Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105 (Online)
Volume **5**, Number **3**, August **2020**

Reversible Visible Watermarking Algorithm for 3D Models

Jinliang Cao
Institute of Digital Media and Communications
Taiyuan University of Science and Technology,Taiyuan,china,030024
504758009@qq.com

Zhiwei Niu
Institute of Digital Media and Communications
Taiyuan University of Science and Technology,Taiyuan,china,030024
2444299774@qq.com

Anhong Wang
Institute of Digital Media and Communications
Taiyuan University of Science and Technology,Taiyuan,china,030024
wah_ty@163.com

Li Liu
Institute of Digital Media and Communications
Taiyuan University of Science and Technology,Taiyuan,china,030024
skies5315@sina.com

Received April 2020; revised June 2020

Abstract. *This paper presents an algorithm of reversible visible watermarking for 3D models. The algorithm is used for identifying the copyright and protecting the data of 3D model files. The algorithm can embed the watermark information into the smoothest region of the model. After embedding into the watermark, a few parts of the original model are modified. Also, the original model can be restored without damage. The particularity of the algorithm is that the watermark information will not be visible after the watermark model is printed, and the embedded watermark information can only be seen in mesh and point cloud format. This algorithm can embed Chinese characters, English characters, numbers, graphics marks and other watermark information in 3D model files, which has certain robustness.*
Keywords: 3D model, Reversible watermark, Mesh subdivision;Visible watermarking

1. **Introduction.** With the increasing use of 3D models in many applications, such as medical imaging, computer-aided design, mechanical design, the manufacturing industry, virtual reality, and 3D movies, the need to protect the copyright and authentication of 3D content has become increasingly urgent and critical [1]. Watermark technology is considered as an effective solution to protect the copyright and authentication of 3D models [2].

According to the transparency, a digital watermark can be divided into a visible watermark and an invisible watermark. The invisible watermark is to embed the watermark information into the original model under the condition that changes cannot be observed, and the watermark content in the model cannot be seen by the naked eye after embedding. Visible watermark means that, after embedding the watermark information, we intuitively can see the embedded watermark information on the three-dimensional model.

3D point clouds and meshes are the basic representations of 3D models. In recent years, many watermarking algorithms for 3D mesh models have been proposed. As an important digital product, copyright protection also has attracted people's attention. Ohuchi et al. [3] first proposed the concept of the digital watermarking of 3D model, and they proposed several different visible and invisible algorithms. According to the different embedding domain of a watermark, it can be divided into a spatial-domain watermark and a frequency-domain watermark. The spatial 3D model watermarking algorithm [4-10] embeds the watermark by changing the geometric characteristics or structural connectivity of the model. The frequency-domain 3D model watermarking algorithm [11-13] transforms the geometric information of the 3D model into the frequency domain and then embeds the watermark by changing the coefficients. Zhan et al. [9] calculated the root mean square curvature of each vertex in the local window, divided the vertex into binary bits, and modulated the curvature fluctuation of the vertex of each bin to insert the watermark information. Jiang et al. [10] first mapped the decimal of the vertex coordinates to an integer and then embedded the data into the selected vertex by manipulating its least significant bits. Wang et al. [11] proposed a three-dimensional, semi-irregular mesh watermarking framework and embedded it in wavelet coefficients of different resolution levels. Hamidi et al.[13] proposed a method of inserting watermarks by changing the normal vector of the wavelet coefficients. Liu et al. [14] embedded watermarks by using the location of the feature vertex and built a new coordinate system by using a non-feature vertex, which enabled the coordinate system that was built to avoid the impacts of the embedding of the watermark. Ref. [15] proposed the minimisation of distortion to reduce the watermarking distortion and established the relationship between the error distortion using the mean square error and the selected Graph Fourier coefficients to embed the watermark.

[16-19] are image watermarking algorithms. Among them, [16,17] are reversible watermarking algorithms, and [18,19] are watermarking algorithm based on neural networks.[16] proposed a novel RDH algorithm. In this algorithm, data can be adaptively embedded into the block according to the local complexity, and the smoother the block, the more data it can carry.[17] proposed a reversible information hiding algorithm based on integer transformation. The embedding of information is realized by expanding the difference between a pixel and each of its three adjacent pixels.[18] proposed an adaptive fully dense(AFD) neural network for CT image segmentation. By adding the horizontal connections in UNet structure, it can extract various features from all layers adaptively. And it use ensemble training for the output to extract more edge information in the multiple rounds training.[19] proposed a multilayer dense attention model for image caption. A faster recurrent convolutional neural networks (Faster R-CNN) is employed to extract image features as the coding layer, the long short-term memory (LSTM)-attend is used to decode the multilayer dense attention model, and the description text is generated.

At present, the invisible watermarking algorithm of the 3D model have been studied extensively, and a few visible watermarking algorithms have been developed. But, to date, a reversible, 3D-model, visible watermarking algorithm has not been developed. The term ?reversible? means that the watermark model can be restored to the original model. Compared with an invisible watermark, a visible watermark can allow people to see the embedded watermark information. Lu Chen et al. [20] proposed a visible watermark algorithm that gave the visible watermark effect of embedding a few simple English characters, such as "WM". However, if there is a need to embed complex Chinese characters, the algorithm cannot provide a way to draw the watermark information. The visible watermarking algorithm proposed by An Xinchen et al. [21] can embed both Chinese and English characters, as well as numbers, but the embedding process is relatively

complex in that it requires both the subdivision of the mesh and cutting the subdivision of the mesh. The watermark contains vector information, which requires complex additional processing. The paper has the advantage that it can embed Chinese characters, English characters, numbers, patterns, and symbols. After embedding watermarks, there are slight changes to the model, and it has certain robustness. Also, it can restore the original model losslessly.

2. **Visible watermarking algorithm for 3D model.** The purpose of this paper is to provide an algorithm for embedding a visible watermark in a 3D mesh and a point cloud model, and the algorithm can realize the lossless recovery of the mesh model after extracting the watermark. There are two parts in our algorithm, i.e., (1) the watermark embedding process and (2) the watermark extracting and original model recovery process.

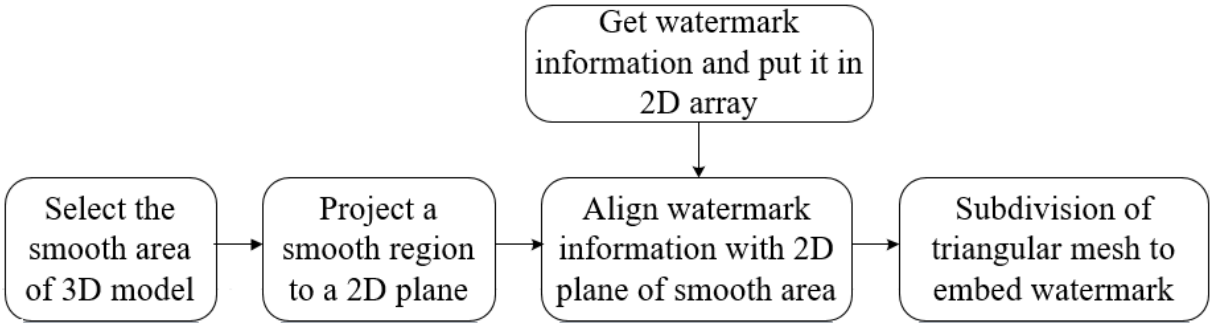


FIGURE 1. Watermark embedding flow chart

2.1. **Watermark embedding process.** The specific operation steps are as follows (as shown in Figure 1): (1) Obtain the two-dimensional array of watermark information; (2) Select the smooth region; (3) Project the smooth region to the two-dimensional plane; (4) Align the watermark array with the two-dimensional smooth region; (5) Embed the watermark in the subdivision mesh.

2.1.1. *Two-dimensional array for obtaining watermark information.* The watermark information here refers to a two-dimensional array with the contents of "0" and "1". Thus, there are two ways to get the two-dimensional array of watermark information.The first way is to use the Windows API function, GetGlyphOutline(), to generate an array of outlines of characters (including Chinese characters, English letters, and Arabic numerals) in a TrueType font. The contents of the array are "0" and "1". The lattice array of different characters can be spliced structurally to express more abundant watermark content, as shown in Figure 2, and the binary image can be converted from the lattice array of characters"科大3D".

The second method to obtain the watermark information is as follows.

Step 1: Read a drawn binary image with a key pattern or logo, and save the gray value of the image in a two-dimensional array;

Step 2. Change the gray value of the key position to "0", and the gray value of the other positions to "1";

Step 3. The binary image is converted from the dot matrix array, as shown in Figure 3.

The watermark information that is obtained, which is recorded as $W = [(a, b), c]$, is stored in the two-dimensional lattice array. Here,(a, b) represents the location of each element in the two-dimensional array, and c represents the corresponding location element, the value of which is either 0 or 1.

FIGURE 2. Binary image

FIGURE 3. Binary image of school badge

2.1.2. *Select smooth area.* In order to ensure the visual effect of the embedded watermark, the best place to embed the watermark is in a smooth area of the 3D model.

When selecting a smooth area, the center point of the smooth area is selected first. (Note that, when the original 3D model is in a point cloud format, a triangle patch can be constructed for the point cloud model using the Delaunay triangulation method, thereby turning the point cloud model into a mesh model.) Assume that the imported original 3D mesh model is $Mod = (V, F)$, where V is the vertex set of the 3D mesh model, and $V = \{V_i | V_i = (x_i, y_i, z_i) | x_i, y_i, z_i \in R, 1 \leq i \leq N\}$, where V_i is the i^{th} vertex; x_i , y_i and z_i are the three-dimensional coordinate values of vertex V_i ; N is the total number of vertices of the three-dimensional mesh model; F is the set of triangular patches of the three-dimensional mesh model, $F = \{f_j | f_j = (v_o, v_p, v_q) | v_o, v_p, v_q \in R, 1 \leq j \leq N_f\}$, where v_o, v_p, v_q are the three vertices of the j^{th} patch; and N_f is the total number of triangular patches of the three-dimensional mesh model. And the value calculated by Eq. (1) is used as a reference value for smoothness.

$$S(V_i) = \sum_{k=1}^{N} (\overrightarrow{p_i} \cdot \overrightarrow{p_k}) = \sum_{k=0}^{N} (x_i \cdot x_k + y_i \cdot y_k + z_i \cdot z_k) \tag{1}$$

$$\overrightarrow{p_i} = \frac{\sum_{k=1}^{N_i} (\overrightarrow{V_i} \cdot \overrightarrow{V_k})}{N_i} \tag{2}$$

$\overrightarrow{p_i}$ in Eq.(1) represents the normal vector of the center point of the smooth area, and $\overrightarrow{p_k}$ represents the normal vector of the vertex V_i closest to the center vertex found by the priority traversal method.

In Eq.(2), the normal vector $\overrightarrow{V_i}$ of the vertex V_i is from the centroid of the model to the vertex V_i. In order to better calculate the smoothness value, use $\overrightarrow{p_i}$ to represent the normal vector of the vertex V_i, and vertex $V_k(k = 1, 2, ..., N_i)$ represents the vertices of a 1-ring neighborhood of vertex V_i, and N_i represents the number of vertices in the 1-ring neighborhood of vertex V_i.

By traversing all of the vertices of the model, we calculate the S value of all vertices and find the vertex with the largest S value as the center point of the smooth area. After the center point is found, the k-order neighborhood triangles of the center point are used as the smooth area. The value of k depends on the amount of watermark information that is embedded.

2.1.3. *Project a smooth area to a 2D plane.* The triangle patch of the selected smooth region is composed of vertices; in fact, it is a collection of some vertices. The projection of the smooth region to the two-dimensional plane is conducted in the direction of the normal vector $\overrightarrow{P_i}$ of the center point of the smooth region. The projection process actually is the transformation of the three-dimensional coordinate system. Taking the coordinate origin as the rotation center point, the z-circle direction of the original coordinate axis is transformed into the direction of the normal vector $\overrightarrow{P_i}$ of the center point of the smooth area. The specific formula is as follows:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = R_1(\alpha)R_2(\beta) \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3}$$

In Eq.(3), $R(\cdot)$ is the rotation matrix, (x, y, z) is the vertex coordinate value of the original smooth area, (X, Y, Z) is the vertex coordinate value of the projected smooth area, and (X, Y) is the coordinate value of the 2D plane.

It can be understood specifically that the vertices in the smooth region first rotate α angle around x-circle, then rotate β angle around y-circle, and, finally, z-circle reaches the normal vector $\overrightarrow{P_i}$ direction, where α is the angle between the normal vector $\overrightarrow{P_i}$ and the $x \cdot z$ plane of the three-dimensional coordinate system, and β is the angle between the normal vector $\overrightarrow{P_i}$ and the $y \cdot z$ plane of the three-dimensional coordinate system.

The two-dimensional smooth area after the projection is recorded as s, where $s = (V', C)$, $V' = \{v'_i | v'_i = (x, y), 1 \leq i \leq N'\}$, where N' is the total number of vertices in the smooth region, V' is the vertices in the two-dimensional smooth region, and $C = \{c_j, 1 \leq i \leq N'\}$, where c is the sequence number of the vertex V' in the smooth region in the original model.

2.1.4. *Align the Watermark Array with the 2D Smooth Region.* Before alignment, the minimum actual distance d between the array elements must be specified, because there is an actual distance between the vertices of the three-dimensional model. We need to calculate the average value L of the vertex distance from each vertex in the smooth area to the vertex of the 1-ring domain, and, find the mean value $L' = L/N'$ of values L of all the vertex in the smooth area, where N' is the total number of vertices in the smooth area. Watermark embedding works is best when $d = 1.4L'$.

We traverse the two-dimensional smooth region s, and find the maximum and minimum values of the abscissa and ordinate of each vertex in the region, which are recorded as $X_{max}oX_{min}oY_{max}oY_{min}$. Similarly, we traverse the watermark information W, and find the

maximum and minimum values of the abscissa and ordinate of each vertex in the region, which are recorded as $X^W_{max}oX^W_{min}oY^W_{max}oY^W_{min}$.Then we multiply each vertex in the two-dimensional smooth region by d to meet the conditions $(X_{max} - X_{min}) > d(X^W_{max} - X^W_{min})$ and $(Y_{max} - Y_{min}) > d(Y^W_{max} - Y^W_{min})$, i.e., each vertex of the watermark information must be within the range of the vertices of the smooth region.

After that, the center point $V_0 = (X_0, Y_0)$ of the smooth region is aligned with the vertex $v_0 = (x_0, y_0)$ of the watermark information center. If the numbers of rows and columns of the array are odd, the coordinates of the watermark center point are at the midpoint of the array. If the numbers of the rows and columns of the array are even, the center point of the watermark is the number of rows and columns of the array divided by 2. All vertices of the watermark information must be at the same vector distance as the vertex translation of the watermark information center.

After alignment, we traverse the triangular patches in the smooth area to determine whether there are any points in each triangular patch where the element in the watermark array is "1". If so, we record these triangular patches $F' = \{f_{t1}, f_{t2}, ..., f_{ti}, ...\}$, and ti is the ordinal number of triangular patches in the original model.

According to the coordinates of the watermark vertex and the three vertices of the triangle patch, we can judge whether the vertex is in the triangle. Triangle ABC and point P are known. If the sum of the areas of the triangles PAB, PAC, and PBC is equal to the area of triangle ABC, then it can be determined that point P is in triangle ABC (including on three sides). If they are not equal, the vertices are outside the triangle.

2.1.5. *Subdivision of the mesh to embed the watermark.* The triangle face F' recorded in chapter 2.1.4 was subdivided in the original three-dimensional model to achieve the embedding of the watermark. In order to find the subdivided triangular patches accurately when the watermark model is restored to the original model, it is necessary to add the vertices inside and on the edges of the subdivided patches. The patch cannot be divided into six, because there are six neighboring patches and vertices in the 1-ring of a vertex on the model. If it is divided into six, it is not easy to find the newly-added vertices and subdivided patches. The subdivision methods are shown in Figure 4:

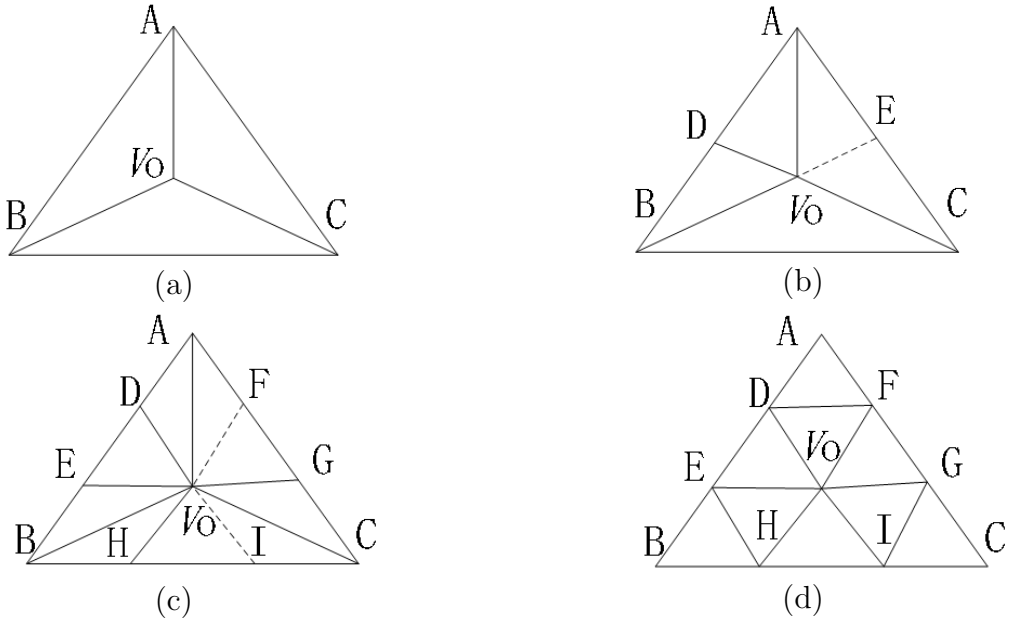


FIGURE 4. Schematic diagram of four mesh subdivision methods

In Figure 4(a), the triangular patchs is divided in the way of one is divided into three. The center of gravity of the triangle must be inside the triangle, and it is easy to calculate. It is assumed that the center of gravity of the triangular face ABC is $V_0 = (x_0, y_0, z_0)$, and the coordinates of the three vertices of the triangular face ABC are (x_1, y_1, z_1)o(x_2, y_2, z_2) and (x_3, y_3, z_3) respectively. Then, $x_0 = (x_1 + x_2 + x_3)/3$, $y_0 = (y_1 + y_2 + y_3)/3$, and $z_0 = (z_1 + z_2 + z_3)/3$. After the coordinates of the center of gravity of the triangular patch have been obtained, the centers of gravity V_0 are connected respectively to the three vertices of the triangular patch ABC. In this way, the triangular patch ABC is divided into three. The vector of the new vertex V_0 can be calculated by the vector sum of the three vertex vectors of ABC. The new triangular patch vector can be replaced by the old triangular patch ABC vector.

In Figure 4(b), the triangular patchs is divided in the way of one is divided into four or five. In the previous method, the center of gravity V_0 of the triangular patch ABC was found. We note that the midpoint of the edge AB of the triangular patch ABC is D, and the midpoint of the edge AC is E. Their coordinates are (x_4, y_4, z_4) and (x_5, y_5, z_5) respectively. And $x_4 = (x_1 + x_2)/2$, $y_4 = (y_1 + y_2)/2$, $z_4 = (z_1 + z_2)/2$, $x_5 = (x_1 + x_3)/2$, $y_5 = (y_1 + y_3)/2$, $z_5 = (z_1 + z_3)/2$. If the triangular patch ABC is to be divided into four parts, we just divide the patch into three parts and connect the vertex D and the center of gravity V_0. If want to divide a triangular patch into five parts, we also need to connect the vertex E with center of gravity V_0. The vector of vertex D is the sum of the vectors of vertices A and B. The vector of vertex E is the sum of the vectors of vertices A and C.

In Figure 4(c), the triangular patchs are divided in the way of one is divided into seven, eight, or nine. We find the three bisection points on the three sides of the triangular patch ABC i.e., AD=DE=EB=1/3AB; AF=FG=GC=1/3AC; BH=HI=IC=1/3BC. In the previous methods, the center of gravity V_0 has been connected to vertices A, B and C. If we want to divide the triangle into seven parts, connect the center of gravity V_0 to four of the six vertices(the vertices is D, E, F, G, H and I). Similarly, to divide a patch into eight or nine, we shoud connect the center of gravity V_0 to five or six of the vertices DoEoFoGoHoI. It also can be divided into nine according to way of figure 4(d), but it is difficult to find the subdivided mesh in the inverse process.

If the original model data are in a point-cloud format, and, if you want to get a watermark model in a point-cloud format, you must delete the data of the triangular patch and the normal vector of the triangular patch.

2.2. **Reverse process-restore to original model.** Figure 5 shows that the reverse process requires three steps, i.e., 1. Find the smooth area of the embedded watermark; 2. Find the newly-added vertices and divided triangles when embedding the watermark; 3. Delete the newly-added vertices and combine the divided triangular patch into one. The details are as follows:

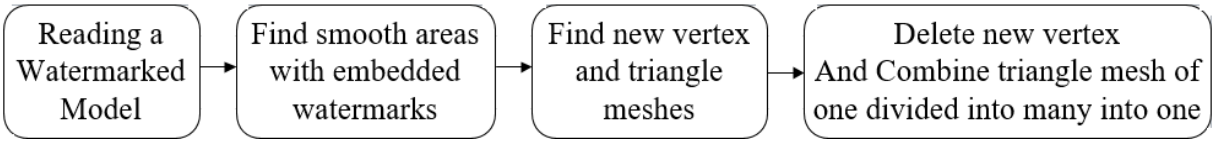


FIGURE 5. Reverse process - recovery of the original model flow chart

2.2.1. *Find smooth areas with embedded watermarks.* The smooth area found when embedding the watermark already is the smoothest area of the 3D model. After subdividing a part of the smooth area, the area is still the smoothest area of the model. We just

need to widen the area of the smooth area to ensure that all subdivided triangles are included. The method of finding smooth areas is still the method of selecting smooth areas according to the watermark embedding process.

First, we find the smooth center point according to Eqs. (1) and (2), and, then, we use the $(K+d\times L/2)$-order neighborhood triangular patches of the center point as the smooth area. K is the K-order neighborhood of the embedding process, d is the minimum actual distance between the elements of the watermark array, and L is the maximum value in the rows and columns of the watermark array. The addition of $d \times L/2$ is to ensure that the selected smooth area contains all of the subdivided triangular patches.

2.2.2. *Find the new vertices and the triangular patch that were divided into multiples when embedding the watermark.* We need to iterate through all of the vertices in the smooth region and find the vertices V_0' with three vertices in the 1-ring neighborhood. If the triangular patch was divided into three in the watermark embedding process, it will be three, and if it is to be divided into several, choose the desired number. And we record the subscripts of the vertices, and then we find the vertices and patches of a circle of the vertex.

2.2.3. *Delete the newly-added vertices and combine the divided triangular patch into one.* There are three cases here, i.e., the triangular patch is divided into three, divided into four or five, or divided into seven, eight, or nine. These three cases can be dealt with one by one.

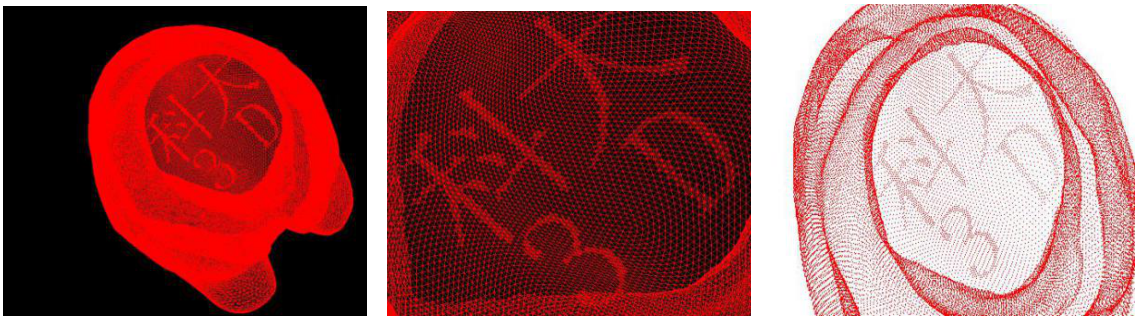
When the patch is divided into three, we delete the vertices V_0' and the 1-ring neighboring patches of this vertices. And then, we use the three vertices of 1-ring neighboring of the vertices V_0' to form a new triangular patch. The normal vector of the new triangular patch is the normal vector of the deleted patch. (The normal vector of the new triangular patch when subdividing the triangular patch is the normal vector of the subdivided triangular patch.)

When it is divided into four or fiveothe vertices of 1-ring neighboring of vertices V_0' must have three vertices on a straight line. We need to delete the middle vertices of the three vertices on a straight line. There is an intermediate vertex when it is divided into four and two intermediate vertices when it is divided into five. We need to delete the intermediate points and vertices V_0', delete patches of the 1-ring neighborhood of vertices V_0', and use the three vertices of 1-ring neighboring of the vertices V_0' to form a new patch. Similarly, the normal vector of the new triangular patch is the normal vector of the deleted patch.

When it is divided into seven, eight, or nine, the vertices of 1-ring neighboring of vertices V_0' must have three or four vertices on a straight line, so we need to delete the middle vertices on a straight line and keep the two vertices at both ends. Finally, there are three vertices in the 1-ring neighborhood of vertices V_0'. We also need to delete all the patches of the 1-ring neighborhood of vertices V_0' and the vertices V_0', and use the three vertices in the 1-ring neighborhood to form a new patch. The normal vector of the new triangular patch is the normal vector of the deleted patch.

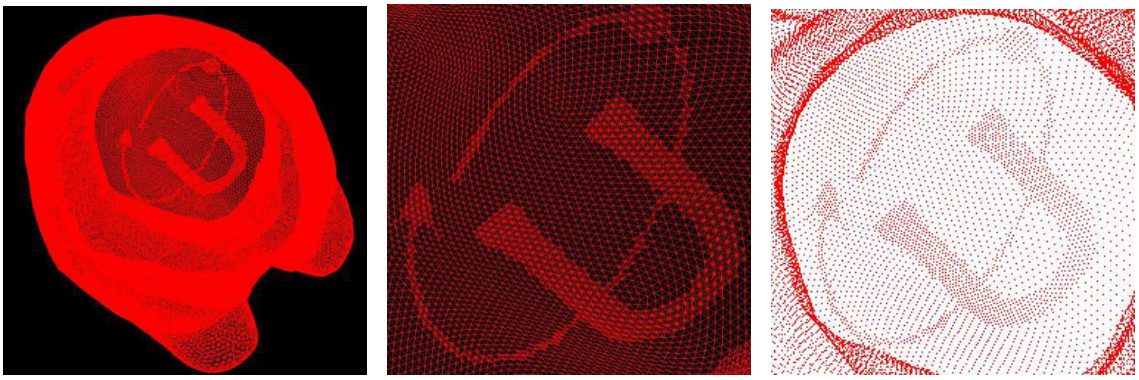
3. **Experimental results and analysis.** In order to show the actual effect of this algorithm, different watermark information is embedded in the 3D models "rabbit" and "horse". The watermark information has the characters "科大3D" and the school emblem pattern. The actual effects are shown in Figures 6, 7, 8, and 9.

The algorithm has certain robustness. As shown in Figure 10, when Gaussian noise is added to the watermark model, the watermark information still can be seen clearly. After a 1% simplification of the watermark model, the watermark on the model shown in



(a) Mesh mode complete model (b) Magnified character part (c) Point cloud mode

FIGURE 6. Rabbit model embedded in ” 3D”



(a) Mesh mode complete model (b) Magnified pattern part (c) Point cloud mode

FIGURE 7. Rabbit model embedded in the school badge pattern

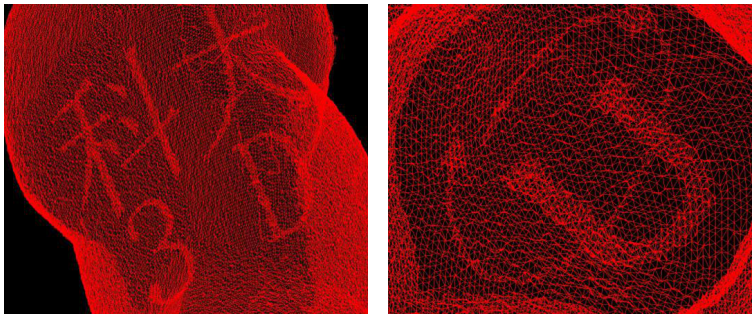
(a) Mesh mode complete model (b) Magnified pattern part (c) Point cloud mode

FIGURE 8. Horse model embedded in” 3D”

(a) Mesh mode complete model (b) Magnified pattern part (c) Point cloud mode

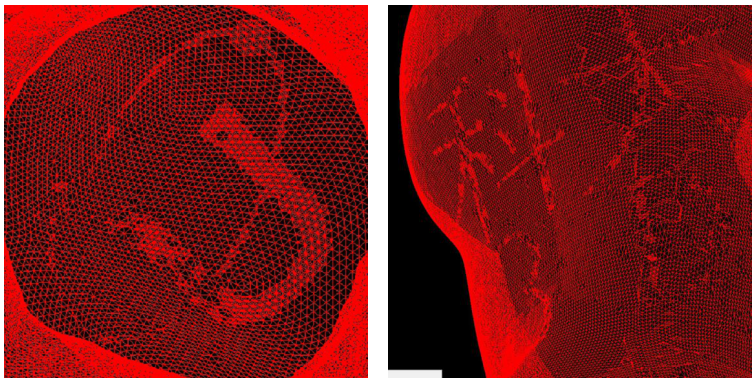
FIGURE 9. Horse model embedded in the school badge pattern

Figure 11 is partially incomplete, but the watermark information can still be seen clearly.



(a) horse model (b) rabbit model

FIGURE 10. Watermarking model with Gaussian noise



(a) rabbit model (b) horse model

FIGURE 11. Simplified 1% watermark model

In order to effectively illustrate the effects of the present paper, the experimental results are displayed and analyzed using tabular data to prove that the present paper has excellent performance.

Table 1 shows the changes in the numbers of vertices and patches of the three-dimensional mesh model after embedding different types of watermark information.

Table 1 shows that, after the watermark has been embedded, the increase in the number of vertices and patches of the model was very small, i.e., both were about 1%. It shows that the embedded watermark has made few changes to the model, which can ensure the copyright of the logo and the appearance of the model. It does not affect the normal use of the model.

TABLE 1. Two standard models

Model	Watermark information	Before embedding		After embedding		Increments(%)	
		Vertex	Patch	Vertex	Patch	Vertex	Patch
rabbit	科大3D	70658	141312	71525	143046	1.23	1.23
rabbit	pattern	70658	141312	71498	142992	1.19	1.19
horse	科大3D	193934	387864	197082	394160	1.62	1.62
horse	pattern	193934	387864	195613	391222	0.866	0.866

4. **Conclusion.** This article applies mainly to the copyright identification and data protection of 3D model files. The 3D model data must display the overall structure of the model and the watermark information embedded in the model on the corresponding software platform. After the model has been printed, no modification to the model can be seen. So it has achieved the role of copyright marking.

The advantage of this algorithm is that it can embed Chinese characters, English characters, and numbers as well as patterns and logos. There are fewer changes to the model when a watermark is being embedded, and it has certain robustness. Also, it can restore the model embedded in the watermark to the original lossless model, thereby achieving the reversible effect.

Acknowledgment. This work was supported by the Shanxi Natural Science Foundation under Grand (No. 201801D121129), National Natural Science Foundation of China (No.61672373, No.61501315), Scientific and Technological Innovation Team of Shanxi Province (No.201705D131025), Key Innovation Team of Shanxi 1331 Project (2017015), Collaborative Innovation Centre of Internet+3D Printing in Shanxi Province (201708), Reasearch Project Supported by Shanxi Scholarship Council of China.

REFERENCES

[1] X. Rolland-Neviere, G. Doerr, and P. Alliez, Triangle surface mesh watermarking based on a constrained optimization framework, IEEE Trans. Inf.Forensics Security, vol.9, no.9, pp. 1491C1501, 2014.

[2] G. N. Pham, S. H. Lee, O. H. Kwon, and K. R. Kwon, *A 3D printing model watermarking algorithm based on 3D slicing and feature points*, Electronics, vol.7, no.2, p.23, 2018.

[3] R. Ohbuchi, H. Masuda, M. Aono, Watermarking three-dimensional polygonal models, Proceedings of the 5th ACM International Conference on Multimedia. New York: ACM Press, pp. 261-272,1997.

[4] J. Liu, Y. Wang, Y. Li, R. Liu, and J. Chen, A robust and blind 3D watermarking algorithm using multiresolution adaptive parameterization of surface, Neurocomputing, vol.237, pp. 304C315, 2017.

[5] Q. S. Ai, Q. Liu, Z. D. Zhou, L. Yang, and S. Q. Xie, A new digital watermarking scheme for 3D triangular mesh models, Signal Process., vol.89, no.11, pp. 2159C2170, 2009.

[6] Y. Wang, J. Liu, Y. Yang, D. Ma, and R. Liu, 3D model watermarking algorithm robust to geometric attacks, IET Image Process, vol.11, no.10, pp. 822C832, 2017.

[7] X. Feng, W. Zhang, and Y. Liu, Double watermarks of 3D mesh model based on feature segmentation and redundancy information, Multimedia Tools Appl, vol.68, no.3, pp. 497C515, 2014.

[8] S. M. Mun, H. U. Jang, D. G. Kim, S. Choi, and H. K. Lee, A robust 3D mesh watermarking scheme against cropping, in Proc. IC3D, Liege, Belgium, pp. 1C6, 2015.

[9] Y. Z. Zhan, Y. T. Li, X. Y. Wang, and Y. Qian, A blind watermarking algorithm for 3D mesh models based on vertex curvature, Zhejiang Univ. Sci. C, vol.15, no.5, pp. 351C362, 2014.

[10] R. Jiang, H. Zhou, W. Zhang, and N. Yu, Reversible data hiding in encrypted three-dimensional mesh models, IEEE Trans. Multimedia, vol.20, no.1, pp. 55C67, 2018.

[11] K. Wang, G. Lavoue, F. Denis, and A. Baskurt, Hierarchical watermarking of semiregular meshes based on wavelet transform, IEEE Trans. Inf. Forensics Security, vol.3, no.4, pp. 620C634, 2008.

[12] J. U. Hou, D. G. Kim, and H. K. Lee, Blind 3D mesh watermarking for 3D printed model by analyzing layering artifact, IEEE Trans. Inf. Forensics Security, vol.12, no.11, pp. 2712C2725, 2017.

[13] M. Hamidi, M. El Haziti, H. Cherifi, and D. Aboutajdine, A robust blind 3-D mesh watermarking based on wavelet transform for copyright protection, in Proc. ATSIP, Fez, Morocco, pp. 1C6, 2017.

[14] Liu, Jing , et al. A Watermarking Method for 3D Models Based on Feature Vertex Localization. IEEE Access (2018):1-1.

[15] H. Al-Khafaji and C. Abhayaratne, Graph Spectral Domain Blind Watermarking, ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, United Kingdom, pp. 2492-2496, 2019.

[16] S.W. Weng., Y.-J. Liu, J.-S. Pan and N. Cai, Reversible data hiding based on flexible block-partition and adaptive block-modification strategy, Journal of Visual Communication and Image Representation, vol. 41, pp. 185-199, 2016.

[17] S. Weng, Y. Zhao, J. Pan and R. Ni, A Novel Reversible Watermarking Based on an Integer Transform, IEEE International Conference on Image Processing, San Antonio, TX, pp. 241-244, 2007.
[18] E. K. Wang, C. M. Chen, M. M. Hassan, A deep learning based medical image segmentation technique in Internet-of-Medical-Things domain, Future Generation Computer Systems, pp. 135-144, 2020.
[19] E. K. Wang, X. Zhang, F. Wang, T. Wu and C. Chen, Multi-layer dense attention model for image caption, in IEEE Access, vol. 7, pp. 66358-66368, 2019, doi: 10.1109/ACCESS.2019.2917771.
[20] C. Lu, C. Q. Zhu, Y. H. Wang, Visible watermarking of 3D mesh model in dual view. Journal of Image and Graphics, pp. 1068-1073, 2014.
[21] X. C. An, R. R. Ni, Y. Zhao, Visible watermarking of 3D model based on mesh subdivision and boundary adaptation [J]. Journal of Applied Sciences, pp. 503-514, 2016.