Abstract—Three robust blind watermarking methods of 3D models based on Octree are proposed in this paper: OTC-W, OTP-W and Zero-W. Primary Component Analysis and Octree partition are used on 3D meshes. A scrambled binary image for OTC-W and a scrambled RGB image for OTP-W are separately embedded adaptively into the single child nodes at the bottom level of Octree structure. The watermark can be extracted without the original image and 3D model. Those two methods have high embedding capacity for 3D meshes. Meanwhile, they are robust against geometric transformation (like translation, rotation, uniform scaling) and vertex reordering attacks. For Zero-W, higher nodes of Octree are used to construct ‘Zero-watermark’, which can resist simplification, noise and remeshing attacks. All those three methods are fit for 3D point cloud data and arbitrary 3D meshes.

Index Terms—Primary Component Analysis, Octree, robust, watermarking

I. INTRODUCTION

Nowadays, many kinds of information including text, image, audio, video and 3D model can be transmitted and published conveniently through Internet. Meanwhile, the dangers of copying, tampering, or transmitting copyrighted data without authorization have brought about an increasing demand for robust copyright protection methods. Watermarking technology, originating from the digital steganography, is becoming one of the main methods for the copyright protection of digital works.

Many publications on watermarking have been concentrated on various media data types such as text, image, audio, and video. With the fast development of 3D scanner and graphics card, 3D modeling and display technology have become increasingly efficient. Furthermore, the universal popularity of 3D online games has led to the widespread use of 3D models in various applications such as video gaming, engineering design, architectural walkthrough, virtual reality, e-commerce and scientific visualization. Therefore, the watermarking of 3D models has gained more attention in recent years. Watermarking algorithms show a mutual improvement on audio, image, and video, which are regularly sampled signals. However, it is still very challenging to extend these known watermarking algorithms into irregularly sampled 3D models.

II. RELATED WORK

Watermarking approaches can be categorized into fragile watermarking [1-4] and robust watermarking [5-10] according to their respective objectives. The main purpose of the former is to detect slight changes for authenticating the integrity of digital content. In contrast, the latter is to hide a watermark in digital contents in an imperceptible way in order to withstand various malicious attacks. Robustness and invisibility are the main requirements of a robust watermarking algorithm. Our methods are concerned with the robust watermarking issue for 3D models. On the other hand, among various representations of 3D data, meshes provide an effective means on which many watermarking algorithms are used. As a result, we focus on the watermarking algorithms on this majority representation of 3D models in this paper. More information about other representations can be found in [20] for point data, [21] for 3D Non-Uniform Rational Basis Spline (NURBS) data, and [22] for 3D models with texture data.

Generally, according to the embedding domain of watermark, algorithms on 3D meshes include the following two parts: spatial domain method and frequency domain method. Ohbuchi proposed TSQ and TVR spatial algorithms for watermarking in [5] which can be robust against translation, rotation and uniform scaling attacks. However, it fails after noise and topology attacks. In [6], a novel watermarking method which can be robust against simplification attacks with informed detection is proposed. Watermark is embedded based on changing discrete normals of vertexes in [7], which can resist both translation and uniform scaling attacks, but is sensible with noise and topology attacks. In transform domain watermarking, Ohbuchi presented an informed-detection robust algorithm that embeds message bits by deforming the “low-frequency” components of the shape by using the mesh spectral analysis [8]. However, the meshes must be regular, and original model is needed in the detection process. A watermarking method using
Laplace operator with blind detection is proposed in [9], but failed under topology changed. In [10], watermark is embedded using spread spectrum watermarking techniques. The algorithm is robust against mesh smoothing, random noise addition, and mesh simplification using informed detection.

Among above methods, the spectral algorithms are lossy codecs that either aggressively truncate high-frequency geometric information or perform a remeshing on the input manifold model to achieve high efficiency. However, they require complicated computation and are only fit for manifold mesh. This paper does not consider the frequency domain method. In spatial domain methods, space partition based on Kdtree and Octree is only fit for manifold mesh. This paper does not consider the frequency domain method. In spatial domain methods, space partition based on Kdtree and Octree is only fit for manifold mesh. This paper does not consider

Watermarking algorithms can also be separated into blind detection and informed detection methods according to the detection procedure. Original 3D model is needed in informed watermarking detection while it is not needed in blind watermarking. Majority of early research on mesh watermarking method are focused on the informed detection. Recently, blind watermarking is the major research issue. The advantages of blind watermarking can be found in [11]. Many blind watermarking methods can only be robust against basic attacks, such as rotation, translation and scaling, but fail under remeshing and clipping attacks.

In this paper, three novel blind spatial watermarking methods including three algorithms (OTP-W, OTC-W and Zero-W) for arbitrary 3D meshes based on Octree is proposed. OTP-W and OTC-W have a large embedding space to embed a binary image and a RGB image, which are robust against translation, rotation, uniform scaling and vertex reordering attacks with blind detection. Zero-watermark is constructed based on Octree in Zero-W, which can resist attacks of simplification, noise and remeshing. Detailed information about Zero-watermark is available in [12].

Our methods proposed in this paper mainly based on the following two ideas: lossless Octree coding theory proposed in [13] and PCA (Primary Component Analysis) preprocess method for 3D meshes mentioned in [14]. The PCA preprocess is used to ensure the 3D meshes have a unique posture, which is a prerequisite before embedding and blind detection. In the Octree partition process, nodes in the bottom level of Octree represent the details of 3D meshes, where watermarks are embedded in OTC-W and OTP-W. Nodes in higher level of Octree indicate the position distribution of 3D meshes’ vertices in low resolution. ‘Zero-watermark’, considered as the unique feature of 3D meshes to some extent, can be constructed based on the nodes of higher level of Octree. ‘Zero-watermark’ is registered in a Third Party Certificate Organization after compression which reduces the capacity of ‘Zero-watermark’ within 10 KB. In the detection process, original 3D model is not needed, as we just reconstruct the ‘Zero-watermark’ to compare with the one got from the Third Party Certificate Organization, and the copyright can be achieved according to the threshold.

Section III illustrates the preprocess of Octree partition and PCA. The implementation of watermark embedding and extraction algorithms will be given in Section IV. After that, we will discuss the experimental results in Section V. Finally, a conclusion will be made in section VI.

III. OCTREE PARTITION AND PCA PROCESS FOR OPT-W

Fig. 1 is the process of embedding, extraction and detecting watermark.

![Figure 1: Process of embedding and detecting watermark](image)

A. Octree partition

The Octree partition rule is that the model is subdivided recursively until there is no vertex in the subspace or the assumed precision is reached, as shown in Fig. 2.

![Figure 2: Octree partition](image)

Nodes at the bottom level of Octree structure represent the details of 3D meshes. Tiny disturbance of vertices’ coordinates is made in a very small space while embedding the watermark into these Nodes. The embedding process changes little of the 3D meshes’ shape. Thus the Octree partition in the detection process...
is quite the same as the one in the embedding process, which ensures that the watermark can be detected correctly without original 3D meshes even after translation, rotation, uniform scaling or vertex reordering attacks.

3D meshes are moved to the origin of Cartesian coordinate space. The origin is the center of bounding box of 3D meshes. The bounding box size can be calculated by the maximal distance of all vertices to origin. Then a top-down Octree partition is made. The partition uses uniform axial subdivision, and the size of each new child’s bounding box is 1/2 size of its father’s bounding box. The partition stops at the level specified.

Cai proposed a geometric-driven compression method of encoding/decoding 3D meshes based on Octree in [13]. Vertices of 3D meshes are re-classified to 12 levels according to the Octree rule. All the nodes of the Octree are statistically analyzed to identify the type of nodes with the max proportion and encode them with fewer bits. The nodes with only one non-empty child node play a dominant role in all of the nodes, which is defined as M nodes. 8 bits are used to encode the Non-M node. For M node, there are only 8 cases of distribution for its single child, so M node only needs 3 bits to be encoded. In our watermarking method, original coordinates of vertices are replaced with the center coordinates of the bounding box based on lossless compression encoding method.

B. PCA adjusting model posture

Octree partition is invariant to both uniform scaling and translation, but sensible to rotation of 3D meshes. In order to detect watermark after rotation attack, the model must be adjusted automatically to a unique posture. PCA (Primary Component Analysis) preprocess method for 3D meshes are given in [14], which is used to ensure the 3D meshes have the unique posture before embedding and detection.

3D meshes are comprised of a set of vertices \( V \) and a set of connections between these vertices. Each vertex in \( V \) can be represented as \( v_i = \{x_i, y_i, z_i\} \). First, the mass center \( K \) of \( V \) is calculated, as in (1), where \( N \) is the total number of vertices.

\[
K = \frac{1}{N} \sum_i v_i 
\]  

(1)

Then 3D meshes are translated so that the center \( K \) can fall on the center of Cartesian space. Each element \( v'_i = \{x'_i, y'_i, z'_i\} \) in the set \( V' \) after translation can be computed as in (2).

\[
\begin{align*}
x'_i &= x_i - k_x \\
y'_i &= y_i - k_y \\
z'_i &= z_i - k_z 
\end{align*}
\]

(2)

Finally, we calculate the principal component eigenvector of covariance matrix \( C \) of vertex coordinates after translation, as in (3).

\[
C = \begin{bmatrix} \sum_{i=0}^{N} x'_i x'_i & \sum_{i=0}^{N} x'_i y'_i & \sum_{i=0}^{N} x'_i z'_i \\ \sum_{i=0}^{N} y'_i x'_i & \sum_{i=0}^{N} y'_i y'_i & \sum_{i=0}^{N} y'_i z'_i \\ \sum_{i=0}^{N} z'_i x'_i & \sum_{i=0}^{N} z'_i y'_i & \sum_{i=0}^{N} z'_i z'_i \end{bmatrix}
\]

(3)

Three mutually orthogonal eigenvectors: \( \beta_1, \beta_2, \beta_3 \) can be produced by arranging three eigenvalues \( \lambda_1, \lambda_2, \lambda_3 \) of matrix \( C \) in a descent order. The first rotation matrix \( M_1 \) can be obtained by making \( \beta_1 \) coincide with \( Y \) axis in Cartesian coordinate space, and then \( \beta_2 \) will be changed to \( \beta_2' \) as the rotation happens. We make \( \beta_2' \) coincide with \( X \) axis to get the second rotation matrix \( M_2 \). As the three eigenvectors are mutually orthogonal, \( \beta_2' \) must be aligned to \( Z \) axis. After that, the final rotation matrix \( M' = M_1 \times M_2 \) will be obtained. The 3D meshes will be transformed to a unique posture according to the matrix \( M' \). \( M' \) is saved since the 3D meshes need to be set back to the original posture after watermark embedding.

IV. WATERMARK EMBEDDING AND EXTRACTION

A. OTC-W Algorithm

(1) OTC-W embedding

In the OTC-W method, a binary image with proprietary right information is chosen as the watermark. M nodes of the bottom level of Octree are served as the watermark host.

For a binary image, its size can be calculated according to \( S = \text{width} \times \text{height} \). There are only two values for each pixel of the image: 0 or 1. The watermark is embedded into the encoded bit-stream data of M nodes encoded with 3 bits, which is an embedding unit. We get \( S' \) by complementing the size \( S \) with 0 to be the integral multiple of 3. In order to scramble the watermark before embedding, \( S' \) is segmented as \( S' = (3^m \times n) \) following the row order, where \( m \) and \( n \) are both integers (\( n \) is the number of groups of watermark, and \( m \) is the unit number of each group). The process of scrambling is performed on the indexes array of groups \( K = \{1,2,...,n\} \) using pseudo random algorithm, so new indexes array of groups \( K' = \{a_1,a_2,...,a_n\} \) and scrambled data \( D = \{d_1,d_2,...,d_n\} \) can be obtained. \( K' \), width and height are saved in the key information for the watermark detection process. The process of OTC-W embedding is shown in Algorithm1.

Algorithm 1: OTC-W embedding

1. build nodeList containing all the M nodes at the bottom level of octree;
2. init nodeCount=0;
3. for all \( d_i \in D \) for all \( m_j,m_j,m_j \in d_i \)
4. replace code of nodeList[nodeCount] with \( m_j,m_j,m_j \);
5. nodeCount++;
6. end for
7. end for
8. for all nodei \( \in \text{nodeList} \)
9. change the vertex coordinates according to the newcode
10. end for
Table I shows the mapping relationship between code bits and childnode index in Octree.

<table>
<thead>
<tr>
<th>Code bits</th>
<th>Childnode index</th>
<th>Space index of childnode (see Fig. 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>001</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>010</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>011</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>100</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>101</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>110</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>111</td>
<td>7</td>
<td>4</td>
</tr>
</tbody>
</table>

For example, if the original code bits of an M node are 001, it indicates the coordinate of vertex is the center of bounding box which belongs to the child node labeled 1. If the code data has been replaced with watermark code 110, then the coordinate will be changed to the center of bounding box which belongs to child node labeled 6.

(2) OTC-W extraction

During the watermark extraction process, the PCA preprocess described in Section III.B is performed on 3D meshes at first, making the posture of 3D model to be the same as the unique posture in the embedding. The scrambling key and watermark image size saved in embedding process are also used. The extraction algorithm is shown as follows:

Algorithm 2: OTC-W extraction
1. initialize the watermark image WM[width][height]
2. for all M node, ∈ the bottom level
3. if WM has not all been filled full
4. Get code data m, m1, m2 of M node,;
5. Fill WM with m, m1, m2 according to the scrambling information;
6. else
7. return
8. end if
9. end for

(3) Analysis of capacity of OTC-W

M nodes take a large proportion of the total amount of nodes at the bottom level. As Table II shows, the number of M nodes at the bottom level is almost equal to the number of vertices of 3D meshes. 3 bits of watermark information can be embedded into an M node, then we can figure out an estimation of embedding capacity: 3 * nv bits, where nv is the total number of vertices. It is a large embedding space for OTC-W.

B. OTP-W Algorithm

(1) OTP-W embedding

OTP-W algorithm only uses Octree partition, which is different from the method combined with Octree codec in OTC-W Algorithm. The final published model is not the private format, but the general format, e.g. obj or wrl file. And the watermark is extended from binary image to gray image or RGB image with more information.

We define di as the average vectorial error distance between the vertices included in current child node and the center point of the bounding box, as shown in (4).

\[ d_i = \frac{1}{n} \sum_{j=1}^{n} d(j, \text{center}) \quad (i \in \{x, y, z\}) \quad (4) \]

\( d(j, \text{center}) \) is the vectorial distance between vertex \( j \) and the center point. The signed value of \( d_i \) decides how to embed the watermark. Assuming the watermark is \( w \), indicating the scrambled values of R, G and B channels. Embedding intensity factor is \( a \). We adjust \( a \) to be small enough to ensure perturbation value \( a \times w \) a very small one, which meets the need of watermark transparency. The process of the watermark embedding is shown in Algorithm 3.

Algorithm 3: OTP-W embedding
1. for each dimension \( i \)
2. if \( (d_i < 0) \)
3. \( v_i = v_{\text{center}}, i^\alpha w \)
4. else
5. \( v_i = v_{\text{center}}, i^\alpha w \)

At the bottom level space of Octree structure, the distance between the center vertices coordinates and the actual vertices coordinates is small. The superimposed values of watermark signal in the algorithm make up the gap between them to further enhance the transparency of the watermarking.

(2) OTP-W extraction

The process of the watermark extraction is shown in Algorithm 4.

Algorithm 4: OTP-W extraction
1. for each dimension \( i \)
2. if \( (d_i < 0) \)
3. \( w = (v_{\text{center}}, i^\alpha v_i) / a \)
4. else
5. \( w = (v_{\text{center}}, i^\alpha v_i) / a \)

The original 3D model is no need in the watermark extraction process. Therefore, our method is a blind watermarking.

(3) Analysis of capacity of OTP-W

Similar to the method in OTC-W Algorithm, the watermark is embedded to the bottom level space of Octree structure. However, we use a RGB image instead of a binary image and the color values are directly combined to the \( X, Y, Z \) axis coordinates of the vertices. Assuming the number of the vertices of the model is \( n_v \), the number of embedded pixels is \( n_w \) for RGB colorful images. For gray images, the capacity for watermark is \( 3 \times n_v \). For the bunny model with 35947 vertices, a color image with size about 180 x 180 can be embedded.
C. Zero-W Algorithm

OTC-W and OTP-W are robust against translation, rotation and uniform scaling attacks, due to the PCA and Octree coder and Octree partition. Furthermore, as the embedding process does not change the topology, the vertex reordering attacks can be resisted as well. However, some complicated attacks, as of vertex resampling, mesh simplification and noise addition change the number of total vertices and their relative positions, so the watermark embedded above will not be reserved after those attacks. In this paper, Zero-W based on some nodes of higher level of Octree, instead of M nodes of the bottom level, is used to make the watermarking system robust against aforementioned attacks. In fact, there is no watermark embedded into the 3D model in Zero-W, where some features are constructed according to the higher level nodes of Octree of 3D meshes without any changes. We register the Zero-watermark information in the Third Party Certificate Organization after Gzip [15] compression which reduces the size of Zero-watermark within 10 KB. Original 3D model is not needed in watermark detection since we merely compare the Zero-watermark reconstructed from the published model with the one obtained from the Third Party Certificate Organization, and the copyright can be determined according to the threshold.

(1) Zero-W embedding

The position of construction of Zero-W affects the watermark detection result. If the selected Octree level is too high, it might result in the possibility that different models match the same Zero-watermark. If the level used to construct 'Zero-watermark' is too low, then on one side, the size of 'Zero-watermark' will grow in a rapid speed; on the other side, the Zero-W method will not be robust against simplification. In this paper, all of the nodes in the 6th level of Octree are selected to construct 'Zero-watermark' after lots of experiments.

The number of positions in the 6th level of octree is 8^6. Position of each node will be figured out after a top-down traveling for Octree. The algorithm of constructing 'Zero-watermark' can be described as follows:
Algorithm 5: Zero-W construction
1. Initialize 'Zero-watermark' ZW[8^6], all set to 0;
2. for all node, ∈ the 6th level
3. get nodePosition, from node;
4. ZW[nodePosition[i]] = 1
5. end for

The size of 'Zero-watermark' will be several KBs after Gzip compression. Table III shows the compression results of some models.

From the results in the table, it can be seen that there is no real 'embedding' process in Zero-W. The constructed 'Zero-watermark' is only to be registered in the Third Party Certificate Organization after compression.

<table>
<thead>
<tr>
<th>Model</th>
<th>number of vertices</th>
<th>number of nodes of 6th level</th>
<th>Zero-W size after Gzip(KB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>bunny</td>
<td>35947</td>
<td>4787</td>
<td>2.25</td>
</tr>
<tr>
<td>horse</td>
<td>19851</td>
<td>3296</td>
<td>1.41</td>
</tr>
<tr>
<td>p47 10</td>
<td>49217</td>
<td>1855</td>
<td>1.16</td>
</tr>
<tr>
<td>blackface</td>
<td>18182</td>
<td>3876</td>
<td>1.99</td>
</tr>
</tbody>
</table>

(2) Zero-W extraction

The reconstruction of 'Zero-watermark' is the same as that in Section IV.C.(1). The reconstructed 'Zero-watermark' will be compared with the one of original 3D meshes obtained from the Third Party Certificate Organization, and then a relevant value λ will be calculated. The threshold λ is defined as 0.6 to determining whether ZW' and ZW belong to the same 3D meshes. The relativity can be calculated as follows:
Algorithm 6: Watermark relativity computation
1. int equal=0, sum=0;
2. for all element ZW[i] ∈ ZW'
3. if ZW'[i]=1 and ZW[i]=1
4. equal++;
5. sum++;
6. end if
7. if (ZW'[i]=1 and ZW[i]=0) or (ZW'[i]=0 and ZW[i]=1)
8. ZW[i]=1;
9. sum++;
10. end if
11. end for
12. float λ = equal/sum;

(3) Analysis of capacity of Zero-W

'Zero-watermark' describes the position distribution of 3D meshes' vertices in low resolution. Theoretically, the number of 'Zero-watermark' of 3D meshes can be calculated: 2^{n^6} =2^{262114}, a rather large scope in practice. The probability of registering a duplicate 'Zero-watermark' of different 3D meshes is very low. Even if it were possible, it could be solved by additional labeling.

V. Experiments

Our methods are implemented in Java 1.6.0 and Java3D 1.5.2. Jama external package [16] is used for the PCA algorithm. The experiments running environment is: Intel Pentium Dual CPU E2140 @ 1.6GHz, 1 GB RAM, NVIDIA GeForce 7300 GT. Fig. 3 and Fig. 4 are the results of OTC-W and OTP-W on bunny model.

<table>
<thead>
<tr>
<th>Model</th>
<th>number of vertices</th>
<th>number of nodes of 6th level</th>
<th>Zero-W size after Gzip(KB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>bunny</td>
<td>35947</td>
<td>4787</td>
<td>2.25</td>
</tr>
<tr>
<td>horse</td>
<td>19851</td>
<td>3296</td>
<td>1.41</td>
</tr>
<tr>
<td>p47 10</td>
<td>49217</td>
<td>1855</td>
<td>1.16</td>
</tr>
<tr>
<td>blackface</td>
<td>18182</td>
<td>3876</td>
<td>1.99</td>
</tr>
</tbody>
</table>

Figure 3. OTC-W on bunny. (a) is the original model. (b) is the model watermarked. (c) is the original watermark (size: 64*64). (d) is the extracted watermark.
Two images are obtained from original and embedded 3D meshes rendered on the same lighting and viewing point conditions. The Peak Signal to Noise Ratio (PSNR) is used for the similarity computation of the original and embedded 3D meshes. PSNR can be calculated as shown in (5).

$$PSNR = 10\log_{10} \frac{M \times N \times \max_{m,n} I_{m,n}^2}{\sum_{m,n} (I_{m,n} - I_{m,n}')^2}$$  \hspace{1cm} (5)$$

$I$ is the image of original meshes, and $I'$ is from embedded meshes.

Table IV shows the results. Lots of experiments indicate that the human eyes will not be able to distinguish image A and B if the PSNR of them is above 38dB [19]. The embedding algorithm meets the imperceptible request of watermarking from Table IV.

**TABLE IV. WATERMARK SIMILARITY EVALUATION**

<table>
<thead>
<tr>
<th>Model</th>
<th>Watermark size</th>
<th>PSNR$_{m,n}$ (dB)</th>
<th>Correlat $\text{Correlat}_{m,n}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>bunny, 35947 vertices, 69541 faces</td>
<td>64×64</td>
<td>46.7275</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>128×64</td>
<td>46.5610</td>
<td>0.9992</td>
</tr>
<tr>
<td></td>
<td>200×40</td>
<td>46.3370</td>
<td>0.9978</td>
</tr>
<tr>
<td></td>
<td>200×200</td>
<td>Out of capacity</td>
<td></td>
</tr>
<tr>
<td>horse, 19851 vertices, 39698 faces</td>
<td>64×64</td>
<td>48.0426</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>128×64</td>
<td>48.0426</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>200×40</td>
<td>47.9294</td>
<td>0.9645</td>
</tr>
<tr>
<td></td>
<td>200×200</td>
<td>Out of capacity</td>
<td></td>
</tr>
<tr>
<td>p47_10, 49217 vertices, 96720 faces</td>
<td>64×64</td>
<td>42.9435</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>128×64</td>
<td>42.9439</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>200×40</td>
<td>42.9387</td>
<td>0.9481</td>
</tr>
<tr>
<td></td>
<td>200×200</td>
<td>41.7826</td>
<td>0.9461</td>
</tr>
<tr>
<td>blackface, 18182 vertices, 36299 faces</td>
<td>64×64</td>
<td>43.8628</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>128×64</td>
<td>44.2379</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>200×40</td>
<td>43.8628</td>
<td>0.9955</td>
</tr>
<tr>
<td></td>
<td>200×200</td>
<td>Out of capacity</td>
<td></td>
</tr>
</tbody>
</table>

C. Attacks experiment

The bunny model is used in the attack experiment. Table V shows the results after a series of attacks, where the OTC-watermark size is 64×64, and the OTP-watermark size is 128×64.

VI. CONCLUSIONS

In this paper, three novel blind watermarking methods are proposed. After the preprocess of the PCA (Primary Component Analysis), Octree is applied to partition the 3D meshes. The watermarks, a binary image for OTC-W and a RGB image for OTP-W, are embedded into the bottom level space of Octree structure separately. Experiments have proved that those two methods can supply large spaces for watermark embedding. Meanwhile, they are robust against translation, rotation, uniform scaling and vertex random reordering. For Zero-W, higher nodes of Octree are used to construct ‘Zero-watermark’, which can resist simplification, noise and remeshing attacks. All those three methods are fit for arbitrary 3D meshes and 3D point sampled data.
ACKNOWLEDGMENT

Our work is supported by National Education Science Outline of the Eleventh Five-year Plan in China (Grant No: ACA07004), Ministry of Science and Technology in China (Grant No.2006BAK12B09) and Beijing Municipal Science & Technology Commission (Grant No. Z07000100560714).

REFERENCES

Su Cai was born in Hubei Province, China, in 1981. He received the BS and PhD degree in computer science from Beihang University, Beijing, China, in 2002 and 2008, respectively. He researched in 3D graphics compression and digital protection in the State Key Laboratory of Virtual Reality Technology, China from 2001 to 2008.

Dr Cai is currently a lecturer in the School of Educational Technology, Beijing Normal University. His research interests include computer graphics, virtual reality in education and advanced learning system.

Xukun Shen was born in Yunnan Province, China, in 1964. He received his BS, MS and PhD degrees in computer science from Beihang University (the former name is Beijing University of Aeronautics & Astronautics), China, in 1987, 1991 and 2007 respectively. He joined the Department of Computer Science & Engineering, Beihang University, in October 1991. From October 1991 to October 1997, he was a research assistant professor in natural language processing and analogical reasoning. From September 1997 to July 2002, he was an associate professor in virtual reality. He is now a professor of the School of Computer Science & Engineering and the duty director of the State Key Laboratory of Virtual Reality Technology, China. His current research interests are in fields of graphic modeling and rendering, digital asset management.

Prof. Shen is a director of the Committee of Federation of Chinese System Simulation.