applied sciences

MDPI

Article

Security Protection of 3D Models of Oblique Photography by Digital Watermarking and Data Encryption

Yaqin Jiao [1,†], Cong Ma [2,†], Juhua Luo [1] and Yinguo Qiu [1,*]

[1] Key Laboratory of Watershed Geography, Nanjing Institute of Geography and Limnology, Chinese Academy of Sciences, Nanjing 210008, China; ttoa0718@163.com (Y.J.); jhluo@niglas.ac.cn (J.L.)
[2] School of Marine Technology and Geomatics, Jiangsu Ocean University, Lianyungang 222005, China; mctjuly@163.com
* Correspondence: ygqiu@niglas.ac.cn
† These authors contributed equally to this work and should be considered co-first authors.

Abstract: To clarify the copyrights of 3D models of oblique photography (3DMOP) and guarantee their security, a novel security protection scheme of 3DMOP was proposed in this study by synergistically applying digital watermarking and data encryption. In the proposed scheme, point clouds were clustered first, and then the centroid and feature points of each cluster were calculated and extracted, respectively. Afterward, the watermarks were embedded into the point clouds cluster-by-cluster, taking distances between feature points and centroids as the embedding positions. In addition, the watermarks were also embedded using texture coordinates of 3DMOP to further enhance the robustness of the watermarking algorithm. Furthermore, Arnold transformation was performed on texture images of 3DMOP for security protection of classified or sensitive information. Experimental results have verified the strong imperceptibility and robustness of the proposed watermarking algorithm, as well as the high security of the designed data encryption algorithm. The outcomes of this work can refine the current security protection methods of 3DMOP and thus further expand their application scope.

Keywords: 3D models of oblique photography; digital watermarking; data encryption; security protection

check for updates

Citation: Jiao, Y.; Ma, C.; Luo, J.; Qiu, Y. Security Protection of 3D Models of Oblique Photography by Digital Watermarking and Data Encryption. *Appl. Sci.* **2023**, *13*, 13088. https://doi.org/10.3390/app132413088

Academic Editor: David Megías

Received: 7 November 2023
Revised: 3 December 2023
Accepted: 3 December 2023
Published: 7 December 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the rapid development of oblique photogrammetry technology based on unmanned aerial vehicle (UAV), 3D models of oblique photography (3DMOP) are playing an increasingly significant role in the establishment of digital cities [1,2] and twin watersheds [3], due to their huge advantages such as high accuracy, clear texture information, virtual simulation capability, etc. Despite their significant utility, the security issues they face are becoming increasingly prominent [4]. Compared with other types of spatial data, 3DMOP have higher positioning accuracies of clearer texture information, in which classified elements (e.g., prisons and barracks) can be accurately located and sensitive information (e.g., car numbers and human portraits) may be exposed thoroughly [5]. When a leakage of this classified or sensitive information occurs, irreparable losses will be caused to data sharing, commercial interests, and even national security. Against this background, it is particularly important to take effective protective measures to prevent data theft and illegal dissemination. Digital watermarking [6] and data encryption [7], two representative technologies of data security protection, have been considered as effective means for security protection of 3DMOP [8].

Digital watermarking, an important branch of data hiding, has played an important role in security protection of digital products, such as digital images [9,10], CAD graphics [11,12], remote sensing images [13,14], vector maps [15,16], etc., and a huge amount of

research results have been obtained. However, few works have been carried out focusing on the security protection of 3DMOP.

While 3DMOP have played an important role in various fields, they are facing significant security issues. Traditional security protection methods of 3DMOP still have limitations, especially in the security protection of secret information in texture images. To make a breakthrough in this field, a novel security protection scheme of 3DMOP is proposed in this paper, the novelty of which can be summarized as follows:

(1) To enhance the robustness of the proposed watermarking scheme, point clouds are clustered and watermarks are embedded cluster-by-cluster, using slightly adjusted distances between centroids and feature points.
(2) Watermarks are embedded into both coordinates of point clouds and texture coordinates of 3DMOP, and double assurances of security of 3DMOP can be thus realized.
(3) Arnold transformation is carried out on texture images of 3DMOP, and secret and sensitive information in texture images can be prevented from being exposed.

2. Related Works

As a professional application of digital watermarking, 3DMOP's watermarking technology has become a research hotspot in recent years. According to the embedding location of the watermark, traditional research can be divided into two categories, i.e., algorithms based on 3D meshes [17–22] and 3D point clouds [23–28].

2.1. 3DMOP Watermarking Algorithm Based on 3D Mesh

The 3DMOP watermark algorithm based on a grid embeds the watermarks into the grid features. Zhou et al. (2007) proposed a robust digital watermarking algorithm for 3D mesh models based on wavelet transform [17], and Xing et al. (2009) proposed a 3D model digital watermarking algorithm based on DWT (discrete wavelet transform) and SVD [18]. The above two types of watermarking algorithms are robust against cropping and noise attacks but cannot resist translation and rotation attacks common in the application of 3DMOP. Soliman et al. (2015) used genetic algorithms to embed watermarks into selected points after K-means clustering, and the algorithm has strong robustness, but it is a non-blind watermarking algorithm and has poor practicality [19]. Zhan et al. (2014) proposed a robust watermarking algorithm for 3D mesh models based on vertex curvature, which uses the root mean square curvature of vertices for watermark embedding [20]. Zhang et al. (2014) extracted significant regions in 3D models, converted them to spherical coordinates, and performed wavelet transform to embed watermarks into low-frequency and high-frequency regions [21]. Algorithms in [20,21] have good robustness against translation, rotation, noise attacks, etc. Zhu et al. (2014) proposed a digital watermarking algorithm for 3D mesh models based on roughness [22], which selects candidate vertices through the normal vector and the angle between the weighted normal vector of the local geometric space closure-ring neighborhood. It can effectively resist simplification attacks, but the above three algorithms are not robust against cropping attacks.

2.2. 3DMOP Watermarking Algorithm Based on 3D Point Cloud

The 3DMOP watermark algorithm based on point clouds embeds watermarks by modifying the coordinates of the 3D point cloud. Wang et al. (2009) modified the integral invariants of some vertices by shifting the vertices at the vertex and its adjacent positions to embed the watermarking [23]. However, this algorithm is a semi-fragile watermarking, which can usually be used for data integrity authentication, but cannot be used for copyright protection of 3D models. Wu et al. (2012) selected the one-dimensional DWT low-frequency signal of the distance between each point in the point cloud model and the center of gravity of the model to embed watermarking [24], which is robust to cropping attacks, but less robust to common rotation and translation attacks. Feng et al. (2016) proposed a 3D point cloud algorithm based on angle quantization exponential modulation [25], which is robust to affine transformation, reordering, low-intensity noise, etc. Shang et al. (2015)

transferred the 3D model from a rectangular coordinate system to a spherical coordinate system [26] and used the distance from the vertex to the centroid as the embedding object to complete the watermark embedding, which is robust to some affine attacks. The above two algorithms are robust to translation and rotation attacks, but not robust to cropping attacks. Wang et al. (2018) proposed a 3D model blind watermarking algorithm based on distance mapping mechanism [27], which is robust to common attacks such as translation, cropping, and rotation, but the effect is poor under simplification and noise attacks. Gong et al. embedded watermarking based on the characteristic line ratio of the 3D model, and the algorithm is robust to geometric attacks and noise, but the effect is poor for cropping attacks [28]. In summary, there are still deficiencies in the robustness and applicability of 3D model digital watermarking algorithms.

In the past, research on oblique photography 3D models focused more on embedding copyright information and the loss of model accuracy during watermark embedding. When embedding watermarks, more attention is paid to embedding watermarks in model vertex coordinate data and model texture data, with less research on embedding watermarks in texture coordinate data [29,30]; at the same time, the security of texture information is also neglected. 3DMOP, a special type of geospatial data, have different characteristics with other types of spatial data. In these different characteristics, the most special one is real texture information. A lot of secret information, e.g., advertising boards, car numbers and human portraits, can be conveniently viewed by users, including illegal users. This threatens the security of 3DMOP greatly. However, traditional security protection methods of 3DMOP generally ignored this point. Accordingly, there is still much space for security improvement of 3DMOP.

3. Main Idea of 3DMOP Security Protection

A mesh model is a representation of a 3D model, which usually contains vertex data, texture coordinate data, normal data, face data etc. Together they constitute the basic geometry and texture of a mesh model.

Vertex coordinates are the coordinates of each vertex in a 3D space; texture coordinate data is usually used to determine how the texture fits on the surface of the model; normal vectors are used to describe the orientation of each vertex or face, affecting the lighting and rendering of the model; face data defines the polygon faces of the model, usually consisting of vertex indices and texture coordinate indices. When correctly constructing the mapping relationship between face data and texture data, texture data is a two-dimensional image that contains information such as the color and texture of the model surface.

As shown in Figure 1, this article encrypts the vertex coordinate data, texture coordinate data, and texture image data in the mesh model separately (i.e., use the vertex coordinate watermark algorithm for vertex coordinate data, and use the texture coordinate watermark algorithm for texture coordinate data). The next section further introduces the principles and implementation details of related encryption algorithms.

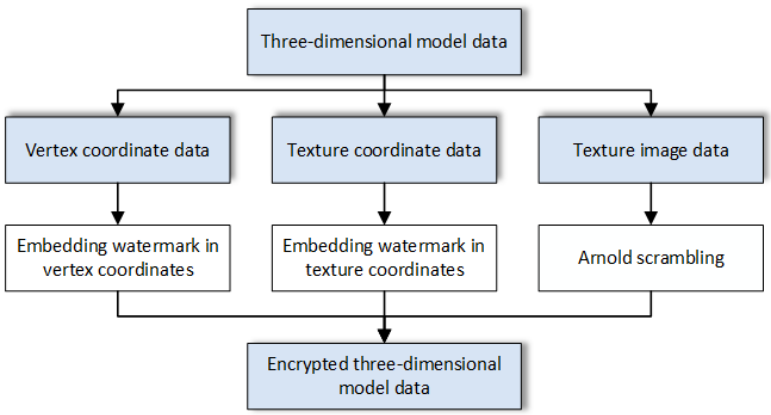


Figure 1. Main idea of the 3DMOP security protection.

4. The Proposed Scheme

Mesh models are a representation of 3D models that typically contain vertex data, texture coordinate data, normal data, face data, and other data that together form the basic geometric shape and texture of the mesh model. As shown in Figure 1, different encryption methods are used for vertex coordinate data, texture coordinate data, and texture images in this article. The next section further introduces the principles and implementation details of related encryption algorithms.

4.1. Digital Watermarking Algorithm of Vertex Coordinates

4.1.1. Watermark Embedding

The embedding process of the vertex coordinate watermark is shown in Figure 2.

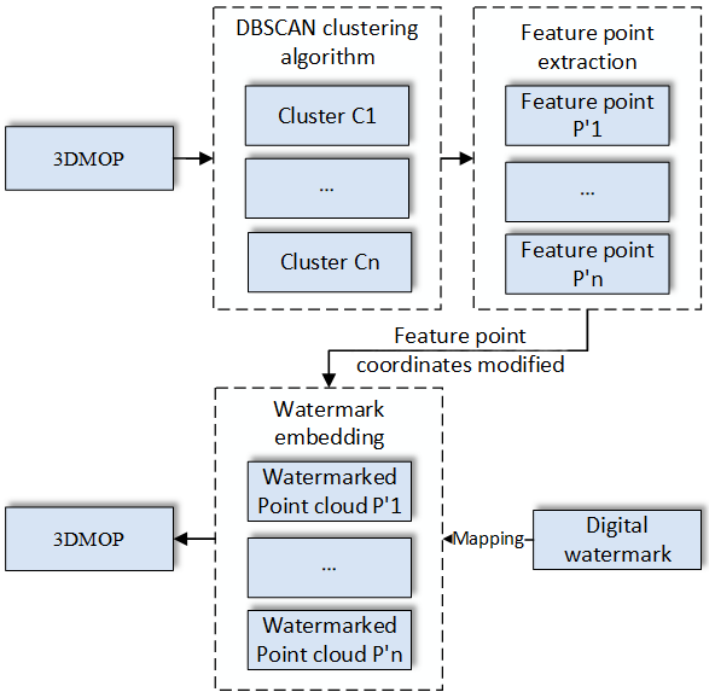


Figure 2. Flow chart of embedding the vertex coordinate watermark.

This section first extracts the vertex coordinates of the oblique photography 3D model, then performs clustering algorithm processing on the vertex coordinate data, divides it into several clustering clusters, extracts the feature points in the clustering clusters, and establishes the mapping relationship between the feature points and the watermark data, thus completing the embedding of the watermark. The process of embedding the vertex coordinates watermark is shown in Figure 2, which can be summarized as follows:

Step 1. Preprocess the oblique photography 3D model data, extract the vertex coordinates in the oblique photography 3D model, and obtain the data of the point cloud set $P = \{P_1, P_2, P_3, \ldots, P_n\}$, where $P_i(X_i, Y_i, Z_i)$ is the vertex coordinate data, and $i \in [1, N]$, N is the number of vertices.

Step 2. The cluster obtained after processing by the clustering algorithm is represented as $C = \{C_1, C_2, C_3, \ldots C_n\}$. The disordered point cloud in the cluster is processed by establishing a KD tree (K Dimension Tree) to obtain the K-neighborhood of the target point, thus improving the efficiency of point cloud data retrieval. Then, the point cloud density is calculated, and the point cloud is downsampled according to the density information to obtain a subset of point clouds, thus completing the extraction of the corresponding feature points $P'_n = \{P'_1.P'_2.P'_3, \ldots P'_i\}$ in the cluster.

Step 3. Taking a single cluster as an example, calculate the virtual centroid coordinates $(\overline{X},\ \overline{Y},\ \overline{Z})$ in the P'_n cluster based on the feature points, as calculated in Equation (1):

$$\begin{cases} \overline{X} = \frac{\sum_i^n x_i}{n} \\ \overline{Y} = \frac{\sum_i^n y_i}{n} \\ \overline{Z} = \frac{\sum_i^n z_i}{n} \end{cases} \tag{1}$$

calculate the distance between the feature points $P'_n = \{P'_1.P'_2.P'_3,\ \ldots P'_i\}$ and the virtual centroid coordinates to obtain $D = \{D_1,\ D_2,\ D_3,\ \ldots D_i\}$, where i is the number of feature points.

Step 4. Establish a mapping relationship between the watermark data $w = \{w_{(i,j)},\ 0 \le i \le h-1,\ 0 \le j \le h-1\}$ and the distance D between the feature point and the virtual centroid. For the m-th feature point P'_m, use Equation (2) to calculate the row and column numbers $r,\ l$ of its corresponding watermark value w:

$$\begin{cases} D_m = \sqrt{\left(X'_m - \overline{X}\right)^2 + \left(Y'_m - \overline{Y}\right)^2 + \left(Z'_m - \overline{Z}\right)^2} \\ d_m = trunc(D_m, p) \\ d'_m = Int\left(D_m * 10^6\right) \\ r, l = divmod\left(Hash(d_m),\ h^2\right) \end{cases} \tag{2}$$

In Equation (2), $trunc(a, b)$ means to retain b digits after the decimal point for a (choosing to retain four digits after the decimal point and calculating the mapping relationship through the hash value), $Int(*)$ represents taking the integer part of a decimal number, $Hash(*)$ represents a function that converts an input of any length into a fixed length output, $divmod(a, b)$ represents the quotient and remainder, where a is the divisor and b is the dividend and d_m represents the distance between the feature point with the number of digits retained after the decimal point and the virtual centroid coordinate point. For D_m, the selection of the number of retained digits directly affects the robustness and imperceptibility of the watermark; the smaller the number of retained digits, the higher the watermark strength, but the weaker the imperceptibility. The number of bits embedded in this section of the watermark is selected to be six digits after the decimal point after multiple experiments.

Step 5. According to the corresponding position of r, l in w, the watermark value $w_{[r,l]}$ corresponding to the feature point is obtained.The watermark value $w_{[r,l]}$ and D_m are modified according to Equations (3) and (4) on the original feature point coordinates, thus completing the watermark embedding.

$$dis = \begin{cases} 10^{-6},\ \ g(d'_m) \neq w_{[r,l]} \\ 0,\ \ g(d'_m) = w_{[r,l]} \end{cases} \tag{3}$$

$$\begin{cases} x' = x + \left(\frac{\overline{X} - x}{\sqrt{\left(\overline{X}-x\right)^2} + \sqrt{\left(\overline{Y}-y\right)^2} + \sqrt{\left(\overline{Z}-z\right)^2}} \right) * dis \\ y' = y + \left(\frac{\overline{Y} - y}{\sqrt{\left(\overline{X}-x\right)^2} + \sqrt{\left(\overline{Y}-y\right)^2} + \sqrt{\left(\overline{Z}-z\right)^2}} \right) * dis \\ z' = z + \left(\frac{\overline{Z} - z}{\sqrt{\left(\overline{X}-x\right)^2} + \sqrt{\left(\overline{Y}-y\right)^2} + \sqrt{\left(\overline{Z}-z\right)^2}} \right) * dis \end{cases} \tag{4}$$

4.1.2. Watermark Extraction

As illustrated in Figure 3, the watermark extraction process follows the same data processing approach as the watermark process. The watermark extraction process can be considered the inverse of the embedding process.

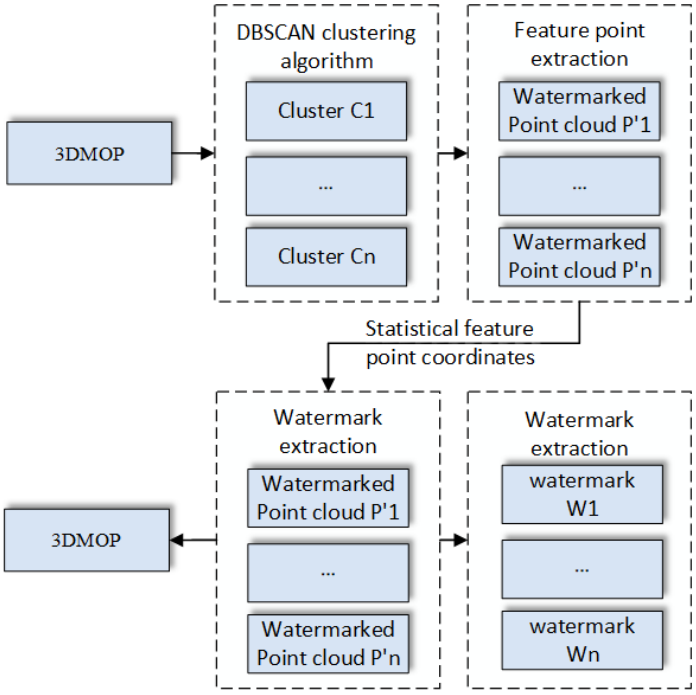


Figure 3. The vertex coordinates of the watermark extraction flow chart.

First, extract the vertex coordinates from the 3D model. The vertex coordinate dataset is denoted as $P = \{P_1,\ P_2,\ P_3,\ \ldots P_n\}$, where $P_i(X_i, Y_i, Z_i)$ represents the coordinates of the vertices, with $i \in [1,\ N]$, and N being the total number of vertices. Next, apply the DBSCAN algorithm to perform clustering on the dataset P, dividing it into multiple clusters denoted as $C = \{C_1,\ C_2,\ C_3,\ \ldots C_n\}$. Subsequently, extract feature points from each point cloud cluster and compute the virtual centroids of different clusters. Calculate the distances of feature points to the virtual centroids, resulting in a distance set $D = \{D_1,\ D_2,\ D_3,\ \ldots D_i\}$. Establish a mapping relationship between the distances D and the extracted watermark w' based on Equation (2). The extraction process is performed as illustrated in Equation (5):

$$w'_{[r,l]} = \begin{cases} 1, & \left(\left[d'_m * 10^6\right]\right) mod2 \neq 0 \\ 0, & \left(\left[d'_m * 10^6\right]\right) mod2 = 0 \end{cases} \tag{5}$$

From all the clusters $C = \{C_1,\ C_2,\ C_3,\ \ldots C_n\}$, extract $W' = \{W'_1,\ W'_2,\ W'_3,\ \ldots W'_m\}$ for each cluster. In this case, statistical methods are used to select the optimal values. There is a many-to-one relationship between the feature points and watermark positions, meaning that multiple watermark values correspond to the same r, l value. To extract the watermark information, the proportion of identical values in r, l is calculated. The most frequent result value is assigned as the watermark value for that row and column, thus completing the watermark information extraction.

4.2. Digital Watermarking Algorithm of Texture Coordinates

4.2.1. Watermark Embedding

The watermark embedding algorithm process is illustrated in Figure 4.

This section extracts the texture coordinates of the oblique photography 3D model and embeds the watermark by modifying the texture coordinates. The process of embedding the watermark is shown in Figure 4 and can be summarized as follows:

Step 1. First, the texture coordinate data needs to be preprocessed. The texture coordinate data extracted from the 3D model data is represented as $T = \{T_1,\ T_2,\ T_3,\ \ldots T_n\}$, where $T_i(U_i,\ V_i)$ is the texture coordinate, $i \in [1,\ N]$, and N is the number of texture coordinates. The texture coordinate values are usually between [0, 1], indicating the position on the texture image.

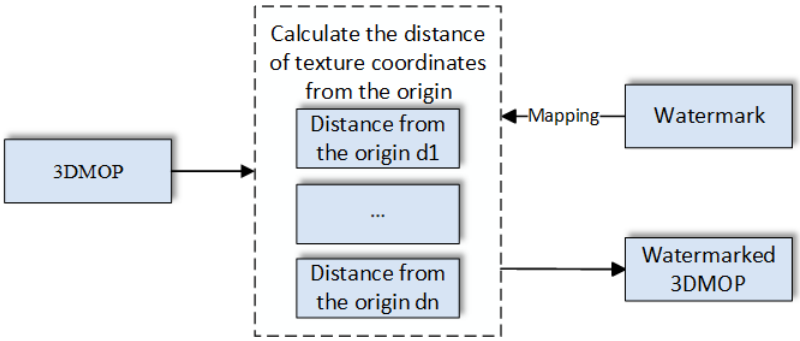


Figure 4. Flow chart of the texture coordinate watermark embedding.

Step 2. Calculate the distance between each texture coordinate and the origin of the texture coordinates [0, 0], resulting in $D_T = \{D_{T1},\ D_{T2},\ C_{T3},\ \ldots D_{Ti}\}$, where i is the number of texture coordinates.

Step 3. The watermark data w is a binary matrix of $h * h$ dimensions. A mapping relationship is established between the watermark data $= \left\{w_{(i,j)},\ 0 \le i \le h-1,\ 0 \le j \le h-1\right\}$ and the distance D between the feature point and the virtual centroid. For the m-th texture coordinate $T_m(U_m, V_m)$, the corresponding row and column numbers r, l in the watermark w are calculated according to Equation (6).

$$\begin{cases} D_{Tm} = \sqrt{(U_m)^2 + (V_m)^2} \\ d_{Tm} = trunc\left(D_{Tm} * 10^2,\ \ p\right) \\ d'_{Tm} = Int\left(D_{Tm} * 10^6\right) \\ r, l = divmod\left(Hash(d_{Tm}),\ \ h^2\right) \end{cases} \tag{6}$$

In Equation (6), $trunc(a, b)$ represents the truncation of a to b digits after the decimal point, $Int(*)$ represents the integer part of a decimal number, $Hash(*)$ represents a function that converts an input of arbitrary length into a fixed length output, $divmod(a, b)$ represents the quotient and remainder, where a is the divisor and b is the dividend, d_m represents the number of digits to retain after the decimal point for the distance D_{Tm} from the texture coordinate to the origin of the texture coordinate, and mod represents the remainder calculation.

Step 4. According to the corresponding position of r, l in w, obtain the watermark value $w_{[r,l]}$ corresponding to the feature point. The watermark value $w_{[r,l]}$ and D_m are modified according to Equations (7)–(9) on the original feature point coordinates, thus completing the watermark embedding.

$$g(x) = \begin{cases} 1, & x\ mod2\ \neq 0 \\ 0, & x\ mod2 = 0 \end{cases} \tag{7}$$

$$dis = \begin{cases} 10^{-6}, & g\left(d'_{Tm}\right)\ \neq w_{[r,l]} \\ 0, & g\left(d'_{Tm}\right) = w_{[r,l]} \end{cases} \tag{8}$$

$$\begin{cases} u' = u + \left(\frac{u}{\sqrt{(u)^2} + \sqrt{(v)^2}}\right) * dis \\ v' = v + \left(\frac{v}{\sqrt{(u)^2} + \sqrt{(v)^2}}\right) * dis \end{cases} \tag{9}$$

4.2.2. Watermark Extraction

As illustrated in Figure 5, the extraction procedure for watermarks follows the same methodology as the embedding process of watermarks. The process of watermark extraction can be perceived as the inverse operation of watermark embedding.

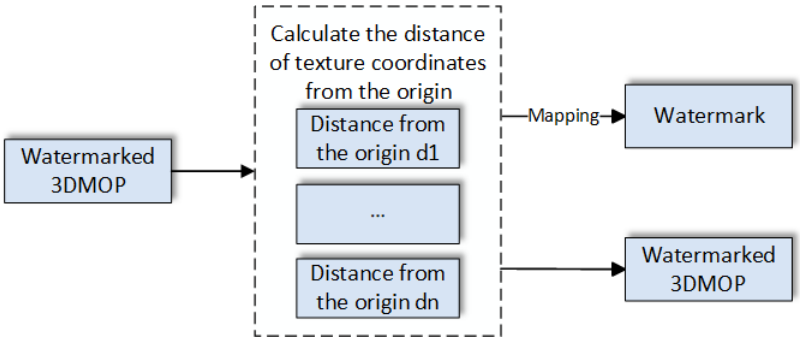


Figure 5. Flow chart of texture coordinate watermark extraction.

Initiate the data preprocessing by extracting texture coordinate data from the 3DMOP, represented as $T = \{T_1,\ T_2,\ T_3,\ \ldots T_n\}$, where $T_i(U_i,\ V_i)$ denotes the texture coordinates, with $i \in [1,\ N]$, and N indicating the number of texture coordinates. Texture coordinates typically manifest as values within the range of [0, 1], signifying positions on the texture image.

Calculate the distance of each texture coordinate from the texture coordinate origin [0, 0], resulting in $D_T = \{D_{T1},\ D_{T2},\ C_{T3},\ \ldots D_{Ti}\}$, where i represents the number of texture coordinates. The watermark data, denoted as w, is a binary matrix of size $h * h$. Establish a mapping relationship between the watermark data $w' = \left\{w'_{(i,j)},\ 0 \le i \le h-1,\ 0 \le j \le h-1\right\}$ and the distance D from the feature points to the virtual centroid. For the m-th texture coordinate $T_m(U_m,\ V_m)$, compute its corresponding watermark value $w'_{(i,j)}$ using Equations (10) and (11):

$$\begin{cases} D_{Tm} = \sqrt{(U'_m)^2 + (V'_m)^2} \\ d_{Tm} = trunc\left(D_{Tm} * 10^2,\ \ p\right) \\ r, l = divmod\left(Hash(d_m),\ \ h^2\right) \\ d'_{Tm} = Int\left(D_{Tm} * 10^6\right) \end{cases} \tag{10}$$

$$w'_{(r,l)} = \begin{cases} 1, & d'_{Tm} mod 2 \ \neq 0 \\ 0, & d'_{Tm}\ mod 2 = 0 \end{cases} \tag{11}$$

In Equation (10), $trunc(a, b)$ signifies the retention of b decimal places for a. $Int(*)$ denotes the extraction of the integer part of a decimal value. $Hash(*)$ is a function that transforms arbitrary length input into fixed length output. $divmod(a, b)$ signifies the division with a remainder, where a is the dividend, and b is the divisor. Here, d_m represents the distance D_{Tm} from the texture coordinate to the texture coordinate origin, with a specified number of decimal places retained, and mod signifies the modulo calculation.

Repeat the aforementioned procedure, iterating through all texture coordinates. There exists a many-to-one mapping relationship between texture coordinates and watermark positions, with one watermark position having multiple watermark values. Calculate the proportion of identical values in r and l, and assign the watermark values of the corresponding row or column based on the result with the highest count, thereby accomplishing the extraction of watermark information.

4.3. Algorithm of Texture Image Encryption

4.3.1. Arnold Transform Permutation Encryption

Arnold transformation can realize the displacement of image pixel positions and is widely used in image scrambling and image encryption [31]. The narrow sense Arnold transformation formula is shown in Equation (12), and the generalized Arnold transformation formula is shown in Equation (13):

$$\begin{bmatrix} x'_n \\ y'_n \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_n \\ y_n \end{bmatrix} mod(N) \tag{12}$$

$$\begin{bmatrix} x'_n \\ y'_n \end{bmatrix} = \begin{bmatrix} 1 & b \\ a & ab+1 \end{bmatrix} \begin{bmatrix} x_n \\ y_n \end{bmatrix} mod(N) \tag{13}$$

In Equations (12) and (13), (x_n, y_n) is the position coordinate of the pixel point in the original image, (x'_n, y'_n) is the corresponding coordinate of the transformed pixel point (x_n, y_n), mod is the modulo operation, and parameters a, b are positive integers. N is the end of the digital image matrix. The Arnold transformation in a narrow sense is also periodic. For an $N * N$ image, the generalized Arnold transformation is not periodic, so the number of permutations in the algorithm can be selected between 1 and N.

4.3.2. Arnold Transform Permutation Recovery

Arnold permutations require the inverse of the matrix in the permutation algorithm, and the matrices T and T^{-1} are taken as shown in Equations (14) and (15)

$$T = \begin{bmatrix} 1 & a \\ b & ab+1 \end{bmatrix} \tag{14}$$

$$T^{-1} = \begin{bmatrix} ab+1 & -b \\ -a & 1 \end{bmatrix} \tag{15}$$

where, a, b are positive integers. The original coordinates $(x_n,\ y_n)$ are obtained by applying Arnold permutations to the transformed coordinates (x'_n, y'_n) as shown in Equation (16).

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} ab+1 & -b \\ -a & 1 \end{bmatrix} \begin{bmatrix} x'_n \\ y'_n \end{bmatrix} mod(N) \tag{16}$$

5. Experimental Results and Analysis

Ten sets of 3DMOP data in OBJ format were used as the original data to test the performance of the algorithm. Table 1 shows the configuration of the experimental computer, and Table 2 lists the basic attributes of the ten sets of data, including the number of vertex coordinates, texture coordinates, and patch numbers.

Table 1. Experimental computer configuration.

Hardware Specifications	CPU	System	RAM	Hard Disk	Graphics Card
Model specifications	8 cores 2.9 GHz	Windows	16 GB DDR4	512 GB	GeForce GTX 1650

Table 2. Basic information of the experimental data.

Model	Number of Vertex Coordinates	Number of Texture Coordinates	Polygon Count
M-1	409,363	517,477	812,385
M-2	434,616	544,407	862,869
M-3	306,074	369,923	607,361
M-4	354,881	447,715	702,367
M-5	214,421	270,264	424,207
M-6	411,733	522,175	816,387
M-7	364,746	447,715	723,858
M-8	263,998	333,056	522,892
M-9	303,368	383,885	600,056
M-10	452,311	577,170	897,153
Average	351,551	441,378	696,953

When embedding watermarks into vertex coordinates, it is necessary to classify, partition, or group the vertex coordinate data to enhance the robustness of the watermarking algorithm [32]. Therefore, this article compared several clustering algorithms, and the results are shown in Figure 6. Finally, the DBSCAN clustering algorithm was used to cluster the vertex coordinate data to improve the robustness and stability of the vertex coordinate watermark embedding algorithm.

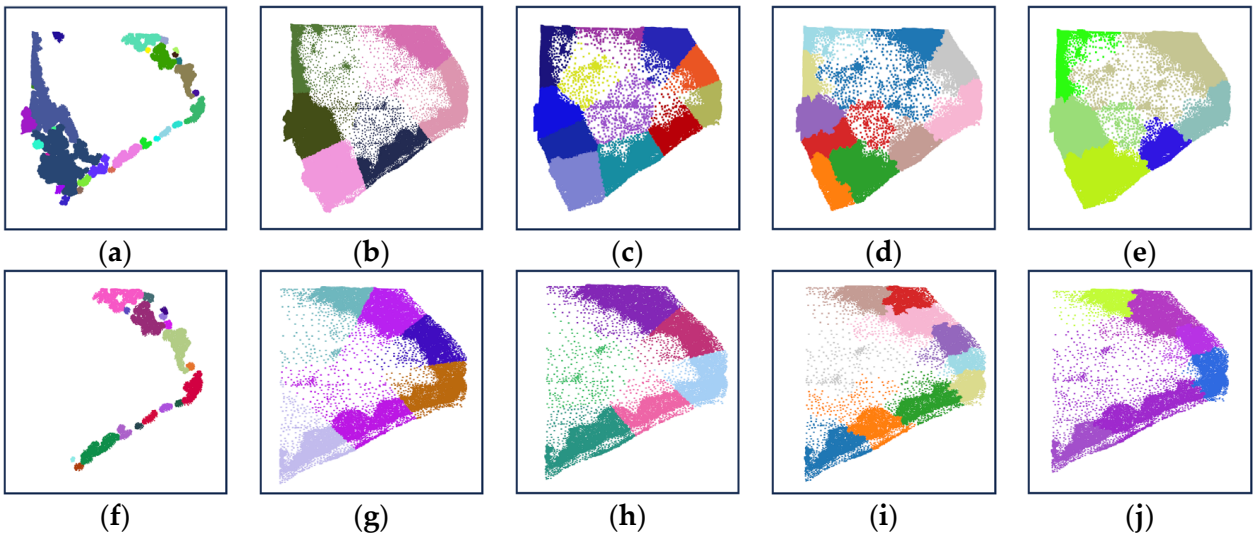


Figure 6. Results before and after clustering with different clustering algorithms, with different colors representing distinct clusters. (**a**,**f**) show the clustering results before and after pruning using the DBSCAN clustering algorithm, (**b**,**g**) represent the results before and after pruning with the K-means clustering algorithm, (**c**,**h**) display the results before and after pruning with the mean-shift clustering algorithm, (**d**,**i**) demonstrate the results before and after pruning using the hierarchical clustering algorithm, and (**e**,**j**) showcase the results before and after pruning with the agglomerative clustering algorithm.

The same data was processed before and after clipping using the same clustering algorithm. The DBSCAN clustering algorithm showed minor change in clustering clusters after clipping. Comparative analysis of the clustering clusters obtained by the clustering algorithm showed that the DBSCAN clustering algorithm had less influence on the clustering results after clipping, as shown in Table 3.

Table 3. Number of clusters before and after data clipping by different algorithms.

Clustering Algorithm	Number of Clustering Clusters in the Original Data	Number of Clustering Clusters in the Cropped Data	Number of Identical Clustering Clusters
DBSCAN	30	17	17
K-means	18	18	0
Mean-shift	13	6	1
Hierarchical	10	10	0
Agglomerative	10	10	0

5.1. Analysis of Imperceptibility

To quantitatively analyze the errors introduced by the model data before and after embedding the watermark, three commonly used evaluation metrics can be employed. These metrics include the Hausdorff distance, signal-to-noise ratio (SNR), and peak signal-to-noise ratio (PSNR). These metrics facilitate a quantitative evaluation between the original model and the watermarked model. The Hausdorff distance method is a commonly used

method for comparing the similarity between two data sets. The Hausdorff distance $H(A, B)$ between the original data A and the watermarked data B was defined as shown in Equation (17). The Hausdorff distance $H(A, B)$ in Equation (18) is the single Hausdorff distance from set A to set B, which first calculated the distance between each point a_i in set A and the point b_i with the minimum distance from the point in set B, and then sorted the calculated distance set. The maximum value in the sequence was taken as the value of $H(A, B)$. Equation (19) represents the calculation of $H(B, A)$, which is similar to Equation (18). The smaller the Hausdorff distance $H(A, B)$, the smaller the error of the model, and the better the imperceptibility of the watermark.

$$H(A, B) = max\ (h(A,\ B),\ \ h(B, A)) \tag{17}$$

$$h(A,\ B) = \max_{a \in A}\ \min_{b \in B} \|a - b\| \tag{18}$$

$$h(B,\ A) = \max_{b \in B}\ \min_{a \in A} \|b - a\| \tag{19}$$

The original coordinates were considered as normal signals, and the data that changed after embedding the watermark were considered as noise. The signal-to-noise ratio (SNR) reflects the ratio between the strength of the normal signal and the noise, and the peak signal-to-noise ratio (PSNR) can reflect the ratio between the maximum signal strength and the noise strength. The larger the values of SNR and PSNR, the smaller the change in the data caused by embedding the watermark, which indicates that the imperceptibility of the watermark is better. The calculation methods of SNR and PSNR are shown in Equations (20) and (21).

$$SNR = 10log_{10}\left(\sum_{i=0}^{N}\|Z_i\|^2 / \sum_{i=0}^{N}\left\|Z_i' - Z_i\right\|^2\right) \tag{20}$$

$$PSNR = 10log_{10}\left(\sum_{i=0}^{N}\|Z_i\|^2 / \frac{1}{N}\sum_{i=0}^{N}\left\|Z_i' - Z_i\right\|^2\right) \tag{21}$$

Using the above three parameters to quantitatively analyze the data before and after embedding the watermark, the calculation results are shown in Table 4.

Table 4. Imperceptibility evaluation.

Model	Vertex Coordinates			Texture Coordinates		
	SNR	PSNR	Haousdorff	SNR	PSNR	Haousdorff
M-1	182.800	184.579	1×10^{-6}	121.259	126.085	1×10^{-6}
M-2	183.972	185.334	1×10^{-6}	121.070	125.944	1×10^{-6}
M-3	183.731	185.263	1×10^{-6}	120.449	125.234	1×10^{-6}
M-4	165.106	149.681	1×10^{-6}	121.806	126.648	1×10^{-6}
M-5	170.467	174.429	1×10^{-6}	121.420	126.067	1×10^{-6}
M-6	171.666	174.594	1×10^{-6}	121.848	126.489	1×10^{-6}
M-7	169.355	152.655	1×10^{-6}	121.755	126.459	1×10^{-6}
M-8	185.329	186.308	1×10^{-6}	122.954	127.740	1×10^{-6}
M-9	182.411	181.094	1×10^{-6}	122.424	127.230	1×10^{-6}
M-10	164.001	166.509	1×10^{-6}	121.856	126.672	1×10^{-6}
Average	175.884	174.045	1×10^{-6}	121.684	126.456	1×10^{-6}

The threshold for PSNR and SNR was typically set at 28, and when the value is greater than 28, it is considered that the watermark performed well in imperceptibility [28]. In ten

experiments, the mean PSNR and SNR values for vertex coordinate watermarking before and after embedding were 174.045 and 175.884, respectively. For texture coordinate watermarking, the mean PSNR and SNR values before and after embedding were 121.684 and 126.456, respectively. These values meet the imperceptibility requirements. Additionally, a smaller Hausdorff distance implies greater dissimilarity between two sets. As indicated in Table 4, this algorithm exhibits good imperceptibility.

5.2. Analysis of Imperceptibility

To verify the robustness of the algorithm, ten datasets were subjected to common data attacks, and the algorithm proposed in this paper was applied to extract watermarks from the attacked data. The normalized correlation (NC) value was used as a quantitative evaluation of the extraction results [33]. NC was commonly used to measure the similarity between two two-dimensional images. In this paper, the original watermark-containing model and the extracted watermark information after attack were compared to reflect the robustness of the research experimental results. Generally, a correlation coefficient greater than 70% can be considered as indicating that the algorithm had a certain degree of resistance to the attack [28]. The NC calculation is shown in Equation (22).

$$NC = \frac{\sum_{i=1}^{L} w(i)w'(i)}{\sqrt{\sum_{i=1}^{L} w^2(i)}\sqrt{\sum_{i=1}^{L} w'^2(i)}} \tag{22}$$

In Equation (22), $w(i)$ is the original watermark data; $w'(i)$ is the extracted watermark data after the watermark attack; L is the length of the watermark sequence. The threshold value selected in this article was 0.8. When the NC value is greater than this value, it is considered that the watermark robustness is strong. Compared with the two-dimensional model, the attack methods for the 3DMOP were more complex and richer. In this study, common operations such as translation, cropping, rotation, noise attack, and simplification attack were used to perform attack experiments on the model.

5.2.1. Watermark Extraction from the Unattacked Model

The embedded watermark data is a 48 × 48 pixel binary image. The watermark extraction is performed on the unattacked 3D model data after embedding the watermark. The extracted watermark information is shown in Figure 7. When the model data is not attacked, the extraction result is consistent with the embedded image.

(**a**) (**b**)

Figure 7. Embedded watermark (**a**) and extracted results (**b**).

5.2.2. Model Translation and Rotation

In the practical use of 3D models, operations such as translation and rotation are often performed. Therefore, performing translation and rotation attack testing was a significant factor in detecting the robustness of watermarking algorithms. In the experiment, the watermarked data was translated as a whole, and the watermark was detected. The results are shown in Table 5 where it can be seen that the watermark information can be completely detected.

Table 5. Normalized correlation coefficients after translation and rotation attacks.

3DMOP	Translation (μ, Unit: meter)			Rotate (θ, Unit: °)		
	1.5	3	15	3.25	30	90
M-1	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0
M-2	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0
M-3	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0
M-4	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0
M-5	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0
M-6	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0
M-7	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0
M-8	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0
M-9	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0
M-10	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0
Average	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0	NC = 1.0

5.2.3. Model Cropping

Watermark data was independently and repeatedly embedded in the data, so after cropping the data, as long as some of the watermark embedding positions remain intact, the watermark information can be detected. The experiment cropped the watermarked data to different degrees, and the experimental results are shown in Tables 6 and 7. As shown in Table 7, when the cropping ratio was 70%, the vertex watermark algorithm was affected to varying degrees, but the copyright information can still be properly extracted; the texture watermark algorithm was basically unaffected. The experimental results showed that the algorithm in this paper was robust to cropping attacks.

5.2.4. Randomly Adding and Deleting Points in the Model

A fixed proportion of random addition and deletion points were attacked on the model, with a random addition proportion of 10–30% and a deletion proportion of 1–10%. The experimental results are shown in Table 8. After applying noise attacks to the model, there was a certain degree of accuracy loss. The embedding feature selected for embedding the vertex coordinates watermark is the distance between the feature point and the virtual centroid. The coordinates of the virtual centroid depend on the coordinates of the feature point, so the vertex coordinates watermark is not affected when a small number of point clouds are added. When the proportion of point attacks exceeds 30%, the vertex coordinate watermark NC value is less than 0.8, which is not robust to the attack, but still allows for the identification of copyright information. Deleting points has a small impact on the vertex coordinates watermark. The embedding feature selected for embedding the texture coordinates watermark is the global feature, and the mapping relationship between the texture coordinates and the watermark is a many-to-one mapping. Therefore, it is basically immune to the image attack of adding or deleting points.

Table 6. Normalized correlation coefficient after the clipping attack.

Cropping Ratio	Cropping Locations	NC for Vertex Coordinate Extraction Results	NC for Texture Coordinate Extraction Results	Extraction Successful
30%		0.9569688990161647	1.0	√
40%		0.951453182187509	1.0	√
50%		0.9403246919632546	1.0	√
60%		0.8944271909999159	1.0	√
70%		0.3899434844341313	1.0	√

Table 7. Normalized correlation coefficient after the clipping attack (ten sets of data).

Model		Cropping Ratio				
		30%	40%	50%	60%	70%
M-1	Vertex co-ordinates	0.973275	0.973275	0.973275	0.978778	0.972987
	Texture co-ordinates	1.0	1.0	1.0	1.0	1.0
M-2	-	0.956968	0.951453	0.940324	0.894427	0.389943
	-	1.0	1.0	1.0	1.0	1.0
M-3	-	1.0	1.0	1.0	0.994778	0.994778
	-	1.0	1.0	1.0	1.0	1.0
M-4	-	0.9895285	0.989528	0.978945	0.978945	0.9574271
	-	1.0	1.0	1.0	1.0	1.0
M-5	-	0.881917	0.881917	0.843274	0.843274	0.809663
	-	1.0	1.0	1.0	1.0	1.0
M-6	-	1.0	1.0	1.0	1.0	1.0
	-	1.0	1.0	1.0	1.0	1.0
M-7	-	0.984140	0.984140	0.99477	0.962513	0.945905
	-	1.0	1.0	1.0	1.0	1.0
M-8	-	0.827743	0.801314	0.801314	0.801314	0.801314
	-	1.0	1.0	1.0	1.0	1.0
M-9	-	0.957947	0.946484	0.946484	0.935414	0.935414
	-	1.0	1.0	1.0	1.0	1.0
M-10	-	1.0	0.973610	0.973610	0.973610	0.940965
	-	1.0	1.0	1.0	1.0	1.0
Average	-	0.957151	0.9501721	0.945199	0.934644	0.876500
	-	1.0	1.0	1.0	1.0	1.0

5.2.5. Comparison of Similar Algorithms

As shown in Figure 8, the amplitude range of noise attacks is 0–1%. The algorithm of Shang et al. [34] selects the distance from the point in the model to the center of the model as the embedding feature, so the image affected by noise attacks is larger. Gong et al. [28] select the ratio between the triangular network lines approximately perpendicular to the ground as the watermark embedding feature. Zhang et al. [35] embed watermarks in 3D point cloud data based on redundant discrete wavelet transform and singular value decomposition of the matrix, which is robust to noise attacks. The algorithm of this paper selects the distance from the texture coordinate to the origin of the texture coordinate as the embedding feature for the texture coordinate watermark and performs hash mapping. The vertex coordinate watermark performs clustering on the point cloud before embedding, and extracts feature points from the clustered clusters. Therefore, its robustness against noise attacks is better than other algorithms.

Table 8. Normalized correlation after the add–delete point attack.

Model		Random Addition of Points			Random Deletion of Points		
		10%	20%	30%	1%	5%	10%
M-1	Vertex coordinates	0.984140	0.956747	0.869417	0.973275	0.809885	0.687757
	Texture coordinates	1.0	1.0	1.0	1.0	1.0	1.0
	Extraction successful	√	√	√	√	√	√
M-2	-	1.0	1.0	0.989528	0.994778	0.994778	0.859990
	-	1.0	1.0	1.0	1.0	1.0	1.0
	-	√	√	√	√	√	√
M-3	-	1.0	0.994778	0.989528	0.994778	0.74579	0.298984
	-	1.0	1.0	1.0	1.0	1.0	1.0
	-	√	√	√	√	√	√
M-4	-	0.962851	0.957427	0.940965	0.929829	0.797130	0.381881
	-	1.0	1.0	1.0	1.0	1.0	1.0
	-	√	√	√	√	√	√
M-5	-	0.994428	0.960324	0.912949	0.918936	0.849836	0.781735
	-	1.0	1.0	1.0	1.0	1.0	1.0
	-	√	√	√	√	√	√
M-6	-	1	0.994778	0.994832	1	0.993832	0.989743
	-	1.0	1.0	1.0	1.0	1.0	1.0
	-	√	√	√	√	√	√
M-7	-	0.978720	0.911909	0.888523	0.978720	0.870571	0.361986
	-	1.0	1.0	1.0	1.0	1.0	1.0
	-	√	√	√	√	√	√
M-8	-	0.978778	0.989473	0.984140	0.989473	0.811472	0.65748
	-	1.0	1.0	1.0	1.0	1.0	1.0
	-	√	√	√	√	√	√
M-9	-	0.994778	0.984469	0.946484	1	0.823823	0.628899
	-	1.0	1.0	1.0	1.0	1.0	1.0
	-	√	√	√	√	√	√
M-10	-	0.992192	0.912870	0.872018	0.677003	0.456435	0.204124
	-	1.0	1.0	1.0	1.0	1.0	1.0
	-	√	√	√	√	√	√
Average	-	1.0	1.0	1.0	1.0	1.0	1.0
	-	0.988588	0.966277	0.938838	0.945679	0.815355	0.585258
	-	√	√	√	√	√	√

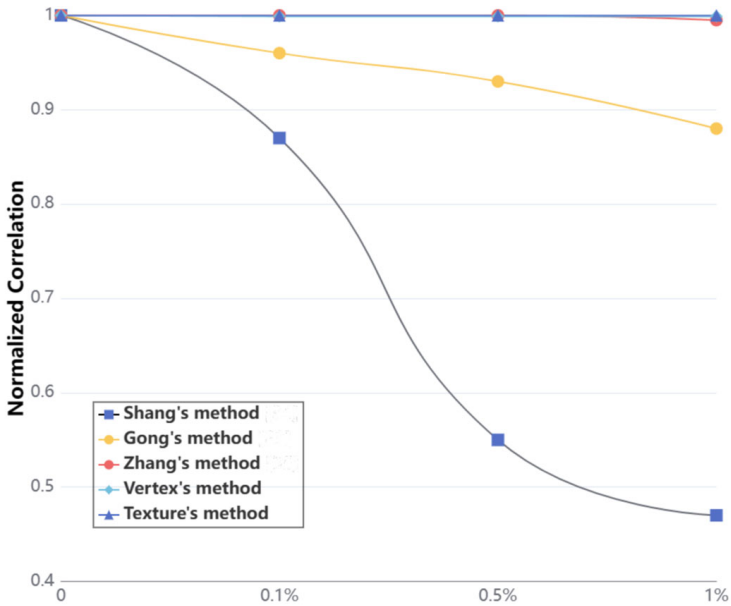


Figure 8. Robustness comparison of noise attacks [28,34,35].

Figure 9 shows the correlation coefficients of different algorithms after the cropping attack. The amplitude range of cropping attack is 10–40%. The algorithm of Shang et al. is more vulnerable to cropping attacks [34], while the algorithm of Wang et al. [27] selects the height value for sorting of point clouds in the 3D model as the watermark embedding feature. Therefore, the impact of cropping attacks on the algorithm of Gong et al. [28], the algorithm of Wang et al., and the algorithm of Zhang et al. [35] is relatively small in some areas. In this algorithm, the vertex coordinates are processed through clustering and feature point extraction, which has strong robustness against cropping, and is superior to similar 3D model spatial watermark algorithms. It maintains the same level of robustness as frequency domain algorithms. The embedded features in the texture coordinates establish a multi-to-one mapping relationship with the watermark data, so it is basically not affected by cropping, which is superior to other watermark algorithms.

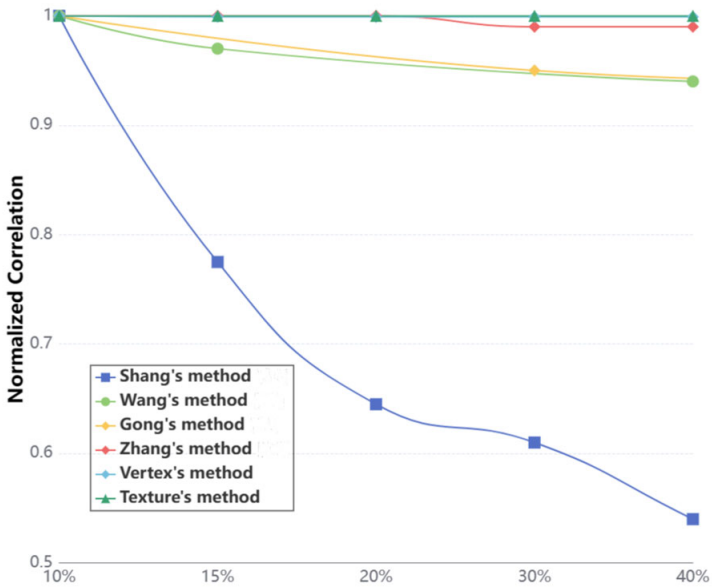
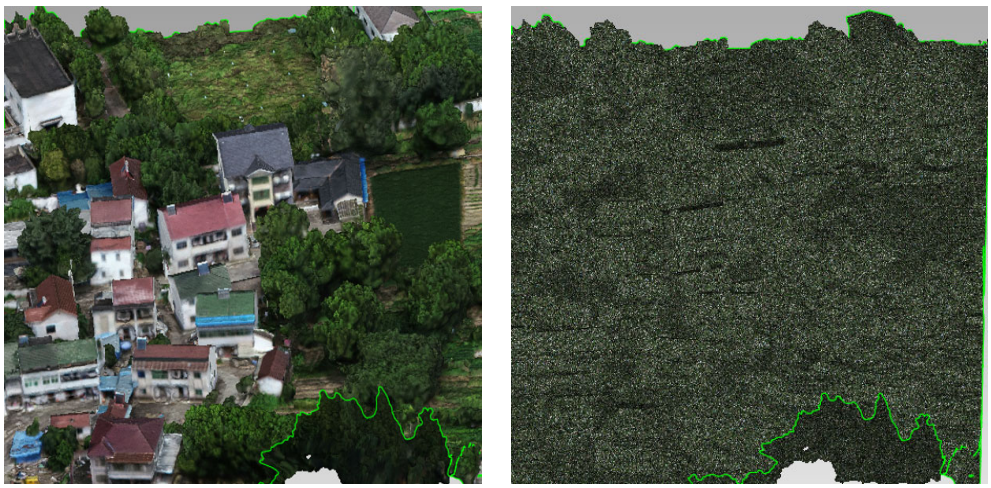


Figure 9. Robustness comparison of the clipping attacks [27,28,34,35].

5.3. Security Analysis

The Arnold scrambling algorithm based on the spatial domain achieves the purpose of encrypting images by changing the spatial position of pixel points and destroying the spatial position correlation of adjacent pixel points. As can be seen from Figure 10, after the algorithm scrambling, the information of the original image is completely hidden, and a good scrambling effect is visually achieved. The key space size of the Arnold permutations algorithm is related to the parameters a and b and the number of permutations. The selection of parameters a and b and the number of permutations has relatively few restrictions, and selecting appropriate values can ensure security of the image encryption. In certain cases, adjustments can be made as appropriate.



(**a**) (**b**)

Figure 10. The model before texture image encryption (**a**) and the model after texture image encryption (**b**).

5.3.1. Mean Squared Error

The similarity between the pre-encrypted image and the post-encrypted image is compared by calculating the mean squared error (MSE). MSE is a widely used quantitative metric in image processing and computer vision that measures the degree of difference between two images. The difference values of corresponding pixels in the two images are squared, and then the average of all squared difference values is taken. The mean squared error calculation is shown in Equation (23):

$$MSE = \frac{1}{N}\sum\left(P_{(i,j)} - P'_{(i,j)}\right)^2 \tag{23}$$

N signifies the total pixel count within the image. $P_{(i,j)}$ represents the pixel value at the original image's location (i, j), while $P'_{(i,j)}$ denotes the pixel value at the corresponding position i, j in the encrypted image. A smaller MSE indicates a higher similarity between the images. Conversely, a larger MSE implies greater dissimilarity between the images. According to Table 9, the MSE in this algorithm indicates a significant difference between the original and encrypted images, affirming the effectiveness of image encryption.

Table 9. Mean square error before and after image encryption.

Model	MSE
M-1	102.75754
M-2	104.24337
M-3	95.72083
M-4	102.57641

Table 9. *Cont.*

Model	MSE
M-5	102.65943
M-6	101.45861
M-7	102.39607
M-8	90.54710
M-9	96.78182
M-10	102.92381
Average	100.2065

5.3.2. Structural Similarity

The Structural Similarity Index (SSIM) is a widely utilized quantitative metric in the domains of image processing and computer vision.

$$SSIM(X,\ Y) = \frac{\left(2 * \mu_x * \mu_y + C1\right) * \left(2 * \sigma_{xy} + C2\right)}{\left(\mu_x^2 + \mu_y^2 + C1\right) * \left(\sigma_x^2 + \sigma_y^2 + C2\right)} \tag{24}$$

In Equation (24), X and Y respectively denote two images, μ_x and μ_y represent the mean luminance of these two images, σ_x and σ_y represent their luminance variances, and σ_{xy} denotes the luminance covariance between the two images. The constants $C1$ and $C2$ in the formula are typically set as:

$$\begin{cases} C1 = (k1 * L)^2 \\ C2 = (k2 * L)^2 \end{cases} \tag{25}$$

In Equation (25), $k1$ is a small positive constant (typically 0.01), $k2$ is another small positive constant (typically 0.03), and L represents the dynamic range of pixel values (for 8-bit images, L = 255). By assessing the similarity of images in terms of luminance, contrast, and structure, a comprehensive similarity metric is obtained, ranging from 0 to 1. A higher SSIM value closer to 1 indicates greater similarity between the two images, while a value closer to 0 suggests more significant dissimilarity. The experimental results are shown in Table 10. The SSIM mean value of ten sets of data is 0.01, which is significantly different from the original image and can serve as an encryption method.

Table 10. Structural similarity before and after image encryption.

Model	SSIM
M-1	0.0136642
M-2	0.0070133
M-3	0.007334
M-4	0.005984
M-5	0.021913
M-6	0.022779
M-7	0.0080442
M-8	0.01073335
M-9	0.00992666
M-10	0.0221195
Average	0.0129511

5.4. Efficiency Testing

The embedding time of watermarking under the same device is shown in Table 11. The embedding time of the vertex coordinate watermarking algorithm is slightly longer than that of the coordinate watermarking algorithm, but both algorithms can embed the watermarking within one minute.

Table 11. Time required for watermark embedding.

Model	Number of Vertex Coordinates	Vertex Coordinates Watermark	Number of Texture Coordinates	Texture Coordinate Watermark
M-1	409,363	54.738214	517,477	15.169614
M-2	434,616	43.703921	544,407	9.550489
M-3	306,074	29.185463	369,923	15.143082
M-4	354,881	48.079023	447,715	13.318522
M-5	214,421	20.260518	270,264	7.458641
M-6	411,733	44.232667	522,175	14.5824425
M-7	364,746	50.468684	447,715	12.992846
M-8	263,998	28.209486	333,056	9.155057
M-9	303,368	40.457431	383,885	11.133360
M-10	452,311	53.368458	577,170	16.566770
Average	351,551	41.27039	441,378	12.50708

6. Discussion

Compared to traditional 3D model watermarking algorithms, this article improves the current 3DMOP security protection method through the collaborative application of digital watermarking and data encryption, and further expands its application scope.

Compared with traditional 3D model watermarking algorithms, this paper uses clustering algorithms to group vertex coordinate data and perform multiple embeddings of vertex coordinate watermarks, improving the robustness of the vertex coordinate watermarking algorithm. We chose texture coordinates, which are rarely used in traditional watermarking algorithms, as the target for watermark embedding. Texture coordinates are used to determine how texture maps on the model surface are mapped to geometric coordinates and are less affected by watermarking attacks such as cropping, rotation, and point deletion. Experimental results show that texture coordinate watermarking can effectively resist various common attacks. In the past, research on oblique photography 3D models focused more on embedding copyright information, ignoring texture image data. Texture image data contains rich privacy information, including license plate numbers, advertising logos, face recognition data, and detailed information about surrounding buildings. Therefore, algorithm encryption is used to improve the security of texture data and ensure that sensitive information is not leaked by unauthorized access.

However, some aspects of the algorithm in this article still need further research and resolution. For example, for vertex coordinates, when performing common model watermark attacks, the sensitivity of different regions of data varies depending on the same proportion of point deletion and cropping. The different regions of action also have different results in extracting the vertex coordinate watermark. The 3DMOP data may undergo a data format conversion during use, but the watermark embedded in the texture coordinates will change after the 3D model format conversion, making it impossible to extract the copyright information normally.

In subsequent research, we will conduct in-depth research on the above issues, such as whether it is possible to modify the clustering algorithm to make the clustering clusters evenly distributed and not affected by common watermark attacks, further improving the robustness of watermark cropping/deletion; we will further study how to fully extract the texture coordinate watermark after the 3D model format conversion, thereby effectively improving the robustness of the texture coordinate watermark algorithm; when conducting security analysis on encrypted images, the structural similarity index can be evaluated in combination with the Diophantine equation fuzzy set.

7. Conclusions

In response to the copyright ownership and privacy information leakage issues faced by 3DMOP, a security protection strategy for 3DMOP is proposed. The security protection strategy for 3DMOP mainly consists of two aspects: (1) embedding of copyright informa-

tion in 3D models. When embedding copyright information, vertex coordinates and texture coordinates of 3D models are selected as embedding features for embedding separately. Compared to traditional 3D model spatial watermarking algorithms, the vertex coordinate watermark embedding algorithm in this paper improves the robustness of the watermark algorithm by clustering point cloud data in the data preprocessing part to obtain clustering clusters and extracting feature points in the clustering clusters. After common attacks on 3D models, the copyright information can still be completely extracted; using texture coordinate information as an embedding feature is less common in related oblique photography 3D model watermarking algorithms. After embedding the texture coordinate watermark, it is less affected by common watermark attack methods, and the robustness and imperceptibility of the watermark algorithm are strong. Embedding copyright information in both watermark embedding features enables complete extraction of copyright information after one watermark embedding feature is attacked. (2) Encryption of texture information in 3D models. Encrypted texture information is evaluated using MSE and SSIM, both of which have high security. This method improves the current security protection scheme for 3DMOP.

In future work, we will further study and optimize image encryption algorithms to improve the efficiency of image encryption, solve the problem of texture coordinate watermarking being prone to extraction failure after 3DMOP format conversion, and further enhance the robustness of watermarking algorithms.

Author Contributions: Conceptualization, Y.J. and Y.Q.; methodology, Y.Q., J.L. and Y.J.; software, Y.J.; validation, Y.J. and Y.Q.; formal analysis, Y.J.; investigation, Y.J.; resources, Y.Q.; data curation, Y.J. and C.M.; writing and original draft preparation, Y.J. and C.M.; writing—review and editing, Y.J. and Y.Q.; supervision, Y.Q.; project administration, Y.Q.; funding acquisition, Y.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant No. 42101433) and the Natural Science Foundation of Jiangsu Province (Grant No. BK20201100).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are available from the corresponding author upon request. The data are not publicly available due to privacy.

Acknowledgments: We would like to express our sincere gratitude to the anonymous reviewers for their kind comments and suggestions for improving the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, L.; Wu, Z.P.; Jiang, X.Y.; Chen, C. Application of the UAV oblique photography technique in 3D city modeling. *Geomat. Spat. Inf. Technol.* **2015**, *38*, 30–32.
2. Jeddoub, I.; Nys, G.-A.; Hajji, R.; Billen, R. Digital Twins for cities: Analyzing the gap between concepts and current implementations with a specific focus on data integration. *Comput. Graph.* **2023**, *122*, 103440. [CrossRef]
3. Qiu, Y.; Duan, H.; Xie, H.; Ding, X.; Jiao, Y. Design and development of a web-based interactive twin platform for watershed management. *Trans. GIS* **2022**, *26*, 1299–1317. [CrossRef]
4. Qiu, Y.; Gu, H.; Sun, J.; Duan, H.; Luo, J. Rich-information watermarking scheme for 3D models of oblique photography. *Multimed. Tools Appl.* **2019**, *78*, 31365–31386. [CrossRef]
5. Hadi, H.; Cao, Y.; Khaleeq, U.; Nisa, N.; Jamil, A.; Ni, Q. A comprehensive survey on security, privacy issues and emerging defence technologies for UAVs. *J. Netw. Comput. Appl.* **2023**, *213*, 103607. [CrossRef]
6. Liu, J.; Cui, H.; Dai, X. Three dimensional point clouds watermarking algorithm based on sphere degenerated octree. *Adv. Mater. Res.* **2011**, *314–316*, 2064–2070. [CrossRef]
7. Gao, S.; Wu, R.; Wang, X.; Wang, J.; Li, Q.; Wang, C.; Tang, X. A 3D model encryption scheme based on a cascaded chaotic system. *Signal Process.* **2023**, *202*, 108745. [CrossRef]
8. Li, H.; Zhu, H.H.; Hua, W.H.; Shang, H.; Liu, H.T.; Li, G.; Yang, F.; Song, J. Key Technologies and Methods for Vector Geographic Data Security Protection. *EarthScience* **2020**, *45*, 4574–4588.

9. Gupta, S.; Saluja, K.; Solanki, V.; Kaur, K.; Singla, P.; Shahid, M. Efficient methods for digital image watermarking and information embedding. *Meas. Sens.* **2022**, *24*, 100520. [CrossRef]
10. Evsutin, O.; Dzhanashia, K. Watermarking schemes for digital images: Robustness overview. *Signal Process. Image Commun.* **2022**, *100*, 116523. [CrossRef]
11. Peng, F.; Liu, Y.; Long, M. Reversible watermarking for 2D CAD engineering graphics based on improved histogram shifting. *Comput.-Aided Des.* **2014**, *49*, 42–50. [CrossRef]
12. Zhou, M.; Liu, Y.; Long, Z.; Chen, L.; Zhu, C. A reversible visible watermarking for 2D CAD engineering graphics based on graphics fusion. *Signal Process. Image Commun.* **2019**, *78*, 426–436.
13. Zhu, P.; Jia, F.; Zhang, J. A copyright protection watermarking algorithm for remote sensing image based on binary image watermark. *Optik* **2013**, *124*, 4177–4181. [CrossRef]
14. Li, M.; Zhang, J.; Wen, W. Cryptanalysis and improvement of a binary watermark-based copyright protection scheme for remote sensing images. *Optik* **2014**, *125*, 7231–7234. [CrossRef]
15. Qiu, Y.; Duan, H. A novel multi-stage watermarking scheme of vector maps. *Multimed. Tools Appl.* **2021**, *80*, 877–897. [CrossRef]
16. Ren, N.; Tong, D.; Cui, H.; Zhu, C.; Zhou, Q. Congruence and geometric feature-based commutative encryption-watermarking method for vector maps. *Comput. Geosci.* **2022**, *159*, 105009. [CrossRef]
17. Zhou, X.; Yu, X.M.; Zhu, T.; Wang, X.X.; Shi, J.Y. 3D mesh model watermarking algorithm based on wavelet transform. *Comput. Appl.* **2007**, *27*, 1156–1159.
18. Xing, J.J.; Liu, Q. 3D digital watermarking algorithm based on DWT and SVD. *J. Wuhan Univ. Technol.* **2009**, 107–109. [CrossRef]
19. Soliman, M.M.; Hassanien, A.E.; Onsi, H.M. A robust 3D mesh watermarking approach using genetic algorithms. *Adv. Intell. Syst. Comput.* **2015**, *323*, 731–741.
20. Zhan, Y.Z.; Li, Y.T.; Wang, X.Y.; Qian, Y. A blind watermarking algorithm for 3D mesh models based on vertex curvature. *J. Zhejiang Univ. C* **2014**, *15*, 351–362. [CrossRef]
21. Zhang, J.H.; Wen, X.B.; Lei, M.; Xu, H.X.; Qin, C.; Liu, J. Robust approach of 3D mesh watermarking in wavelet domain. *Comput. Eng. Appl.* **2014**, *50*, 98–102.
22. Zhu, L.L.; Zhang, J.X.; Wang, B. Digital watermarking for 3Dmesh model based on roughness. *J. Chongqing Univ. Technol.* **2014**, *28*, 87–91.
23. Wang, Y.P.; Hu, S.M. A new watermarking method for 3D models based on integral invariants. *IEEE Trans. Vis. Comput. Graph.* **2009**, *15*, 285–294. [CrossRef]
24. Wu, Y.B.; Geng, G.H.; He, Y. Digital watermark algorithm based on DWT for 3D point cloud model. *Comput. Eng.* **2012**, *38*, 151–152.
25. Feng, X. A new watermarking algorithm for point model using angle quantization index modulation. In *2015 4th National Conference on Electrical, Electronics and Computer Engineering*; Atlantis Press: Amsterdam, The Netherlands, 2016.
26. Shang, J.J.; Sun, L.J.; Wang, W.; Qin, Y.; Zhou, Z. Holographic digital blind watermark algorithm for 3D point cloud model based on discrete cosinine transform. *Packag. Eng.* **2015**, *13*, 111–114.
27. Wang, G.; Ren, N.; Zhu, C.Q.; Jing, M. The digital watermarking algorithm for 3D models of oblique photography. *J. Geo-Inf. Sci.* **2018**, *20*, 738–743.
28. Gong, W.T.; Zhu, C.Q.; Cul, H.C.; Gong, W.T.; Zhu, C.Q.; Cui, H.C.; Ren, N. Digital watermarking algorithm for oblique photography 3D model based on feature line proportion. *Sci. Surv. Mapp.* **2022**, *47*, 80–88, 152.
29. Zhao, H.C.; Kun, H.K.; Wang, X.C. Grayscale watermarking algorithm via BEMD and texture complexity. *J. Graph.* **2022**, *43*, 659.
30. Guo, N.; Huang, Y.; Niu, B.; Lan, F.; Niu, X.; Gao, Z. Double adaptive image watermarking algorithm based on regional edge features. *J. Xidian Univ.* **2023**. [CrossRef]
31. Huang, Q.M.; Arzugul·Hekim Li, G.D. Fast medical image encryption algorithm based on the combination of finitedomain and Arnold mapping. *Comput. Appl. Softw.* **2023**, *40*, 319–323.
32. Weng, S.; Liu, Y.; Pan, J.S.; Cai, N. Reversible data hiding based on flexible block-partition and adaptive block-modification strategy. *J. Vis. Commun. Image Represent.* **2016**, *41*, 185–199. [CrossRef]
33. Wang, K.; Lavoue, G.; Denis, F.; Baskurt, A. Robust and blind mesh watermarking based on volume moments. *Comput. Graph.* **2010**, *35*, 1–19. [CrossRef]
34. Shang, J.J.; Sun, L.J.; Wang, W.J. Blind watermarking algorithm base on SIFT for 3D point cloud model. *Opt. Tech.* **2016**, 506–510. [CrossRef]
35. Zhang, G.Y.; Liu, J.H.; Mi, J. Research on Watermarking Algorithm for 3D Color Point Cloud Model. *Comput. Technol. Dev.* **2023**. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.