3D models but can resist common processes with little impact on appearance of models.

According to purposes, watermarking can be robust or fragile. Robust watermarking schemes are mainly for ownership assertion. As the name suggested, these methods should be immune from malicious or non-malicious attacks. There are studies in [1]-[7] focusing on robust watermarking for 3D models. [1] claims that the method can resist vertex reordering, noise, smoothing, rotation,

simplification, cropping, and combination of above mentioned attacks. [5] reports to resist simplification, noise and cropping attacks. [6] claims to resist translation, rotation and the three attacks mentioned in [5]. These papers presented experiments to support their claims without detailed explanations on how experiments were simulated. Contrary to robustness, fragile watermarking is designed to detect any modification on protected data. There are applications that rigorous data information is very important, such like 3D models for auto parts or medical science, which slight changes in the appearances may lead to unacceptable parts or medical misjudgments. Therefore, those models should be watermarked for verifying the data integrity. Semi-fragile watermarking methods are robust to common processes (non-malicious) but sensitive to malicious attacks. Sometimes, it is difficult to distinguish methods between fragile ones and semi-fragile ones since methods may resist one common process only but not the others. [8]-[12] focus on fragile or semi-fragile watermarking studies. In general, for 3D models, uniform scaling, rotation, etc., are non-malicious attacks and smoothing, simplification, cropping, etc., are malicious attacks.

This paper is organized in the following. In Section 2, related works on 3D watermarking are discussed. In Section 3, the proposed method will be explained. Experiments and analysis are provided in Section 4, and, finally, conclusions and future works are given.

## 2. Related Works

Some robust watermarking methods, such like [1], [5]-[7], would perform registration and/or resampling before extracting watermarks. Registration can recover the original geometric information (location and size). By this way, these methods are robust to translation, rotation, (uniform & non-uniform) scaling, and other geometric attacks. Resampling can make models into a fixed form. Consequently, these methods can resist simplification, cropping, vertex reordering, and

other topological attacks. Although these kinds of processes can enhance the algorithms but they usually need the information of original models and are computation expensive.

In 1999, Yeo et al. [8] proposed the first fragile watermarking scheme for 3D models. They use two hash functions and slightly perturbed the positions of vertices to embed watermarks. The method has two problems: convergence and causality. Since then, improved semi-fragile or fragile methods were proposed to solve these problems. As in [11], mesh parameterization is used to transform vertices into 2D plane and then vertex coordinates are modified to embed watermark. Their method solves the causality but may have loopholes causing detection errors. Chou et al. [9] later proposed a fragile watermarking method as an improved of [8] and [11]. They use multi-function vertex embedding to solve causality and adjusting vertex so that the impact of embedding watermarks to models is limited.

Unlike still images, video sequences, or audio, 3D models lack a structural ordering relation among vertices. Therefore, most of schemes would use some way to order vertices and perform embedding process. Following the same ordering, the watermark can be extracted later. This ordering should be invariant to common models processes. One solution to this problem is by transforming 3D points into another coordinate system that has a structural ordering. In [8], it first evaluates the center of gravity of the model then projects all points into a 2D image. The method in [11] applies cylindrical coordinates to project all vertices to the X-axis. In [9], it uses X-coord. of vertices to find out all “marked” vertices, then uses Y, Z coordinates of those “marked” vertices as embedded features. The method in [3] translates coordinates of vertices to have gravity of the 3D model located in the center of Cartesian space. In [2] and [12], they transform all vertices into spherical coordinates.

### 3. Proposed Scheme

As the goal of protecting the integrity of digital content and robust to common processes, a semi-fragile watermarking scheme is proposed and explained in the following.

#### 3.1. Watermark generating and embedding

To obtain a structural ordering relation among vertices, 3D vertices are projected on XY-plane. Consider a 3D model  $O = \{P, C\}$ , where  $P = \{p_i = (x_i, y_i, z_i), i = 1, \dots, N\}$  is the vertex set, and  $C$  is the connectivity relationship (face) on  $O$ . Watermark

embedding and generating process are depicted in five steps.

#### (1). Calculating the XY-plane centroid $G_{XY}$ and sorting all points of the 3D model.

By projecting all points of the model on XY-plane, the centroid is obtained from Eq.(1). For point  $p_i$ , the distance  $D_i$  is calculated by Eq. (2). Then, points on the 3D model are rearranged in ascending order according to distances  $D_i$ .

$$G_{XY} = (\bar{x}, \bar{y}) \quad \text{where} \quad (\bar{x}, \bar{y}, \bar{z}) = \frac{1}{N} \sum_{i=1}^N p_i \quad (1)$$

$$D_i = \sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2} \quad (2)$$

#### (2). Dividing sorted points into $H$ sections.

Sorted points are divided into  $H$  sections for different watermarks embedding where  $H = \lceil N/k_1 \rceil$ . Since statistics from points within one section are used for embedding/extracting, small  $k_1$  makes statistics vulnerable to outliers and large  $k_1$  makes statistics insensitive to small variations. Besides, any section with points less than  $k_2$  will be merged to one of its neighboring section that has fewer points. In our experiment,  $k_1$  and  $k_2$  are chosen as 100 and 75 heuristically. To be robust to reordering attack, points of the same distance are adjusted to be in the same section.

There are two ways to divide points into sections: *equal section width* and *equal number of points*. The former, each section covers the same width, may have sections with few or many points, and the latter, each section has the same number of points, may have sections with very narrow or very wide ranges. We adopt the combination of these two methods with a weight  $\alpha$  as in Eq.(3). To divide into  $H$  sections, each of these two methods produces  $H-1$  thresholds,  $t_{11}, \dots, t_{1(H-1)}$  and  $t_{21}, \dots, t_{2(H-1)}$ , let  $t_i$  be the final threshold, then

$$t_i = (t_{1i} * \alpha) + [t_{2i} * (1 - \alpha)], \quad 1 \leq i \leq H - 1 \quad (3)$$

with  $t_0 = \min\{D_i\}$  &  $t_H = \max\{D_i\}$ ,  $i = 1, 2, \dots, N$ , point  $p_i$  is in section  $k$  ( $\geq 2$ ) if  $t_{k-1} < D_i \leq t_k$ , or  $k=1$  if  $t_0 \leq D_i \leq t_1$ . In our experiment,  $\alpha = 0.5$ .

#### (3). Watermark generating and embedding for each section.

After all the points are reordered and divided into  $H$  sections, the watermarks for section  $k$  will be generated and embedded,  $k = 1, 2, \dots, H$ :

- With a meaningful number as the seed number, such as the owner's ID number, a decimal digit sequence of length  $3H$  is generated by random number generator. The three digits  $d_1, d_2, d_3$  positioned at  $3k, 3k+1, 3k+2$  of the sequence are

to be composed into 4-digit watermarks for section  $k$  in the next step.

- Consider points in section  $k$ , we first normalize Z-coord. of these points to be zero mean and unit variance by Eq.(4)-(6). Further divide the section into three subsections by the ratio of 3:4:3. Upon each subsection, extract 4-digit numbers from the 4<sup>th</sup> to 7<sup>th</sup> decimal places of normalized Z-coord. of every point. For example, 3368 is extracted from 1.392336821. Let  $d_{01}$ ,  $d_{02}$ ,  $d_{03}$  be the thousands place of the average of these 4-digit numbers from three subsections. The watermarks  $w_{k1} = d_{01}d_1d_2d_3$ ,  $w_{k2} = d_{02}d_1d_2d_3$ ,  $w_{k3} = d_{03}d_1d_2d_3$  are consequently generated for three subsections of the section  $k$ .
- For each point  $p$  in the subsection  $i$  of section  $k$ , the 4-digit watermark  $w_{ki}$  is to be embedded into  $p$ 's normalized Z-coord.. Embedding is simply to replace the 4<sup>th</sup> to 7<sup>th</sup> decimal places of normalized Z-coord. of  $p$  by  $w_{ki}$ .

$$\mu_k = \sum_{i=0}^{N_k-1} z_i / N_k, \quad (4)$$

$$\sigma_k = \sqrt{\sum_{i=0}^{N_k-1} (z_i - \mu_k)^2 / N_k}, \quad (5)$$

$$z'_i = \frac{z_i - \mu_k}{\sigma_k}. \quad (6)$$

Take *Aardvark* as an example,  $|P|=264$ ,  $|C|=500$ , and  $\lceil N/k_i \rceil = \lceil 264/100 \rceil = 3$ . After merging & adjusting, *Aardvark* is divided into  $H=2$  sections. Using a random number generator with a seed number, a decimal digit sequence of size  $3H = 6$  is generated, say, 346130, where 346 is for section 1 and 130 is for section 2. As for  $k=1$ , the normalized Z-coord. of points are divided into three subsections. And the average of those extracted 4-digit numbers are 4548.7, 4487.1, and 4439. Thus  $d_{01}=4$ ,  $d_{02}=4$  and  $d_{03}=4$ . With the corresponding three digits  $d_1d_2d_3 = 346$ , the watermarks are  $w_{k1} = w_{k2} = w_{k3} = 4346$ . Table 1 shows the embedding result where the 4<sup>th</sup> to 7<sup>th</sup> decimal places of normalized Z-coord. now is replaced by the watermark.

#### (4). Compensation.

Let  $z_i$ ,  $z'_i$ ,  $z''_i$  be the Z-coord. of point  $p_i$  original, after normalized, and after watermarked. Now  $z''_i$  should be reversed by Eq.(7) to obtain the watermarked model where  $\mu_k$  and  $\sigma_k$  are (4), (5).

$$\hat{z}_i = z''_i \cdot \sigma_k + \mu_k. \quad (7)$$

Unlike  $z'$ , the distribution of  $z''$  is neither zero mean nor unit variance anymore. Since keeping the watermarked  $\hat{z}$  as similar to the original  $z$  as

possible is essential to correctly extract the watermarks later, thus, 10% of points from the top and bottom in section  $k$  are chosen respectively to make  $z''$  to be zero mean. Let  $\mu_{k,embedded}$  be the average of  $z''$  in the section  $k$  of total  $N_k$  points and  $m = \lfloor N_k \cdot 10\% \rfloor$  the number of points that is 10% of points in section  $k$ , then for  $i=1, 2, \dots, N_k$ ,

$$z''_i = \begin{cases} z''_i - \delta & \text{if } i \leq m \text{ or } i > N_k - m, \\ z''_i & m < i \leq N_k - m, \end{cases} \quad (8)$$

where  $\delta = (\mu_{k,embedded} \cdot N_k) / 2m$ . Table 2 is the compensation result of *Aardvark* for section one which has 130 points. The shaded cells are compensated vertices ( $m=13$ ). As in the table, the average of  $z''$ ,  $\mu_{k,embedded}$ , is 4.04738 E-5, and, after compensation, the average of  $z'''$  is -8.54018 E-18.

#### (5). Obtaining the watermarked model.

Substituting  $z'''$  for  $z''$  in Eq.(7) to reverse the normalization process as in (9), the watermarked model is obtained.

$$\hat{z}_i = z'''_i \cdot \sigma_k + \mu_k. \quad (9)$$

**Table 1.** Watermark embedding for *Aardvark* ( $k=1$ )

3D model - <i>Aardvark</i> watermark embedding					
1 <sup>th</sup> -subsection		2 <sup>th</sup> -subsection		3 <sup>th</sup> -subsection	
Normalization	Embedding	Normalization	Embedding	Normalization	Embedding
$z'$	$z''$	$z'$	$z''$	$z'$	$z''$
1.392336821	1.3924346	0.738737695	0.7384346	-0.04158463	-0.0414346
-1.375404624	-1.3754346	-0.786558529	-0.7864346	0.105422598	0.1054346
⋮	⋮	⋮	⋮	⋮	⋮
-0.731558005	-0.7314346	-1.629717863	-1.6294346	-1.27325839	-1.2734346
0.704162146	0.7044346	-0.728431516	-0.7284346	1.711527886	1.7114346
0.959102973	0.9594346	1.143205366	1.1434346	1.139541688	1.1394346
⋮	⋮	⋮	⋮	⋮	⋮
0.193003867	0.1934346	1.086404156	1.0864346	-1.093030034	-1.0934346
-0.228395906	-0.2284346	-1.053622665	-1.0534346	-1.079737113	-1.0794346

**Table 2.** The compensation for *Aardvark* ( $k=1$ )

3D model - the compensation result of <i>Aardvark</i>					
1 <sup>th</sup> -subsection		2 <sup>th</sup> -subsection		3 <sup>th</sup> -subsection	
$z''$	$z'''$	$z''$	$z'''$	$z''$	$z'''$
1.3924346	1.392232231	0.7384346	0.7384346	-0.0414346	-0.0414346
-1.3754346	-1.375636969	-0.7864346	-0.7864346	⋮	⋮
0.9354346	0.935232231	-0.9974346	-0.9974346	-1.0474346	-1.0474346
-0.9174346	-0.917636969	0.9604346	0.9604346	-1.1624346	-1.1624346
-0.0174346	-0.017636969	-0.4984346	-0.4984346	⋮	⋮
⋮	⋮	⋮	⋮	0.7974346	0.797232231
0.9124346	0.9124346	0.8644346	0.8644346	-1.6524346	-1.652636969
-0.9034346	-0.9034346	0.8614346	0.8614346	-0.7624346	-0.762636969
⋮	⋮	⋮	⋮	-1.0934346	-1.093636969
-0.2284346	-0.2284346	-1.0534346	-1.0534346	-1.0794346	-1.079636969
Average of $z''$ : 0.0000404738		Average of $z'''$ : -8.54018E-18			

### 3.2. Watermark extracting

The extracting process is similar to the embedding process. When an authorized user receives a watermarked model with information of the seed number, he follows the steps 1, 2 in the embedding phase to divide points into  $H$  sections. For each section  $k$ , three 4-digit watermarks  $w_{k1}, w_{k2}, w_{k3}$  (hypothetic watermarks) are generated as in step 3. The extracted watermarks are from the 4<sup>th</sup> to 7<sup>th</sup> decimal places of normalized Z-coord.. Let  $w_i$  and  $\hat{w}_i$  be the hypothetic and extracted 4-digit watermarks corresponding to point  $p_i$ . Theoretically, these two values should be very similar if the watermarked model is not attacked. Therefore, by observing the differences of  $w_i$  and  $\hat{w}_i$  we can determine whether the model is attacked. Since the calculations described above are floating-point operations, it is common that the results have small errors on decimals and possibly cause differences in carrying digits. Thus, if  $d_i$  is the difference of  $w_i$  and  $\hat{w}_i$  for point  $p_i$ , then

$$d_i = \min(|w_i - \hat{w}_i|, 10000 - |w_i - \hat{w}_i|). \quad (10)$$

Let  $\bar{d}_k$ , average of all  $d_i$ , be the extracted result for section  $k$ . When  $\bar{d}_k$  is less than or equal to the threshold, we conclude that points in section  $k$  are authentic. Note that those compensated points are ignored in evaluating  $\bar{d}_k$  since they have no watermark information anymore. Table 3 shows the extracted result for  $k=1$ . The first two columns are the hypothetic  $w_i$  and the extracted  $\hat{w}_i$ ; the last column is  $d_i$ . Finally,  $\bar{d}_1$  is 40.3.

**Table 3.** *Aardvark* extracted result for section one

Hypothetic	Extracted	Differences		
		$ w_i - \hat{w}_i $	$10000 -  w_i - \hat{w}_i $	$d_i$
4346	2386	1960	8040	
4346	6433	2087	7913	
...	...	...	...	...
4346	4396	50	9950	50
4346	4394	48	9952	48
4346	4347	1	9999	1
...	...	...	...	...
4346	6420	2074	7926	
4346	6419	2073	7927	
Extracted result $\bar{d}_1 =$				40.3

## 4. Experiments and Discussions



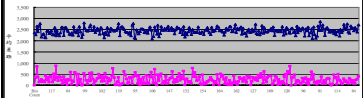


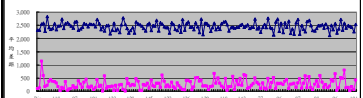


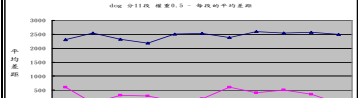
43 models from simple (*Ball*,  $|P|=34$  &  $|C|=64$ ; *Tree*,  $|P|=56$  &  $|C|=104$ ) to complicated (*Horse*,  $|P|=19,851$  &  $|C|=39,698$ ; *Bunny*,  $|P|=35,947$  &  $|C|=69,451$ ) are tested on the proposed scheme.

### 4.1. The perceptual quality and validity

In table 4, we show original and watermarked models, and extracted results of both for *Bunny* ( $H=290$ ), *Horse* ( $H=172$ ), *Dog* ( $H=11$ ). For perceptual quality, before and after watermarking of the model are compared. Apparently there is no difference visually. For validity, the extracted results  $\bar{d}_k$  from un-watermarked models, in blue, and watermarked models, in pink, are shown in the table. Figures in horizontal axis are number of points in section 1, 2, ...,  $H$ , and vertical axis shows the average difference on that section, i.e.,  $\bar{d}_k$  for section  $k$ . We can easily distinguish whether the model is watermarked from the distributions of these two lines.

When extracting watermarks from an un-watermarked model, the extracted  $\hat{w}_i$ , the 4<sup>th</sup> to 7<sup>th</sup> decimal places of normalized Z-coord., is a random value among 0000 to 9999. From uniform distribution,  $\mu$  (mean) and  $\sigma$  (standard deviation) are about 5000 and 2887. Although the first digit of the hypothetic watermark  $w_i$  is from the average of the 4-digit numbers on normalized Z-coord., many first digits in  $w_i$  and  $\hat{w}_i$  are still different due to a very high standard deviation. Thus, we can assume  $d_i$  in (10) is approximately distributed uniformly in  $0 \sim 5000$  with  $\mu \approx 2500$  and  $\sigma \approx 1444$ . By Central Limit Theorem,  $\bar{d}_k$  is approximately Gaussian with  $\mu \approx 2500$  and  $\sigma \approx 1444/\sqrt{N_k}$ . We can see that the average differences for those blue lines are around 2500 conforming to the analysis. On the other hand, when extracting watermarks from a watermarked model, most of first digits in  $w_i$  and  $\hat{w}_i$  are the same. Thus,  $d_i$  is approximately distributed uniformly in  $0 \sim 500$ , consequently,  $\bar{d}_k$  is approximately Gaussian with  $\mu \approx 250$  and  $\sigma \approx 289/\sqrt{N_k}$ . As shown on table 4, the average differences for those pink lines are around 250. To determine the integrity of the model, we take  $[0.5*(t_w + t_w)]$  as the threshold where  $t_w$  is two standard deviations below the mean of  $\bar{d}_k$  for un-watermarked models and  $t_w$  is two standard deviations above the mean of  $\bar{d}_k$  for watermarked models. Taking  $N_k$  to be 100, the threshold is  $[0.5*(2211.2+307.8)] = [1259.5] = 1260$ . Whenever  $\bar{d}_k$  is not greater than 1260, we conclude that points on section  $k$  are authentic such like points on section one of *Aardvark* (table 3).

**Table 4.** Before & after watermarked

before	after	Extracted results: before(blue) & after(pink)
		
		
		

#### 4.2. Attacks simulations




A series of non-malicious attacks are applied on watermarked *Aardvark* ( $H=2$ ) uniformly scaled by 2, & uniformly scaled by 1.5 then translated by 1 unit. As the extracted results summarized in table 5, the proposed scheme is not affected by translation and uniform scaling due to the normalization process in embedding and extracting phases.

Partially enlarged attack is also applied on watermarked *Aardvark* where table 6 shows the (a) original, (b) attacked model such that its nose is enlarged 1.2 times (27 points are involved), and their corresponding extracted results. As in (b), both  $\bar{d}_1$  &  $\bar{d}_2$  yield unacceptable results ( $>1260$ ) since these 27 points are scattered in both sections. We can conclude that the model is attacked even though it is hard to notice from its appearance.

**Table 5.** Extracted results for *Aardvark* undergoes uniform scaling  $\beta$  & translation  $\gamma$

Extracted results, i.e., $\bar{d}_k$ , for <i>Aardvark</i>				
		$(\beta, \gamma)$		
(section) $k$	No. of pts	(0,0)	(2,0)	(1.5,1)
1	130	40.8942	40.8942	40.8942
2	134	340.5648	340.5648	340.5648

**Table 6.** Extracted result on partially enlargement

	(a) No attack	(b) Nose enlarged
		
$k=1$	40.8942	2335.7905
$k=2$	340.5648	2409.8598

Resampling attack (*attack a*) is simulated on *Dog* ( $H=11$ ). The simulation is done by merging two points (no.127 & no.128) into one (no.128) which

causes little difference in appearance. Extracted results are in table 7. As rows indicated *attack a* shown, results on sections 3~6 (shaded cells) are detected as having been affected while other sections are not affected. Although only one point is merged, it causes different partition in  $H$  sections and consequently neighboring sections may also be influenced. Finally, the reordering attack (*attack b*) is applied on *Dog*. The simulation is done by switching two points (no.10 & no.20) and their corresponding face information. Notice that although the ordering of points has been switched, the model is not modified. Therefore, our algorithm should confirm that the model is authentic. In table 7, the rows indicated *attack b* exhibit this fact.

**Table 7.** Extracted results: *Dog* on resampling (*Attack a*) and reordering (*Attack b*)

(section) $k$	1	2	3	4	5	6
Noattack	99.5	293.2	599.9	597.5	402.2	349
<i>Attack a</i>	99.5	293.2	1964	2385	2629	2546
<i>Attack b</i>	99.5	293.2	599.9	597.5	402.2	349
(section) $k$	7	8	9	10	11	
No attack	21.6	187	33.2	505.6	308.2	
<i>Attack a</i>	21.6	187	33.2	505.6	308.2	
<i>Attack b</i>	21.6	187	33.2	505.6	308.2	

#### 4.3. Quality measurements

To measure the impact of watermarking on models, we adopt two formulas from [3] and [13]. Authors in [3] used PSNR, Eq. (11), to evaluate the transparency of 3D watermarked models.

$$PSNR = 10 \log_{10} \frac{N * \max I_n^2}{\sum_n (I_n - I'_n)^2}, \quad (11)$$

where  $N$  is total number of points in the model,  $I_n$  and  $I'_n$  are geometric information on point  $n$  before and after watermarked. Table 8 lists PSNR values of our method and the method in [3]. As it is shown, the PSNR values are very satisfying for our method.

Authors in [13] use MSE, Eq. (12), to measure the quality of the watermarked models.

$$MSE = \frac{1}{N} \sum_{i=1}^N \|v_i - \hat{v}_i\|^2, \quad (12)$$

where  $N$  is total number of points for the model,  $v_i$  and  $v'_i$  are geometric information on point  $i$  before and after watermarked. Table 9 shows MSE values. Although we do not have *Screwdriver*, comparing those values, there is an impressive difference in those values. From both tables, our

method has little impact on the appearances of models; therefore, the method is very suitable for rigorous instrument models which both integrity and precision are very important.

Time consumption is summarized in table 10 when testing environment is Pentium 4 CPU, 3.20GHz, and 512 MB memory. As illustrated, the proposed method is time efficient that even the complicated model *Bunny* of 35,947 vertices & 290 sections takes about 4.5 & 3.5 seconds to complete the embedding and extracting.

**Table 8.** PSNR values

Models	<i>Horse</i>	<i>Bunny</i>	<i>Dog</i>	<i>Aardvark</i>
Our method	<b>88.3738</b>	<b>83.7608</b>	<b>91.5727</b>	<b>90.7543</b>
Method in [8]	38.87	39.78	N/A	N/A

**Table 9.** MSE values

Models	<i>Horse</i>	<i>Bunny</i>	<i>Dog</i>	<i>Aardvark</i>	<i>Screwdriver</i>
Our method	<b>4.22E-10</b>	<b>1.56E-09</b>	<b>2.50E-10</b>	<b>3.03E-10</b>	N/A
Method [12]	N/A	N/A	4.98E-03	N/A	3.51E-05

**Table 10.** Time consumption (in millisecond)

Models ( <i>N,H</i> )	<i>Horse</i> (19851,172)	<i>Bunny</i> (35947,290)	<i>Dog</i> (1871,11)	<i>Aardvark</i> (264,2)
Embed	2453	4438	219	31
Extract	1860	3594	172	16

## 5. Conclusions and Future Works

This paper proposed a semi-fragile watermarking method for 3D models to verify the integrity of models. Our scheme is robust to uniform scaling, translation, and also robust to vertex reordering attack. The method can detect malicious attacks like partially enlarging, resampling. The receiver only needs the seed number to verify the integrity of 3 D model. Because watermarks are embedded in the normalized Z coord. of the 4<sup>th</sup> to 7<sup>th</sup> decimal places, the impact on a model is very small, so this method is suitable for rigorous equipment or medical 3D model.

Our method can find local modification region, however, it could not indicate which points have been attacked exactly. In addition, instead of projecting all points on XY-plane, PCA is a good choice to achieve the same goal with rotation invariance. In the future, we would like to enhance the method according to the abovementioned.

## References

- [1] E. Praun, H. Hoppe, and A. Finkelstein, "Robust mesh watermarking," *Proc. of Int'l Conf. on Computer Graphics and Interactive Techniques*, pp. 49-56, 1999.
- [2] J.-W. Cho, R. Prost, and H.-Y. Jung, "An Oblivious Watermarking for 3-D Polygonal Meshes Using Distribution of Vertex Norms," *IEEE transactions on signal processing*, V.55, No.1, pp. 142-155, Jan. 2007.
- [3] J. Shu, Y. Qi, S. Cai, and X. Shen, "A Novel Blind Robust Digital Watermarking on 3D Meshes," *Proc. of the 2<sup>nd</sup> workshop on Digital Media and its Application in Museum & Heritages*, pp. 25-31, Dec. 2007.
- [4] B. Oliver, "Geometry-Based Watermarking of 3D Models," *IEEE Computer Graphics and Applications*, V.19, Issue. 1, pp. 46-55, 1999.
- [5] Z.Q. Yu, H.H.S. Ip, and L.F. Kwok, "Robust Watermarking of 3D Polygonal Models Based on Vertex Scrambling" *Computer Graphics International*, pp. 254-257, July 2003.
- [6] Z.Q. Yu, H.H.S. Ip, and L.F. Kwok, "A robust watermarking scheme for 3D triangular mesh models," *Pattern Recognition*, V.36, Issue 11, pp. 2603-2614, Nov. 2003.
- [7] Z. Li, W.-M. Zheng, and Z.-M. Lu, "A Robust Geometry-Based Watermarking Scheme for 3D Meshes," *First Int'l Conf. on Innovative Computing, Information and Control*, V.1, pp. 253-256, Aug. 2006.
- [8] B.-L. Yeo and M.M. Yeung, "Watermarking 3D objects for verification," *IEEE Computer Graphics and Applications*, V.19, Issue 1, pp. 36-45, 1999.
- [9] C.-M. Chou and D.-C. Tseng, "A public fragile watermarking scheme for 3D models authentication," *IEEE Computer-Aided Design*, V.38, Issue 11, pp. 1154-1165, Nov. 2006.
- [10] H.-T. Wu and Y.-M. Cheung, "A fragile watermarking scheme for 3D meshes," *ACM Proc. of the 7th workshop on Multimedia and Security*, pp. 117-124, Aug. 2005.
- [11] H.-Y.S. Lin, H.-Y.M. Liao, C.-S. Lu, and J.-C. Lin, "Fragile Watermarking for Authenticating 3-D Polygonal Meshes," *IEEE Transactions On Multimedia*, V.7, No.6, pp. 997-1006, Dec. 2005.
- [12] W. Liu and S.-H. Sun, "Rotation, Scaling and Translation Invariant Blind Digital Watermarking for 3D Mesh Models," *First Int'l Conf. on Innovative Computing, Information and Control*, V.3, pp. 463-466, Aug. 2006.
- [13] T. Harte and A.G. Bors, "Watermarking 3D models," *Proc. of IEEE Int'l Conf. on Image Processing*, V.3, pp. 661-664, Jun. 2002.