Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

A blind watermarking algorithm for 3D mesh models based on vertex curvature*

Yong-zhao ZHAN†, Yan-ting LI, Xin-yu WANG, Yi QIAN
(*School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212013, China*)
†E-mail: yzzhan@ujs.edu.cn

Received Oct. 24, 2013; Revision accepted Feb. 14, 2014; Crosschecked Apr. 11, 2014

Abstract: We propose a robust blind watermarking algorithm for three-dimensional (3D) mesh models based on vertex curvature to maintain good robustness and improve visual masking in 3D mesh models. In the embedding process, by using the local window of vertex, the root mean square curvature is calculated for every vertex of the 3D mesh model and an ordered set of fluctuation values is obtained. According to the ordered fluctuation values, the vertices are separated into bins. In each bin the fluctuation values are normalized. Finally, the mean of the root mean square curvature fluctuation values of the vertices in each bin is modulated to embed watermark information. In watermark detection, the algorithm uses a blind watermark extraction technique to extract the watermark information. The experimental results show that the algorithm has a very good performance for visual masking of the embedded model and that it can resist a variety of common attacks such as vertex rearrangement, rotation, translating, uniform scaling, noise, smoothing, quantization, and simplification.

Key words: Three-dimensional mesh model, Visual masking, Root mean square curvature, Blind watermarking, Attack resisting
doi:10.1631/jzus.C1300306 **Document code:** A **CLC number:** TP391.41

1 Introduction

With the rapid development of ubiquitous intelligence in recent years, 3D models are being increasingly applied to various fields such as visualization in scientific computing, collaborative design, and digital entertainment. As an effective technique for copyright protection, digital watermarking techniques can play an important role in avoiding unauthorized copying, tampering, and illegal dissemination. The research for digital watermarking is focused mainly on image, text, video, and audio. However, it is also important to study the digital watermarking technique for three-dimensional mesh models. A very popular retrieval (Ohbuchi *et al.*, 1997) proposed several watermarking schemes for 3D models and afterwards the digital watermarking for 3D models became a popular research area, but the achievements obtained to date are still not satisfactory for practical applications.

Watermarking algorithms are categorized into zero watermarking and embedded watermarking based on whether it is necessary to modify the information obtained in the original 3D model. In recent years there have been considerable efforts devoted to improving zero watermarking algorithms (Zhang *et al.*, 2009; Wang and Zhan, 2011; Du *et al.*, 2013). Most of these efforts concentrate on model features. Depending on whether the information of the original model is required or not in watermarking detection, watermark algorithms can be categorized into non-blind watermarking algorithms and blind watermarking algorithms. Most previous research focused on non-blind watermarking algorithms.

The earliest non-blind watermarking algorithms (Benedens, 1999; Yu *et al.*, 2003) have low robustness because the watermark information is embedded by directly modifying the geometry and

* Project supported by the Specialized Research Fund for the Doctoral Program of Higher Education of China (No. 20113227110021)

topology information of the 3D model. The algorithm of Yin *et al.* (2001) embeds the watermark into a suitable coarser mesh produced using Guskov's multi-resolution signal processing and a 3D non-uniform relaxation operator. In this scheme of Ai *et al.* (2009) the existing 2D image algorithm is used to embed a watermark into a range image produced by the Voronoi patches of 3D models with feature points from the rapid changing regions. The algorithms proposed by Choi *et al.* (2010) use linear operators derived from scaling functions generated from Chebyshev polynomials to create vectors into which the watermark is inserted, thereby avoiding the cost of finding the eigenvalues and eigenvectors of a Laplacian matrix. By transforming 3D rectangular coordinates of the model into cylindrical coordinates with a constant interval in the Z axis and quantizing the radial angles with required angular change, the scheme of Zhang and Yao (2010) transforms the 3D model into a 2D matrix and embeds the watermark into the matrix. The algorithms of Hu *et al.* (2008) and Salman *et al.* (2008) pre-process the original model and formulate a more stable watermark embedding primitive to the embedded watermark. These approaches provide robustness against common attacks. Since it is hard and even nearly impossible to obtain reliable information about the original model in practice, blind watermarking algorithms provide a better theoretical value and also have good application prospects.

Research on blind watermarking algorithms is now a trend. Several blind watermarking algorithms for 3D mesh models have been developed. Embedding primitives have been determined by topology and then the watermark is embedded (Ohbuchi *et al.*, 1997; 2002). Sun and Pan (2005) created one local coordinate system for each embedding primitive and embedded watermark bit by bit by modifying the position of vertices' projections in the local coordinate system. Kim *et al.* (2005) modified the L_2-norm of the wavelet coefficients at various multi-resolution levels to embed the watermark. Yao *et al.* (2008) proposed a blind algorithm that embeds watermarks into vertex groups by applying a non-uniform discrete Fourier transform. However, their technique cannot resist simplification or cropping. Some algorithms (Hu *et al.*, 2009; Luo and Bors, 2009) based on a block and barycenter of the model have also been introduced, but they canot resist cropping because the barycenter of the model will be changed after a cropping attack. An extended version of the algorithm (Lee and Kwon, 2007) has also been proposed to alleviate the weaknesses of the blind watermarking scheme for simplification and cropping. The algorithm, however, is not a true blind watermarking scheme because it still requires some original data during the extraction process. Based on the idea of adding a watermark to the pixel values of the lower bits in image watermarking, Wagner (2000) constructed a set of parameter vectors that is robust to all kinds of transform operations and embeds watermarks by modulating the relative length of each vector. The algorithm can resist geometric transformations and affine transformations, but the robustness for mesh reconstruction is weak. Zagrouba and Jabra (2009) selected maximum stable partitions according to the vertex curvature and repeatedly embed watermarks into the partitions. The stability of the partitions and the repeated embedding guarantee some robustness to cropping and simplification attacks, but the robustness is still weak because the establishment of the partition is related to model topology. Wagner's scheme and Zagrouba's scheme focus mainly on the robustness performance in 3D mesh model watermarking. Li *et al.* (2004; 2006) proposed algorithms based on spherical parameterization and the addition property of a Fourier transform which are resistant to a variety of common attacks.

Cho *et al.* (2007) converted a 3D mesh model to spherical coordinates and embedded the watermark by modulating the histogram distribution of the vertices. The algorithm takes robustness into account, but does not select the visual masking regions or primitives to embed the watermark. So, the effect of visual masking is unsatisfactory. Wang *et al.* (2011) normalized the original model and established partitions, and then embedded a watermark bit repeatedly at the same part to resist simplification. The algorithm uses the barycenter of the model when the partitions are established and thus is not robust against cropping due to the shift of the barycenter caused by cropping. The algorithm analyzes the distortion distribution and intensity on the watermarked model, but how to enhance the visual masking of the watermark in the watermark embedding is still not considered. To conclude, the algorithms above have some robustness to various attacks, but the 3D model is sensitive to watermark embedding and is deformed easily. This will lead to obvious visual distortion and make the watermark easy to perceive, which is unfavorable for copyright protection of the 3D model. Ma *et al.* (2004) presented a direct approach for fitting a subdivision surface from an irregular and dense triangle mesh of an arbitrary topological type. They developed a topology- and feature-preserving mesh simplification algorithm, which, however, is not a watermarking algorithm. Therefore, a watermarking algorithm is needed that not only resists various attacks on the 3D model but also masks the watermark information embedded and preserves the model shape.

In this paper, a blind watermarking algorithm is proposed for 3D mesh models based on vertex curvature. The fluctuation value of the root mean square curvature of each vertex in the 3D mesh model is calculated based on the premise that all the vertices are divided into bins. After normalizing the fluctuation value of each bin, a watermark is embedded into the bins by modulating the mean fluctuation values of the bins.

2 Root mean square curvature fluctuation of a vertex

2.1 Root mean square curvature of a vertex

Curvature measures the uneven degree of the geometry, which is a local feature. Flatness has different meanings for different geometries. For a curve it is a straight line while for a surface it is a plane. A plane curve is uniquely determined by the curvature of the curve and a space curve is uniquely defined by the curvature (change rate of the unit tangent vector) and twist (change rate of the orthogonal frame). Conversely, a surface is more complicated. The accuracy of the curvature has a great influence on the performance of our algorithm. There exist three types of curvature for a vertex of a 3D model, i.e., Gaussian curvature, mean curvature, and root mean square curvature. In this paper, we choose root mean square curvature as the basis of our research.

The root mean square curvature is used to measure the deviation of a surface related to a plane. If the root mean square curvature of a surface is 0, the surface is a plane. The higher the root mean square curvature is, the more rapidly the deviation of the normal from the surface will change in a small area, and vice versa. For a geometric model, the Gaussian curvature and mean curvature are unable to correctly describe the characteristics of a 3D mesh model. Nevertheless, the root mean square curvature can do this. The human visual system is more sensitive to detail characteristics of a mesh model such as inflection points, creases, sharp edges, and triangular patch distributions. These characteristics can be better positioned with a root mean square curvature.

We calculate the Gaussian curvature and mean curvature using the formulas proposed by Kim *et al.* (1999). Suppose that K denotes the Gaussian curvature, and H denotes the mean curvature. Then, the Gaussian curvature and mean curvature of vertex $\boldsymbol{v}_i$ are proposed as follows:

$$K=\left(2\pi-\sum_{k=1}^{N_1}\theta_k\right)\Bigg/\left(\frac{1}{3}\sum_{k=1}^{N_3}A_k\right), \tag{1}$$

$$H=\sum_{j=1}^{N_2}\omega(e_{i,j})\Bigg/\left(\frac{1}{3}\sum_{k=1}^{N_3}A_k\right), \tag{2}$$

where θ_k (k=1, 2, …, N_1) denotes the interior angle adjacent to vertex $\boldsymbol{v}_i$, $\omega(e_{i,j})$ (j=1, 2, …, N_2) denotes the angles between the normals of the two adjacent triangles, and A_k (k=1, 2, …, N_3) denotes the triangle area corresponding to the 1-ring neighborhood of vertex $\boldsymbol{v}_i$.

The root mean square curvature of vertex $\boldsymbol{v}_i$ (Kim *et al.* 1999) is calculated as follows:

$$k_{\text{rms}}(\boldsymbol{v}_i)=\sqrt{2H^2-K}. \tag{3}$$

2.2 Fluctuation value of a vertex

Firstly, we define a local window of vertex $\boldsymbol{v}_i$ and select a threshold r as the radius of the window according to the watermark strength. Fig. 1 shows the local window $\text{Nv}_i(\boldsymbol{v}_i, r)$ of vertex $\boldsymbol{v}_i$ in a 3D mesh model. The local window is denoted by $\text{Nv}_i(\boldsymbol{v}_i, r)=\{\boldsymbol{v}_j \mid \|\boldsymbol{v}_j-\boldsymbol{v}_i\|\le r\}$, where $\|\boldsymbol{v}_j-\boldsymbol{v}_i\|$ denotes the Euclidean distance between $\boldsymbol{v}_j$ and $\boldsymbol{v}_i$. The points in the local window of vertex $\boldsymbol{v}_i$ are the hollow points in Fig. 1.

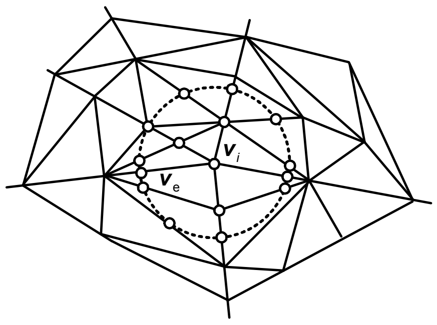

Fig. 1 Local window of vertex $\boldsymbol{v}_i$

Secondly, we calculate the root mean square curvature of the intersection point $\boldsymbol{v}_e$ of a local window $\mathrm{Nv}_i(\boldsymbol{v}_i, r)$ and 3D mesh model. As shown in Fig. 2, $\boldsymbol{v}_e$ denotes the intersection point, $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ denote the vertices of the line segment, and the root mean square curvature of the intersection point is defined as follows:

$$k_{\mathrm{rms}}(\boldsymbol{v}_e)=\sqrt{\left(\frac{d_2}{d_1+d_2}\cdot k_{\mathrm{rms}}(\boldsymbol{v}_1)\right)^2+\left(\frac{d_1}{d_1+d_2}\cdot k_{\mathrm{rms}}(\boldsymbol{v}_2)\right)^2}. \tag{4}$$

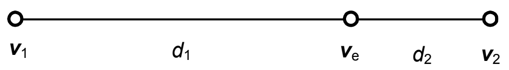

Fig. 2 Intersection point $\boldsymbol{v}_e$ of local window $\mathrm{Nv}_i(\boldsymbol{v}_i, r)$ and model

The root mean square curvature of vertex $\boldsymbol{v}_i$ in its local window $\mathrm{Nv}_i(\boldsymbol{v}_i, r)$ is defined as

$$k_{\mathrm{rms}}(\boldsymbol{v}_i,r)=\frac{1}{W_r(\boldsymbol{v}_i)}\sum_{\boldsymbol{v}_j\in \mathrm{Nv}_i(\boldsymbol{v}_i,r)} k_{\mathrm{rms}}(\boldsymbol{v}_j), \tag{5}$$

where $W_r(\boldsymbol{v}_i)$ denotes the number of vertices in $\mathrm{Nv}_i(\boldsymbol{v}_i, r)$, i.e., the number of the hollow points in Fig. 1.

Finally, we calculate the fluctuation value of vertex $\boldsymbol{v}_i$ as follows:

$$D_{v_i}=\sqrt{\frac{1}{W_r(\boldsymbol{v}_i)}\sum_{\boldsymbol{v}_j\in \mathrm{Nv}_i(\boldsymbol{v}_i,r)}(k_{\mathrm{rms}}(\boldsymbol{v}_i,r)-k_{\mathrm{rms}}(\boldsymbol{v}_j))^2}. \tag{6}$$

3 Blind watermarking algorithm based on vertex curvature

The information for a 3D mesh model includes vertex coordinates, faces, and attributes. Assuming that a 3D mesh model can be denoted as $\boldsymbol{M}=(\boldsymbol{V}, \boldsymbol{F})$, where $\boldsymbol{V}=\{v_i\in\mathbb{R}^3|1\le i\le N\}$ denotes the set of all vertices of the model, N denotes the number of vertices, $\boldsymbol{F}=\{f_i|1\le i\le N_\mathrm{f}\}$ denotes the set of all triangular faces representing mesh topology, and N_f denotes the number of faces.

3.1 Embedding process

Assuming that the watermark information sequence can be denoted by $\boldsymbol{W}=(w_0, w_1, \ldots, w_{L-1})$, $w_i\in\{0, 1\}$, $1\le i\le L-1$. To enhance the security of the watermark information, the watermark information in this paper is generated by logistic chaos mapping. The watermark information generating process can be described by a deterministic nonlinear differential equation. Although the watermark information generating process does not include any random factor, its trajectory may be completely random. In addition, logistic chaos mapping has advantages of privacy and periodicity in a state space. The logistic chaos mapping is defined as

$$\begin{cases} x_{k+1}=F(x_k)=\mu x_k(1-x_k), \\ x_{k+1}\in(0,\ 1),\quad k=0,\ 1,\ \cdots,\ L-1, \end{cases} \tag{7}$$

where $0\le\mu\le4$ is a bifurcation parameter. To make the watermark information into a chaotic state, μ should be in the range of [3.569945, 4] and the key (μ, x_0) is a cipher code where x_0 is an initial value of the sequence. The value of x_0 should be in the range of [0, 1]. According to the chaos sequence x_k (k=1, 2, …, L), the watermark information is defined as follows:

$$w_i=\begin{cases}1, & x_k\ge0.5, \\ 0, & x_k<0.5, \end{cases}\quad i=0,1,\cdots,L-1,\ \ k=1,2,\cdots,L. \tag{8}$$

Compared with the existing digital watermark signal, chaos mapping has three advantages. Firstly, based on the unique cipher code key (μ, x_0), we can construct different watermark sequences that are all unique chaos sequences which can be embedded into the models; moreover, we can almost reproduce the original sequence rapidly as long as we obtain the cipher code. Secondly, without knowing the cipher code or chaos mapping, the chaos sequence cannot be deciphered, which can enhance the security of copyright certification. Thirdly, by using logistic chaos mapping, the similarity between different watermarks can be effectively decreased, which guarantees the uniqueness of the watermark. So, the chaos sequence can effectively solve the problem of how to generate watermark in practical applications and can be treated as a digital watermarking standardization.

The root mean square curvature fluctuation value can be modulated based on the principle as follows: supposing that a continuous random variable X has a uniform distribution over the interval [0, 1], clearly, the expectation of X is given by $E[X]=\int_0^1 XP(X)\mathrm{d}X=0.5$, where $P(X)$ is the probability density function of X. $E[X]$ will be used as a reference value modified by shifting the mean fluctuation in each bin. It is very important to determine that the modified fluctuation value still exists within the range of the original bin because if the fluctuation value shifts from the original bin to another bin, it will have a serious impact on watermark extraction. For a continuous random variable X, the mapping function is defined as follows:

$$Y=X^k,\quad k>0, k\in\mathbb{R}. \tag{9}$$

The expectation of Y is represented as

$$E[Y]=E[X^k]=\int_0^1 X^k P(X)\mathrm{d}X=\frac{1}{k+1}. \tag{10}$$

The expectation is still limited in the range of [0, 1], and Y is convergent. Finally, as shown in Fig. 3, if the embedded bit is 1, set $Y>0.5+\alpha$; if the embedded bit is 0, set $Y<0.5-\alpha$, where $\alpha\in(0, 0.5)$ is modulation intensity. Then the information of the embedded watermark can be distinguished by selecting an appropriate value for parameter k. Because the 3D model is discrete, parameter k cannot be calculated directly. So, first we use an iterative algorithm to modify the valid vertices in a bin so that the root mean square curvature fluctuation values of all the vertices in the bin are changed. In this way, the value of k for the random variable about the mean of the normalized root mean square curvature fluctuation values of all vertices in this bin can be changed to a suitable status and the watermark information can be embedded. The valid vertices in the bin are modified iteratively to make the mean of the normalized root mean square curvature fluctuation values of all vertices in the bin greater than $0.5+\alpha$ to embed a watermark bit of 1 into a bin, and less than $0.5-\alpha$ to embed a watermark bit of 0 into a bin. To improve the robustness against various attacks, we repeatedly embed the watermark.

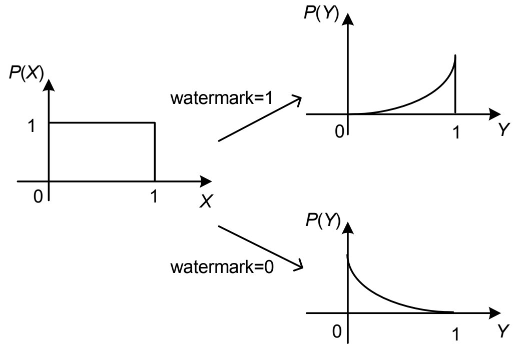

Fig. 3 The proposed watermarking method by modulating the variable of the distribution

The detailed steps of the embedding algorithm are as follows:

Step 1: Set the initial states of all vertices of the 3D mesh model valid, which indicates that all the vertices can be modified.

Step 2: For each vertex $\boldsymbol{v}_j$ (j=1, 2, …, N) of the 3D mesh model, calculate the root mean square curvature fluctuation value $D_{\boldsymbol{v}_j}$ according to Eqs. (1)–(6). To tolerate the vertex reordering more effectively, the fluctuation values of all vertices are sorted in ascending order. According to the length L of the watermark and the repeating times num of the embedding watermark, the sorted fluctuation values are divided into $L\cdot$num bins. The vertices in every bin can be expressed as

$$B_i=\left\{\boldsymbol{v}_j\,\middle|\,D_{\min}+\frac{D_{\max}-D_{\min}}{L\cdot\mathrm{num}}\cdot i\le D_{\boldsymbol{v}_j}\le D_{\min}+\frac{D_{\max}-D_{\min}}{L\cdot\mathrm{num}}\cdot(i+1)\right\}, \tag{11}$$

where i=0, 1, …, L·num−1, B_i is the subset of the vertices of the model, 0≤j≤N−1, N is the number of vertices of the model, and $D_{\max}$ and $D_{\min}$ are the maximum and minimum root mean square curvature fluctuation values in the model, respectively.

Step 3: The fluctuation value of vertex in bin B_i is normalized as

$$D'_{v_j}=\frac{D_{v_j}-D_{\min,i}}{D_{\max,i}-D_{\min,i}}, \tag{12}$$

where D_{v_j} is the root mean square fluctuation value of vertex $\boldsymbol{v}_j$ in the ith bin, and $D_{\max,i}$ and $D_{\min,i}$ are the maximum and minimum root mean square curvature fluctuation values of vertices in the ith bin, respectively. The value of D'_{v_j} in every bin is in the range of [0, 1].

Step 4: Embed the watermark into the bins by modulating the mean fluctuation values of the vertices in each bin. When we embed one watermark bit w_k into the vertices in bin B_i (i=0, 1, …, L·num−1), the mean fluctuation value of vertices in B_i should be greater than 0.5+α if w_k is 1, and less than 0.5−α if w_k is 0. However, we need not calculate the mean fluctuation value of the vertices in the bin because when the goal of modulating the mean fluctuation value is achieved, the sum of the new fluctuation values of vertices in B_i must satisfy

$$\text{sum}_i\in\begin{cases}\big([(0.5+\alpha)(D_{\max,i}-D_{\min,i})+D_{\min,i}]Q_i,\ D_{\max,i}Q_i\big),\\ \quad w_k=1,\\ \big(0,\ [(0.5-\alpha)(D_{\max,i}-D_{\min,i})+D_{\min,i}]Q_i\big),\\ \quad w_k=0,\end{cases} \tag{13}$$

where Q_i is the number of vertices in the ith bin. So, we need only to modulate the fluctuation values of vertices in the bin, which can be done by modifying the vertex coordinates in the bin. An iterative algorithm is used to modify the vertices in the bin in terms of the embedded watermark bit along the normal directions of the vertices until the sum of the fluctuation values of the vertices in the bin satisfies Eq. (13). For every vertex $\boldsymbol{v}_j$ (j=1, 2, …, Q_i) in each bin, check the state of vertex $\boldsymbol{v}_j$ and all the vertices in the local window of vertex $\boldsymbol{v}_j$; the vectors of the effective vertices, the states of which are valid, are modified as follows:

$$\boldsymbol{v}'_j=\boldsymbol{v}_j+(-1)^{w_k+1}\eta\frac{\boldsymbol{n}_j}{|\boldsymbol{n}_j|}, \tag{14}$$

where $\boldsymbol{n}_j$ is the normal vector of the jth vertex, w_k (k=0, 1, …, L−1) is a watermark bit with a value of 0 or 1, η is the strength of modulation determined by the iterative algorithm, $\boldsymbol{v}_j$ is the vector of the jth vertex denoted by the vertex coordinates and origin coordinates, and $\boldsymbol{v}'_j$ is the vector of the jth vertex after being modified, from which the new vertex coordinates can be achieved. Using Eq. (14), the new coordinates of every valid vertex in the bin can be calculated by a given value η. Subsequently, the new fluctuation value of every vertex in the bin is calculated again. If the sum of the new fluctuation values of vertices in the bin satisfies Eq. (13), the iterative algorithm is finished; otherwise, η is modified gradually and the above procedure is repeated. After modifying these vertices, the states of these vertices and the vertices in their local windows are set invalid, which means these vertices cannot be modified in the following calculation.

The iterative algorithm that embeds a watermark bit into a bin is described as follows:

```
(1) Set the initial $\eta$, $\Delta\eta$=0.001;
(2) Repeat {
      For valid vertices $v_j$ in the bin do
      {Calculate the new coordinate of $v_j$ using Eq. (14);
       Calculate the new fluctuation value of $v_j$ using Eq. (6); }
      Calculate the sum of these new fluctuation values in the bin and mark the sum as sum$_i$;
      If the sum$_i$ does not satisfy Eq. (13) then
      {Resume the original coordinates of all vertices in the bin;
       If the watermark bit needed to embed is 1 then
       {  If sum$_i$ is smaller than the lower limit of the range, then $\eta=\eta+\Delta\eta$;
          If sum$_i$ is greater than the upper limit of the range, then {$\Delta\eta=\Delta\eta/2$; $\eta=\eta-\Delta\eta$; }}
       If the watermark bit that needs to be embedded is 0 then
       {  If sum$_i$ is greater than the upper limit of the range, then $\eta=\eta+\Delta\eta$;
          If sum$_i$ is smaller than the lower limit of the range, then {$\Delta\eta=\Delta\eta/2$; $\eta=\eta-\Delta\eta$; }}}
   } until sum$_i$ satisfies Eq. (13) or there is no valid vertex in the bin;
```

(3) Set the states of these modified vertices and the vertices in their local window invalid.

3.2 Extraction process

The extraction process is similar to the embedding process. First, Eqs. (1)–(6) are used to calculate the fluctuation value of each vertex of the detected 3D mesh model. Then the fluctuation value sequence is sorted in ascending order and the $L\cdot$num bins are obtained by dividing the sorted fluctuation value sequence. Normalize the fluctuation values of all vertices in each bin and calculate the mean fluctuation value of all vertices in the bin. The detailed steps can refer to steps 2 and 3 of the embedding process.

By comparing the mean fluctuation value ${D'_{\mathrm{avg}(i)}}^{\mathrm{d}}$ of every bin, one watermark bit can be extracted from the corresponding bin. The formula of extracting one watermark bit in the ith bin is as follows:

$$w_i' = \begin{cases} 1, & {D'_{\mathrm{avg}(i)}}^{\mathrm{d}} \geq 0.5, \\ 0, & {D'_{\mathrm{avg}(i)}}^{\mathrm{d}} < 0.5, \end{cases} \quad 0 \leq i \leq L\cdot\mathrm{num}-1. \tag{15}$$

Finally, after the watermark sequence whose length is $L\cdot$num is extracted, we should count the corresponding bits of the watermark sequence to determine the final watermark whose length is L. For example, we count the watermark information from $w_i', w_{i+L}', \cdots, w_{i+(\mathrm{num}-1)\cdot L}'$ ($0\leq i\leq L-1$). If the number of 1's is greater than the number of 0's, the ith watermark information is 1; otherwise, the ith watermark information is 0. The extracted watermark is $\boldsymbol{W}^{\mathrm{d}}=\left(w_0^{\mathrm{d}}, w_1^{\mathrm{d}}, \ldots, w_{L-1}^{\mathrm{d}}\right)$, whose length is L.

To validate the robustness of our algorithm, the correlation coefficient is introduced to measure the similarity between the extracted watermark sequence and the original watermark sequence:

$$\mathrm{Corr}(\boldsymbol{W}^{\mathrm{d}},\boldsymbol{W})=\frac{\sum_{i=0}^{L-1}(w_i^{\mathrm{d}}-\overline{w^{\mathrm{d}}})(w_i-\overline{w})}{\sqrt{\sum_{i=0}^{L-1}(w_i^{\mathrm{d}}-\overline{w^{\mathrm{d}}})^2\sum_{i=0}^{L-1}(w_i-\overline{w})^2}}, \tag{16}$$

where $\overline{w^{\mathrm{d}}}$ is the mean of the extracted watermark bit, w_i^{d} is the ith watermark bit of the extracted watermark, $\overline{w}$ is the mean of the original watermark bit, and w_i is the ith watermark bit of the original watermark. If the correlation coefficient is negative, it is meaningless for copyright protection of the 3D model and we set it to zero. Usually, a threshold is given when verifying the copyright information. If the correlation coefficient is greater (resp., smaller) than the given threshold, it is considered that the registered original watermark is contained (resp., not contained) in the detected model.

4 Experiments and result analysis

The algorithm is implemented by using a CGAL Graphic Library and Visual Studio 2008. To validate the performance of the algorithm, we choose the Bunny, Horse, Venus, and Dragon models provided by Stanford University as test models. Bunny has 34 835 vertices and 69 666 faces. The length of the embedded watermark sequence is 256 and the watermark will be repeatedly embedded three times. Horse has 112 642 vertices and 225 280 faces. The length of the embedded watermark sequence is 300 and the watermark will be embedded three times. Venus has 100 759 vertices and 201 514 faces. The length of the embedded watermark sequence is 64 and the watermark will be embedded seven times. Dragon has 50 000 vertices and 100 000 faces. The length of the embedded watermark sequence is 128 and the watermark will be embedded three times. This algorithm is compared with the algorithms of Cho *et al.* (2007) and Wang *et al.* (2011) which are concerned with visual masking and robustness. The threshold of the correlation in this experiment is 0.4, α is 0.03, the initial η is 0.001, key (μ, x_0) is set as (3.7, 0.7), and r is 0.1. The results of Wang *et al.* (2011) are adopted from the original paper. We implement the algorithm of Cho *et al.* (2007) using the parameters in Table 1.

Table 1 Parameters for the algorithm proposed by Cho *et al.* (2007)

WM payload (bit)	Mesh model	Strength parameter
67	Bunny	0.03
49	Dragon	0.04
46	Horse	0.05
75	Venus	0.07

4.1 Visual masking

Since there are no recognized standards that can evaluate the visual masking effect, in this study we use the observation comparison method and the maximum root mean square error (MRMS) to evaluate the visual masking for subjectivity and impersonality. The observation comparison method is to determine the degree of deformation between the watermarked model and the original model by observing with the naked eye. The observation comparison results of the model are shown in Fig. 4.

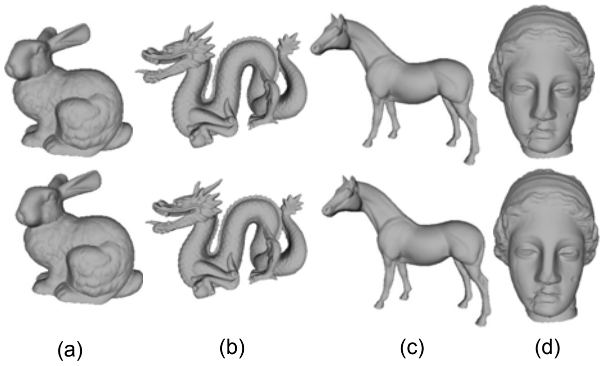

Fig. 4 Comparison of visual effect between the original models (up) and watermarked models (down) of Bunny (a), Dragon (b), Horse (c), and Venus (d)

MRMS is calculated by Metro (Cignoni *et al.*, 1998) to objectively measure the visual masking effect of the 3D model. The root mean square (RMS) is the root mean square of the vertex distance between the two models, the maximum of which is MRMS. MRMS is defined as follows:

$$E(V_1,V_2)=\max\{e_{\mathrm{f}}(V_1,V_2),e_{\mathrm{b}}(V_1,V_2)\}, \tag{17}$$

where

$$e_{\mathrm{f}}(V_1,V_2)=\frac{1}{|V_1|}\int_{V_1}\min_{\boldsymbol{v}_2\in V_2}\{\|\boldsymbol{v}_1-\boldsymbol{v}_2\|\}\mathrm{d}\boldsymbol{v}_1,$$

$$e_{\mathrm{b}}(V_2,V_1)=\frac{1}{|V_2|}\int_{V_2}\min_{\boldsymbol{v}_1\in V_1}\{\|\boldsymbol{v}_2-\boldsymbol{v}_1\|\}\mathrm{d}\boldsymbol{v}_2,$$

and V_1 and V_2 are the vertex sets of the original 3D model and the detected 3D model, respectively. The smaller the MRMS is, the better the visual masking effect of the algorithm will be. With the smaller distortion of the model, the watermark information embedded into the model is not easily perceived. The comparison results of the MRMS of this algorithm and the algorithms proposed by Cho *et al.* (2007) and Wang *et al.* (2011) are listed in Table 2. We can see that our algorithm has a better visual masking effect.

In this study, the fluctuation value of each vertex is calculated using a local window ensuring the masking scale of the 3D model to a certain extent (Lavoue, 2009). In the experiments, the differences between the models of before and after watermark embedding cannot be recognized with the naked eye in Fig. 4. It can also be seen from Table 2 that the MRMS value of our algorithm is lower than the values of Cho *et al.* (2007) and Wang *et al.* (2011). It shows that the visual masking effect of our algorithm is better than those of Cho *et al.* (2007) and Wang *et al.* (2011).

Table 2 Results of the MRMS distance

Model	MRMS ($\times10^{-3}$)			
	Bunny	Dragon	Horse	Venus
Our algorithm	0.42	0.45	0.71	0.36
Cho *et al.* (2007)	0.44	0.67	1.57	0.34
Wang *et al.* (2011)	1.75	1.76	1.04	2.34

4.2 Robustness

To verify the robustness of our algorithm against geometry attacks and connectivity attacks, vertex rearrangement, rotation, translating, uniform scaling, noise, smoothing, quantization, and simplification attacks are executed. Some display results of the attacked Dragon model and Venus model are listed in Figs. 5 and 6, respectively. Then we evaluate the differences between the original watermark sequence and the extracted watermark sequence by calculating the correlation coefficient. The visual distortion is also evaluated using MRMS between the attacked watermarked model and the original model.

Translation and rotation attacks have no effect on the relative position of the vertex and topology, and the uniform scaling does not change the root mean square curvature fluctuation value, which is a local feature of the model. Our algorithm has strong robustness against translation, rotation, and uniform scaling. After these attacks, the correlation coefficients between the extracted watermark and the original watermark are one. The sort operation on the root mean square curvature fluctuation value is adopted in our algorithm to resist the vertex rearrangement attack. So, after the vertex rearrangement attack, the correlation coefficient is also one.

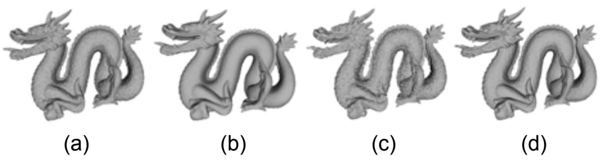

Fig. 5 The Dragon model watermarked by our method and attacked by 0.5% uniform noise (a), 50-iteration Laplacian smoothing (b), 7-bit coordinate quantization (c), and simplification with reducing 95% of vertices (d)

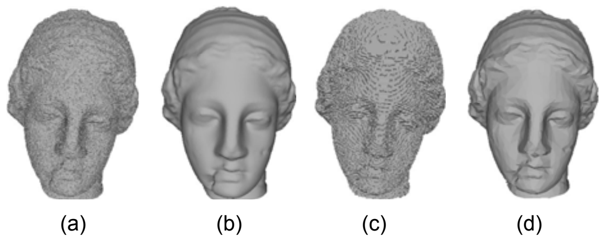

Fig. 6 The Venus model watermarked by our method and attacked by 0.5% uniform noise (a), 100-iteration Laplacian smoothing (b), 7-bit coordinate quantization (c), and simplification with reducing 97.5% of vertices (d)

For the noise attack, the uniform random noise is added to each vertex of the watermarked model. The amplitude of the noise is 0.1%, 0.3%, and 0.5%, respectively. The greater the amplitude is, the larger the distortion of the model is. The experimental results of random noise attacks are shown in Table 3. Our algorithm is better than those of Cho *et al.* (2007) and Wang *et al.* (2011) in terms of robustness and visual masking effect, except that visual masking effect of the Bunny model is slightly worse than that of Wang *et al.* (2011), while its correlation coefficient is better. The correlation coefficient of the Dragon model is worse than that of Wang *et al.* (2011) when the noise amplitude is 0.1%. The experimental results show that our algorithm has better robustness to uniform noise and the visual masking effect is good.

The Laplacian smoothing attack (λ=0.03) is performed 10, 30, and 50 times for Bunny and Dragon, and 10, 50, and 100 times for Horse and Venus. The experimental results are shown in Table 4. When the number of iterations is 30 for Bunny, 10 for Dragon, 50 for Horse, and 50 for Venus, the visual masking of our algorithm is not as good as that of Wang *et al.* (2011), but its robustness is better. Thus, it is noticed that the visual masking and robustness of our algorithm are globally well balanced.

Quantization is a lossy data compression method and it will make the model miss important information. Bits of quantization use binary to represent the coordinate of vertices. The fewer bits of quantization mean the model distortion is more serious. In this study, each coordinate of the vertices is represented with 9-, 8-, and 7-bit (Table 5). When the coordinates of the vertices are represented by 7-bit, the model is damaged too seriously to recover. This means it loses its commercial value and it does not need to be protected by watermark. Although the results of the correlation coefficient of the Bunny model are lower than those of Wang *et al.* (2011) and the correlation coefficient results of the Horse model are lower than those of Cho *et al.* (2007) when the coordinates of the vertices are represented by 7-bit, they are still greater than the threshold. This means there is a strong correlation between the extracted watermark and the original watermark, implying that our algorithm has good robustness against the quantization attack. The MRMS of our algorithm is better than those of Cho *et al.* (2007) and Wang *et al.* (2011); thus, we may conclude that our algorithm has better visual masking effect.

Since our algorithm embeds watermarks into the model repeatedly, the watermark can be extracted from the vertices that survive from simplification. Our algorithm is robust against simplification (Table 6). When the vertices of the Horse are reduced by 97.5%, the robustness of the algorithm is weaker than that of Wang *et al.* (2011), but the correlation coefficient is still higher than the threshold, so our algorithm has good robustness to simplification. The MRMS of our algorithm is better than the MRMS of Cho *et al.* (2007) and Wang *et al.* (2011), except that the visual masking effects of the Bunny model and Venus model are slightly worse than that of Wang *et al.* (2011). In general, our algorithm has good visual masking effect and robustness against the simplification attack.

From the experiment results, we can see that our algorithm generally has better robustness and better visual masking effect than Cho *et al.* (2007) and Wang *et al.* (2011). The main reason is that under common attacks including vertex rearrangement, rotation, translating, uniform scaling, noise, smoothing, quantization, and simplification, 3D mesh models generally can maintain their original shape

Table 3 Experimental results against the uniform noise attack

Model	Amplitude (%)	Corr			MRMS		
		Our algorithm	Cho *et al.*	Wang *et al.*	Our algorithm	Cho *et al.*	Wang *et al.*
Bunny	0.1	1.00	0.72	0.98	0.32	0.48	0.22
	0.3	0.91	0.72	0.85	0.40	0.79	0.66
	0.5	0.80	0.66	0.77	0.98	1.19	1.11
Dragon	0.1	0.94	0.83	0.98	0.33	0.71	0.24
	0.3	0.87	0.80	0.76	0.62	0.99	0.72
	0.5	0.77	0.65	0.61	1.01	1.38	1.20
Horse	0.1	1.00	0.95	0.98	0.19	1.48	0.21
	0.3	1.00	0.95	0.86	0.58	1.61	0.64
	0.5	0.86	0.91	0.77	0.97	1.81	1.07
Venus	0.1	0.95	0.94	0.94	0.24	0.47	0.33
	0.3	0.95	0.87	0.87	0.58	1.04	0.98
	0.5	0.79	0.27	0.78	0.99	1.67	1.63

Table 4 Experimental results against the smoothing attack (λ=0.03)

Model	Number of iterations	Corr			MRMS		
		Our algorithm	Cho *et al.*	Wang *et al.*	Our algorithm	Cho *et al.*	Wang *et al.*
Bunny	10	0.92	0.84	0.73	0.26	0.49	0.26
	30	0.85	0.60	0.62	0.70	0.78	0.69
	50	0.44	0.36	0.27	0.97	1.10	1.04
Dragon	10	0.91	0.80	0.84	0.33	0.67	0.31
	30	0.66	0.51	0.52	0.80	0.96	0.82
	50	0.44	0.26	0.19	1.12	1.34	1.28
Horse	10	1.00	0.95	1.00	0.06	1.46	0.07
	50	1.00	0.95	0.87	0.46	1.46	0.29
	100	0.77	0.72	0.74	0.46	1.49	0.52
Venus	10	1.00	0.94	0.92	0.12	0.34	0.12
	50	0.93	0.63	0.92	0.59	0.56	0.51
	100	0.78	0.45	0.84	0.88	0.89	0.88

Table 5 Experimental results against the quantization attack

Model	Intensity	Corr			MRMS		
		Our algorithm	Cho *et al.*	Wang *et al.*	Our algorithm	Cho *et al.*	Wang *et al.*
Bunny	9-bit	1.00	0.73	0.91	0.61	0.67	0.52
	8-bit	0.91	0.58	0.91	1.00	1.13	1.05
	7-bit	0.58	0.17	0.70	1.84	2.12	2.07
Dragon	9-bit	0.96	0.72	0.96	0.55	0.88	0.57
	8-bit	0.77	0.71	0.63	1.13	1.32	1.13
	7-bit	0.41	0.38	0.23	1.89	2.39	2.29
Horse	9-bit	1.00	0.95	1.00	0.32	1.54	0.49
	8-bit	0.86	0.95	0.70	0.78	1.76	0.97
	7-bit	0.57	0.68	0.49	1.97	2.52	2.05
Venus	9-bit	1.00	0.87	0.92	0.64	0.74	0.66
	8-bit	0.83	0.48	0.81	1.12	1.36	1.32
	7-bit	0.73	0.07	0.79	1.75	2.72	2.70

Table 6 Experimental results against the simplification attack

Model	Vertex reduction ratio (%)	Corr			MRMS		
		Our algorithm	Cho *et al.*	Wang *et al.*	Our algorithm	Cho *et al.*	Wang *et al.*
Bunny	70	1.00	0.11	1.00	0.40	0.46	0.21
	90	0.83	0.01	0.73	0.54	0.60	0.54
	95	0.74	0.00	0.74	0.76	0.83	0.95
Dragon	70	0.95	0.00	1.00	0.33	0.74	0.37
	90	0.66	0.00	0.56	0.97	1.05	1.00
	95	0.30	0.00	0.08	1.65	1.55	1.79
Horse	90	1.00	0.00	1.00	0.10	1.51	0.13
	95	1.00	0.00	0.96	0.08	1.52	0.24
	97.5	0.78	0.00	0.87	0.12	1.55	0.43
Venus	90	0.96	0.01	0.95	0.32	0.41	0.29
	95	0.77	0.00	0.89	0.46	0.53	0.51
	97.5	0.77	0.00	0.84	0.65	0.78	0.91

features, so the fluctuation values of the vertices can be preserved as a local feature and have a good robustness. From the above experimental results, we can see that our algorithm has better robustness than the algorithms of Cho *et al.* (2007) and Wang *et al.* (2011). The modulation is based on the fluctuation value and a local window, so there is little distortion and the algorithm has a good visual masking effect.

5 Conclusions

To maintain good robustness and improve the visual masking effect of watermark embedding, this paper proposes a visual masking algorithm of a blind watermarking technique for 3D mesh models based on the curvature of the vertex. The algorithm selects a curvature called root mean square curvature as the local characteristics and then the fluctuation values of the root mean square curvature are calculated. The fluctuation sequence is ordered based on which vertices are divided into bins. Finally, the watermark is embedded by modulating the mean normalized fluctuation values of the root mean square curvature. The watermark embedding process is based on the value of the root mean square curvature of the 3D model and it can effectively guarantee the visual masking effect of the 3D model watermarking. The experiments show that the algorithm has a very good visual masking effect and can better resist vertex rearrangement, rotation, translating, uniform scaling, noise, smoothing, quantization, and simplification.

References

Ai, Q.S., Liu, Q., Zhou, Z.D., *et al.*, 2009. A new digital watermarking scheme for 3D triangular mesh models. *Signal Process.*, **89**(11):2159-2170. [doi:10.1016/j.sigpro.2009.04.031]

Benedens, O., 1999. Watermarking of 3D polygon based models with robustness against mesh simplification. SPIE: Security and Watermarking of Multimedia Contents, p.329-340. [doi:10.1117/12.344683]

Cho, J.W., Prost, R., Jung, H.Y., 2007. An oblivious watermarking for 3D polygonal meshes using distribution of vertex norms. *IEEE Trans. Signal Process.*, **55**(1):142-155. [doi:10.1109/TSP.2006.882111]

Choi, H.I., Kim, T.W., Kwon, S.H., *et al.*, 2010. Digital watermarking of polygonal meshes with linear operators of scale functions. *Comput.-Aid. Des.*, **42**(3):163-172. [doi:10.1016/j.cad.2009.09.002]

Cignoni, P., Rocchini, C., Scopigno, R., 1998. Metro: measuring error on simplified surfaces. *Comput. Graph. For.*, **17**(2):167-174. [doi:10.1111/1467-8659.00236]

Du, S., Zhan, Y.Z., Wang, X.Y., 2013. A zero watermarking algorithm for 3D mesh models based on shape diameter function. *Comput.-Aid. Des. Comput. Graph.*, **25**(5):653-658 (in Chinese). [doi:10.3969/j.issn.1003-9775.2013.05.008]

Hu, M., Xie, Y., Xu, L.F., *et al.*, 2008. A geometry property based adaptive watermarking scheme for 3D models. *J. Comput.-Aid. Des. Comput. Graph.*, **20**(3):390-394 (in Chinese).

Hu, R., Rondao-Alface, P., Macq, B., 2009. Constrained optimisation of 3D polygonal mesh watermarking by quadratic programming. IEEE Int. Conf. on Acoustics, Speech, and Signal Process., p.1501-1504. [doi:10.1109/ICASSP.2009.4959880]

Kim, M.S., Valette, S., Jung, H.Y., *et al.*, 2005. Watermarking of 3D irregular meshes based on wavelet multiresolution analysis. 4th Int. Workshop on Digital Watermarking, p.313-324. [doi:10.1007/11551492_24]

Kim, S.J., Jeong, W.K., Kim, C.H., 1999. LOD generation with discrete curvature error metric. 2nd Korea Israel Bi-National Conf. on Geometrical Modeling and Computer Graphics in the WWW Era, p.97-104.

Lavoue, G., 2009. A local roughness measure for 3D meshes and its application to visual masking. *ACM Trans. Appl. Percept.* **5**(4):21. [doi:10.1145/1462048.1462052]

Lee, S.H., Kwon, K.R., 2007. A watermarking for 3D mesh using the patch CEGIs. *Dig. Signal Process.*, **17**(2):396-413. [doi:10.1016/j.dsp.2005.04.014]

Li, L., Zhang, D., Pan, Z., *et al.*, 2004. Watermarking 3D mesh by spherical parameterization. *Comput. & Graph.*, **28**(6): 981-989. [doi:10.1016/j.cag.2004.08.002]

Li, L., Pan, Z.G., Zhang, D., 2006. A public mesh watermarking algorithm based on addition property of Fourier transform. *Int. J. Image Graph.*, **6**(1):35-44. [doi:10.1142/S0219467806002070]

Luo, M., Bors, A.G., 2009. Shape watermarking based on minimizing the quadric error metric. IEEE Int. Conf. on Shape Modeling and Applications, p.103-110. [doi:10.1109/SMI.2009.5170170]

Ma, W., Ma, X., Tso, S.K., *et al.*, 2004. A direct approach for subdivision surface fitting from a dense triangle mesh. *Comput.-Aid. Des.*, **36**(6):525-536. [doi:10.1016/S0010-4485(03)00160-X]

Ohbuchi, R., Masuda, H., Aono, M., 1997. Watermarking three-dimensional polygonal models. 5th ACM Int. Conf. on Multimedia, p.261-272. [doi:10.1145/266180.266377]

Ohbuchi, R., Miyazawa, A., Takahasi, S., 2002. A frequency-domain approach to watermarking 3D shapes. *Comput. Graph. For.*, **21**(3):373-382. [doi:10.1111/1467-8659.t01-1-00597]

Salman, M., Ahmad, Z., Worrall, S., 2008. Robust watermarking of 3-D polygonal models. 3rd Int. Symp. on Communications, Control and Signal Processing, p.340-343. [doi:10.1109/ISCCSP.2008.4537246]

Sun, S.S., Pan, Z.G., 2005. A blind 3D mesh watermarking scheme based on local coordinate system. IEEE 7th Workshop on Multimedia Signal Processing, p.1-4. [doi:10.1109/MMSP.2005.248691]

Wagner, M., 2000. Robust watermarking of polygonal meshes. Proc. Geometric Modeling and Processing, p.201-208. [doi:10.1109/GMAP.2000.838252]

Wang, K., Lavoue, G., Denis, F., *et al.*, 2011. Robust and blind mesh watermarking based on volume moment. *Comput. & Graph.*, **35**(1):1-19. [doi:10.1016/j.cag.2010.09.010]

Wang, X.Y., Zhan, Y.Z., 2011. Robust zero watermarking scheme for 3D point model. *Comput. Eng. Appl.*, **47**(28): 7-11 (in Chinese). [doi:10.3778/j.issn.1002-8331.2011.28.002]

Yao, Z., Yang, S., Chen, L., *et al.*, 2008. A non-uniform scale, rotation and translation resilient public watermarking for 3D models. Int. Conf. on Cyber Worlds, p.531-536. [doi:10.1109/CW.2008.97]

Yin, K.K., Pan, Z.G., Shi, J.Y., *et al.*, 2001. Robust mesh watermarking based on multiresolution processing. *Comput. & Graph.*, **25**(3):409-420. [doi:10.1016/S0097-8493(01)00065-6]

Yu, Z.Q., Sip, H.H., Kwok, L.F., 2003. A robust watermarking scheme for 3D triangular mesh models. *Patt. Recogn.*, **36**(11):2603-2614. [doi:10.1016/S0031-3203(03)00086-4]

Zagrouba, E., Jabra, S.B., 2009. A new approach of mesh watermarking based on maximally stable meshes detection. 3rd Int. Conf. on New Technologies, Mobility and Security, p.1-5. [doi:10.1109/NTMS.2009.5384669]

Zhang, D.M., Yao, L., 2010. A non-blind watermarking on 3D model in spatial domain. Int. Conf. on Computer Application and System Modeling, p.V10-267-V10-269. [doi:10.1109/ICCASM.2010.5622796]

Zhang, J.W., Pan, G., Jiang, C., 2009. A locatable zero watermarking scheme and visualization for 3D mesh models. 6th Int. Conf. on Computer Graphics, Imaging and Visualization, p.510-515. [doi:10.1109/CGIV.2009.49]