Hindawi
Journal of Sensors
Volume 2020, Article ID 8851999, 14 pages
https://doi.org/10.1155/2020/8851999

Research Article

Reversible Data Hiding for Encrypted 3D Model Based on Prediction Error Expansion

Li Li,[1] Shengxian Wang,[1,2] Ting Luo,[3] Ching-Chun Chang,[4] Qili Zhou,[1] and Hui Li[2]

[1]*Department of Computer Science, Hangzhou Dianzi University, Hangzhou 310018, China*
[2]*Key Laboratory of Brain Machine Collaborative Intelligence of Zhejiang Province, Hangzhou 310018, China*
[3]*Collage of Science and Technology, Ningbo University, Ningbo 315000, China*
[4]*Department of Electronic Engineering, Tsinghua University, Beijing 100084, China*

Correspondence should be addressed to Ting Luo; luoting@nbu.edu.cn

Received 11 June 2020; Revised 9 July 2020; Accepted 24 July 2020; Published 1 September 2020

Academic Editor: Fei Yu

Since 3D models can intuitively display real-world information, there are potential scenarios in many application fields, such as architectural models and medical organ models. However, a 3D model shared through the internet can be easily obtained by an unauthorized user. In order to solve the security problem of 3D model in the cloud, a reversible data hiding method for encrypted 3D model based on prediction error expansion is proposed. In this method, the original 3D model is preprocessed, and the vertex of 3D model is encrypted by using the Paillier cryptosystem. In the cloud, in order to improve accuracy of data extraction, the dyeing method is designed to classify all vertices into the embedded set and the referenced set. After that, secret data is embedded by expanding direction of prediction error with direction vector. The prediction error of the vertex in the embedded set is computed by using the referenced set, and the direction vector is obtained according to the mapping table, which is designed to map several bits to a direction vector. Secret data can be extracted by comparing the angle between the direction of prediction error and direction vector, and the original model can be restored using the referenced set. Experiment results show that compared with the existing data hiding method for encrypted 3D model, the proposed method has higher data hiding capacity, and the accuracy of data extraction have improved. Moreover, the directly decrypted model has less distortion.

1. Introduction

With the rapid advancement of multimedia processing technologies on the internet, data hiding methods have been developed for the security of three-dimensional (3D) models [1–3]. Data hiding methods achieve integrity authentication and copyright protection by embedded secret data into the content of the original carrier [4, 5]. However, for the occasions with high data security requirement, such as medical images and judicial certification, no modification is allowed on the original carrier. Therefore, reversible data hiding (RDH) has attracted more researchers for potential applications [6–8].

Traditional RDH methods can be divided into three categories: difference expansion (DE), histogram shifting (HS), and lossless compression (LC). DE-based RDH methods embed secret data into carrier image by expanding the difference among adjacent pixels [9, 10]. Prediction error expansion (PEE), which belongs to difference expansion, embeds secret data by expanding the difference between the actual value and the prediction value of pixels [11–14]. HS-based RDH methods generate the feature histogram of original image and embed secret data into the smallest point of the histogram [15, 16]. LS-based RDH methods compress the specified area of the carrier image and embed secret data into the compressed area [17].

With the development of outsourced storage in the cloud, reversible data hiding in encrypted domain (RDH-ED) has been studied for security of multimedia files in the cloud.

The existing RDH-ED methods are mainly classified into reserving room before encryption (RRBE) and vacating room after encryption (VRAE). The RRBE method reserves embedding room before encrypting the original image. For example, Ma et al. proposed a reversible data hiding method, in which the room is reversed by self-embedding before encryption. Zhang [19] constructed the histogram of prediction error and reserved room by HS, which is the most popular method for reversible data hiding. Cao et al. improved the method of [18, 19] by generating prediction error with a small entropy so that the reserved room can be increased, and the data hiding capacity can be improved [20].

The VRAE method directly implements data embedding by modifying the encrypted image [21, 22]. For example, Zhang embedded secret data by flipping the three least significant bits of a pixel [19]. With the help of the spatial correlation of natural images, the receiver extracted the secret data through the evaluation function of texture complexity. Based on the method of [19], Hong et al. increased data hiding capacity by improving the evaluation function [23]. Zhang [19] emptied out space for data embedding by using the typical manner of cipher-text compression.

However, the above RDH methods are for images and cannot be directly used to 3D models because the data structure of the 3D model is different from that of the image. Therefore, Wu and Cheung proposed a RDH method for 3D models based on DE, which embeds secret data by modifying the difference among adjacent vertices [24]. Jhou et al. constructed the histogram of the distance between all vertices and the center of 3D model and embedded secret data by histogram shifting. However, these methods cannot be implemented in the encrypted domain [25]. Jiang et al. proposed a RDH-ED method for 3D models based on stream cipher encryption [26]. By flipping several least significant bits of vertex coordinates, one bit was embedded. The receiver extracted secret data by using the spatial correlation of the original 3D model. Shah and Zhang proposed a watermarking method based on the Paillier cryptosystem, which used VRAE framework to vacate space before encryption [27]. Wang et al. embedded secret data in the encrypted domain by constructing direction histogram and histogram shifting [28].

Combining homomorphic encryption and prediction error expansion, a RDH method for encrypted 3D model is proposed in this paper. In this method, the original 3D model is preprocessed, and the vertex of 3D model is encrypted by using Paillier cryptosystem. In the cloud, in order to improve the accuracy of data extraction, the dyeing method is designed to classify all vertices into the embedded set and the referenced set. After that, the secret data is embedded by expanding direction of prediction error with direction vector. The prediction error of the vertex is computed by using the referenced set, and the direction vector is obtained according to the mapping table. Moreover, the mapping table is constructed to map several bits to a direction vector. Secret data can be extracted by comparing the angle between the direction of prediction error and direction vector, and the original model can be restored using the referenced set. The contributions of the paper are organized as follows.

(1) By designing the dyeing method to classify all vertices into the embedded set and the referenced set, the error rate of data extraction can be reduced

(2) The mapping table is constructed to map several bits to a direction vector, so that several bits can be embedded into a vertex, which improves data hiding capacity

(3) Compared the existing RDH-ED methods, the proposed method has higher capacity, lower bit error rate, and the directly decrypted 3D model has less distortion

The rest of this paper is organized as follows. The Paillier cryptosystem is briefly introduced in Section 2. The related reversible data hiding method is proposed in Section 3. The experimental results are shown in Section 4. The conclusions are discussed in Section 5.

2. Related Algorithm

The Paillier cryptosystem [29], which has been widely used in encrypted signal processing, has homomorphism and probability. Homomorphism means that the product of two ciphertexts is consistent with the sum of two corresponding plaintexts. Probability means that different ciphertexts, which are obtained by encrypting the same plaintext with different parameters, can be decrypted to the same plaintext. The following describes the process of key generation, encryption, decryption, two characteristics, and subtraction homomorphism expansion.

2.1. Key Generation. Randomly pick up two large prime numbers p and q. Calculate $N = pq$ and $\lambda = \text{lcm}(p-1, q-1)$, where lcm $(\cdot)$ stands for the lowest common multiple. Afterwards, select $g \in Z^*_{N^2}$ randomly, which satisfies

$$\gcd\left(L\left(g^{\lambda} \bmod N^2\right), N\right) = 1, \tag{1}$$

where $L(u) = (u-1)/N$, and gcd $(\cdot)$ means the greatest common divisor of two inputs. $Z_{N^2} = \{0, 1, 2, \cdots, N^2 - 1\}$ and $Z^*_{N^2}$ are the numbers in Z_{N^2} which prime with N^2. Finally, we get the public key (N, g) and corresponding private key λ.

2.2. Encryption. Select a parameter $r \in Z^*_{N^2}$ randomly. The plaintext $m \in Z_N$ can be encrypted to the corresponding ciphertext c by

$$c = E[m, r] = g^m \cdot r^N \bmod N^2, \tag{2}$$

where $E[\cdot]$ denotes the encryption function.

2.3. Decryption. The original plaintext m can be obtained by

$$m = D[c] = \frac{L\left(c^{\lambda} \bmod N^2\right)}{L\left(g^{\lambda} \bmod N^2\right)} \bmod N. \tag{3}$$

Moreover, two important characteristics are described as follows (which has been applied in the proposed method).

2.4. Lemma One. For two plaintexts $m_1, m_2 \in Z_N$, compute corresponding ciphertexts c_1, c_2 with r_1, r_2 according to Equation (1), respectively. The equation $c_1 = c_2$ holds if and only if $m_1 = m_2$ and $r_1 = r_2$.

2.5. Homomorphic Multiplication. For $\forall r_1, r_2 \in Z_N^*$, two plaintexts $m_1, m_2 \in Z_N$ and corresponding ciphertexts $E[m_1, r_1], E[m_2, r_2] \in Z_{N^2}^*$ satisfy

$$c_1 \cdot c_2 = E[m_1, r_1] \cdot E[m_2, r_2] = g^{m_1+m_2} \cdot (r_1 \cdot r_2)^N \bmod N^2,$$
$$D[c_1 \cdot c_2] = D\left[E[m_1, r_1] \cdot E[m_2, r_2] \bmod N^2\right] = m_1 + m_2 \bmod N. \quad (4)$$

The original Paillier encryption system only has addition homomorphism and multiplication homomorphism. The subtraction homomorphism of the Paillier encryption system can be realized as follows.

2.6. Subtraction Homomorphic Expansion. In order to calculate the subtraction $m_1 - m_2$ of two numbers m_1, m_2 in the encrypted domain, the negative number $-m_2$ should be expressed by a positive number $N - m_2$. Suppose that $E[m_2]^{-1}$ denotes the ciphertext of $N - m_2$, the ciphertext $E[m_2]^{-1}$ can be calculated by Euclidean algorithm [30] and Modular Multiplication Inverse. Hence, the corresponding result of $m_1 - m_2$ in the encrypted domain can be obtained with $E[m_1] \cdot E[m_2]^{-1} \bmod N^2$.

3. The Proposed Reversible Data Hiding Method

In order to protect 3D model in the cloud, a reversible data hiding (RDH) method for encrypted 3D model is proposed. Figure 1 shows the flowchart of the proposed method. An original 3D model is preprocessed, and the vertex of 3D model is encrypted by using the Paillier cryptosystem. In the cloud, all vertices are classified into the embedded set and the referenced set. Then, the secret data is embedded by expanding direction of prediction error with direction vector. The prediction error of the vertex is computed by using the referenced set, and the direction vector is obtained according to the mapping table. Receiver can extract secret data by comparing the angle between prediction error and the direction vector, and the original model can be restored using the referenced set.

3.1. Preprocessing. Because the input of the Paillier cryptosystem should be a positive integer, the vertex coordinates firstly are converted from decimal to positive integer.

3D models are consisted of vertex data and connectivity data. The vertex data includes the coordinates of each vertex in the spatial domain. The connectivity data reflects the connection relationship between vertices. A 3D model Fairy and its local region are illustrated in Figure 2, and the corresponding format file is shown in Table 1. Each vertex and each face of the 3D model have a corresponding index, respectively. For a 3D model M, let $\{v_i\}_{i=0}^{N_V}$ represent the sequence of vertices, where $v_i = (v_{i,x}, v_{i,y}, v_{i,z})$, and N_V is the number of vertices. Note that each coordinate $|v_{i,j}| < 1, j \in \{x, y, z\}$, and the significant digit of each coordinate is 6.

Normally, uncompressed vertices are 32-bit floating point numbers with a precision of 6 digits. Therefore, the vertex coordinates are converted into an integer with k significant digits by using.

$$v'_{i,j} = \left\lfloor v_{i,j} \cdot 10^k \right\rfloor, \quad j \in \{x, y, z\}. \quad (5)$$

Moreover, all vertex coordinates should be converted to positive integers for encryption by using.

$$v'_{i,j} = v'_{i,j} + 10^k, \quad j \in \{x, y, z\}. \quad (6)$$

If k is greater than 6, the model can be restored losslessly; however, the time cost of encryption and decryption is large. If k is less than 6, the time cost is small, while the model cannot be restored losslessly. In order to balance the time cost and distortion of 3D model, the best k will be selected through the experiment.

3.2. Encryption. Referring to Equation (2), an integer r can be randomly selected to encrypt the vertex coordinates $v'_i = (v'_{i,x}, v'_{i,y}, v'_{i,z})$ with the public key (N, g).

$$c_{i,j} = E\left[v'_{i,j}, r_{i,j}\right] = g^{v'_{i,j}} r_{i,j}{}^N \bmod N^2, \; j \in \{x, y, z\}, \quad (7)$$

where $c_{i,j}$ is the ciphertext from the plaintext $v'_{i,j}$.

3.3. Data Embedding. Firstly, the dyeing method is designed to classify all vertices into the embedded set and the referenced set. Secondly, the prediction error of the vertex in the embedded set is computed using the referenced set. Thirdly, a one-to-one mapping is constructed to map several bits to a direction vector. Finally, for embedding secret data, the direction vector and the embedded key are used to expand the direction of the prediction error. In addition, the embedded key is calculated to reduce the accuracy of data extraction.

3.3.1. Classifying the Vertices. For the vertex of 3D model, 1-ring neighborhood of the vertex refers to these vertices that are directly adjacent to the vertex. As shown in Figure 3, the vertex v'_j is adjacent of the vertex v'_i. If a vertex is modified to embed secret data, its 1-ring neighborhood cannot be modified, and it is mainly because the 1-ring neighborhood is required to calculate the prediction value of the vertex. Therefore, the dyeing algorithm is designed to classify all vertices into the embedded set and the referenced set because the color of adjacent vertices cannot be same. The vertices in the referenced set can be used to calculate the prediction value of the vertices in the embedded set because the referenced set consists of 1-ring neighborhood of all vertices in the embedded set, while the vertices in embedded set are modified to embed secret data.

Suppose that S_e denotes the embedded set, and S_r denotes the referenced set. In order to obtain large data hiding

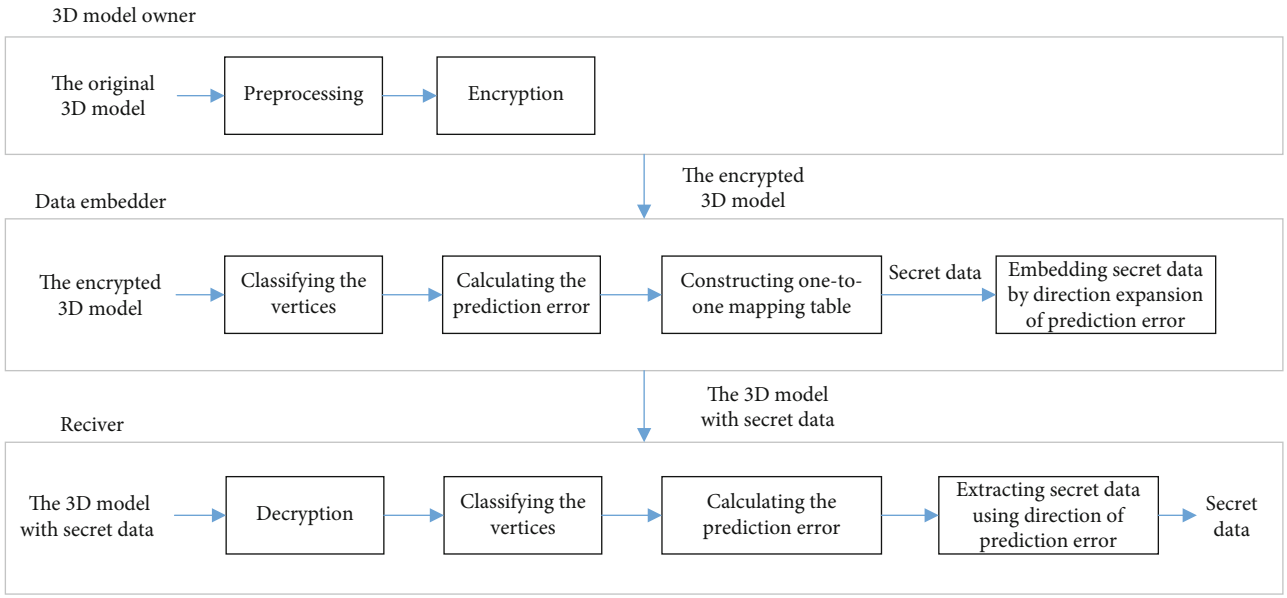

Figure 1: Flowchart of the proposed RDH-ED method.

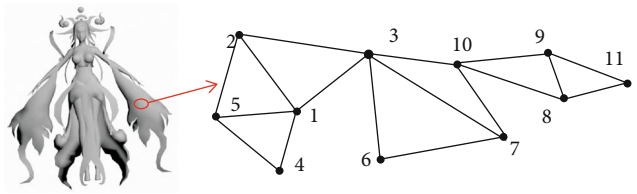

Figure 2: Illustration of a 3D model Fairy and its local region.

Table 1: File format of Figure 2.

Index of vertex	Vertex list			Face information	
	X -axis	Y -axis	Z -axis	Index of face	Elements in each face
1	$v_{1,x}$	$v_{1,y}$	$v_{1,z}$	1	(2,3,1)
2	$v_{2,x}$	$v_{2,y}$	$v_{2,z}$	2	(5,1,4)
3	$v_{3,x}$	$v_{3,y}$	$v_{3,z}$	3	(3,6,7)
4	$v_{4,x}$	$v_{4,y}$	$v_{4,x}$	4	(5,2,1)
5	$v_{5,x}$	$v_{5,y}$	$v_{5,z}$	5	(3,10,7)

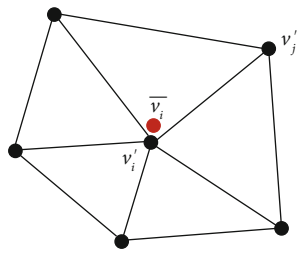

Figure 3: The actual value and the prediction value of the vertex.

capacity, the number of the vertices in the embedded set should be increased. The embedded set is the nonadjacent vertex set. According to graph theory, the existing polynomial algorithms can find the largest nonadjacent vertex set if the graph is a bipartite graph. However, the connectivity data of a 3D model cannot be represented by a bipartite graph since all loops of 3D model are 3. Hence, finding the largest embedded set in a 3D model is NP-hard problem. In order to increase data hiding capacity, the dyeing algorithm is designed to increase the number of the vertices in the embedded set.

Referring to the four-color theorem, different colors are required for any two adjacent vertices. For 3D models, at least seven colors are needed to dye the vertices to ensure that the colors of the adjacent vertices are different. After completing the process of dyeing, the color with most vertices is selected, and all vertices of this color are regarded as the embedded set, and the remaining vertices are regarded as the referenced set. Suppose that C denotes the color set of dyeing all vertices, Z_i denotes the color set of 1-ring neighborhood of the vertex, and c_i denotes the color of the ith vertex. The steps for the vertex classification using the dyeing algorithm are listed as follows.

Step 1. Suppose that the color of all vertices is black, and traverse all vertices in order of the vertex index.

Step 2. For an unclassified vertex v_i, statistic the color set of its 1-ring neighborhood.

Step 3. Dye the vertex v_i using the first color in C but not in Z_i.

Step 4. Determine whether the next vertex v_{i+1} is black. If v_{i+1} is black, the classification ends. If v_{i+1} is not black, loop from Step 2 until no vertex is black.

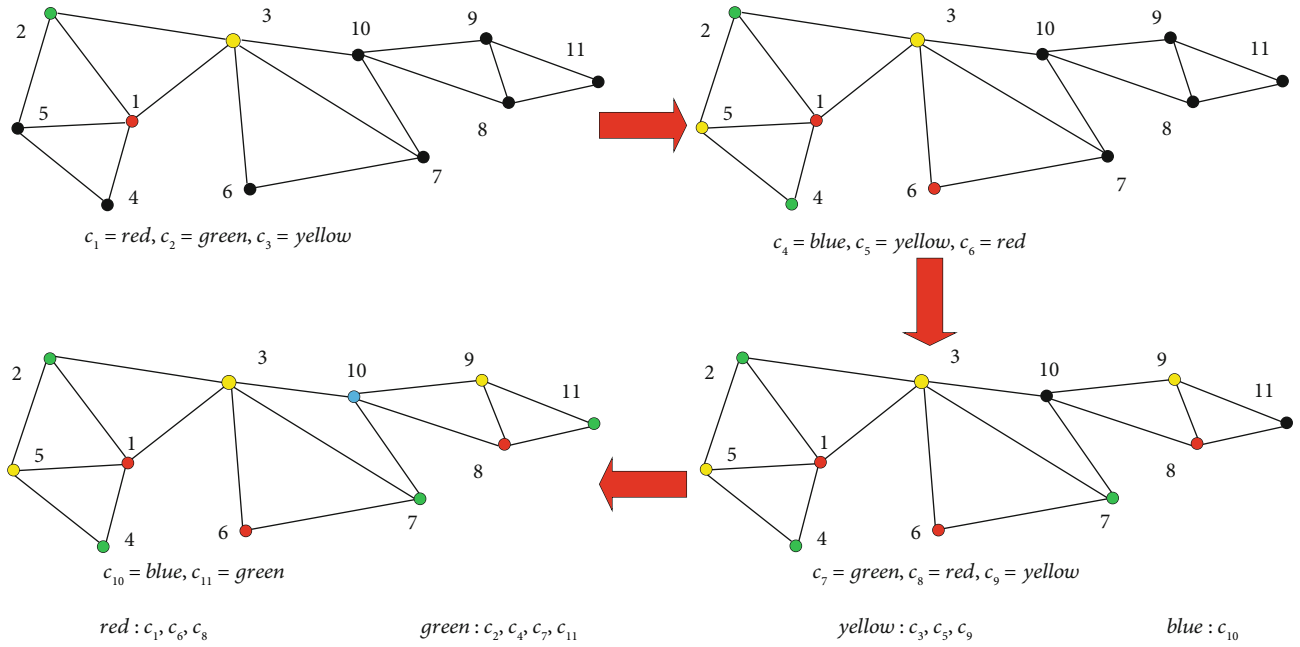

Figure 4: The process of classifying the vertices using the dyeing method.

Table 2: The one-to-one mapping table between weighted sum and direction vector.

Embed one bit into a vertex, which means $n = 1$, and $s_w \in [0, 1]$. On this condition, let $b_{k,j} \in \{-1, 1\}$.								
$s_w \in [0, 1]$	0	(-1,-1,-1)	1	(-1,-1,1)				
Embed two bits into a vertex, which means $n = 2$, and $s_w \in [0, 3]$. On this condition, let $b_{k,j} \in \{-1, 1\}$.								
$s_w \in [0, 3]$	0	(-1,-1,-1)	1	(-1,-1,1)	2	(-1,1,-1)	3	(-1,1,1)
Embed three bits into a vertex, which means $n = 3$, and $s_w \in [0, 7]$. On this condition, let $b_{k,j} \in \{-1, 1\}$.								
$s_w \in [0, 7]$	0	(-1,-1,-1)	1	(-1,-1,1)	2	(-1,1,-1)	3	(-1,1,1)
	4	(1,-1,-1)	5	(1,-1,1)	6	(1,1,-1)	7	(1,1,1)
Embed four bits into a vertex, which means $n = 3$, and $s_w \in [0, 15]$. On this condition, let$b_{k,j} \in \{-1, 0, 1\}$.								
$s_w \in [0, 15]$	0	(-1,-1,-1)	1	(-1,-1,0)	2	(-1,-1,1)	3	(-1,0,-1)
	4	(-1,0,0)	5	(-1,0,1)	6	(-1,1,-1)	7	(-1,1,0)
	8	(-1,1,1)	9	(0,-1,-1)	10	(0,-1,0)	11	(0,-1,1)
	12	(0,0,-1)	13	(0,0,1)	14	(0,1,-1)	15	(0,1,0)

Step 5. Select the most frequently used color, add all vertices of this color to the embedded set, and the remaining vertices are added to the referenced set.

Since all vertices are traversed only once, the time complexity of the dyeing algorithm is $O(n)$. For example, as illustrated in Figure 4, let $C = \{\text{red, green, yellow, blue}\}$. Traverse the vertex v_1, and let $c_1 = \text{red}$ since $Z_1 = \{\text{black}\}$. Traverse the vertex v_2, and let $c_2 = \text{green}$ since $Z_1 = \{\text{black, red}\}$. Traverse the vertex v_2, and let $c_2 = \text{yellow}$ since $Z_1 = \{\text{black, red, green}\}$. After traversing all vertices, the most frequently used color green is selected. The vertex set {2,6,7,11} of color green is regarded as the embedded set, while the remaining vertices as the referenced set.

3.3.2. Computing of Prediction Error. The accuracy of prediction error directly affects the performance of data hiding. As shown in Figure 3, the vertex v'_j is adjacent to the vertex v'_i, and the vertex $\bar{v}_i$ is the prediction value of vertex v'_i. According to the correlation of adjacent vertices, prediction value $\bar{v}_i$ can be calculated by using

$$\bar{v}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} v'_j \quad j \in (1, N_i), \tag{8}$$

where N_i denotes the number of the adjacent vertices of the vertex v'_i.

The prediction error of the vertex v_i' can be calculated by using

$$\Delta v_i = v_i' - \bar{v}_i, \tag{9}$$

where Δv_i denotes the prediction error of the vertex v_i'. Δv_i is a three-dimensional vector with random direction. Due to the spatial correlation, the modulus length $|\Delta v_i|$ is usually small, and experiment results show that the modulus length $|\Delta v_i|$ has the maximum D. Hence, the range of $|\Delta v_i|$ is as follows.

$$|\Delta v_i| \in [0, D]. \tag{10}$$

3.3.3. Constructing the Mapping Table. In order to embed several bits into a vertex in the embedded set, several bits should be mapped to a direction vector.

Data embedder converts secret data into several groups with n bits, and n is a shared parameter. Let a group with n bits denote as $w(w_0, w_1, \cdots, w_{n-1})$. Suppose that s_w denotes weighted sum of w, and s_w is calculated as

$$s_w = \sum_{i=0}^{n-1} w_i \cdot 2^i, \quad s_w \in [0, 2^n - 1]. \tag{11}$$

In order to embed s_w into a vertex, the corresponding direction vector of $s_w$should be constructed. Suppose that $\vec{b}_{s_w}$ denotes the direction vector of s_w, and s_w is embedded by expanding the direction of prediction error with direction vector $\vec{b}_{s_w}$. The one-to-one mapping between weighted sum and direction vector is constructed as shown in Table 2. Let $b_{k,j} \in \{-1, 1\}, k \in [0, 7]$, and if $n \le 3$, eight direction vectors can be constructed. The mapping between direction vector and weighted sum is shown at the top three rows in Table 2. Let $b_{k,j} \in \{-1, 0, 1\}, k \in [0, 26)$, and if $n = 4$, twenty-six direction vectors are constructed as shown at the fifth row in Table 2. Let $b_{k,j} \in \{-2, -1, 0, 1, 2\}, k \in [0, 110)$, and if $n = 5$ or $n = 6$, 110 direction vectors are constructed. Let W_{b_k} denote the direction weight of direction vector. As shown in Table 2, the direction vector $\vec{b}_k$ is sorted according to the value W_{b_k}, and the direction weight W_{b_k} of the direction vector is calculated as follows.

$$W_{b_k} = 9b_{k,1} + 3b_{k,2} + b_{k,3}. \tag{12}$$

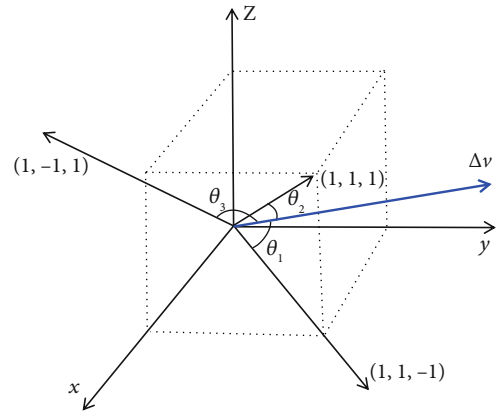

Figure 5: The angle between the prediction error and three direction vectors.

Hence, if the shared parameter n is obtained, the mapping table between weighted sum and direction vector can be constructed. For different s_w, the corresponding direction vector $\vec{b}_{s_w}$ can be found through the mapping table.

For example, if $n = 2$, four direction vectors $\vec{b}_{s_w}$ should be constructed to represent four weighted sum s_w, respectively. Let $b_{k,j} \in \{-1, 1\}$, eight direction vectors can be constructed, and the first four direction vectors are selected according to the direction weight W_{b_k}. As a result, the four direction vectors (-1,-1,-1), (-1,-1,1), (-1,1,-1), and (-1,1,1) are constructed to represent $s_w \in [0, 3]$, respectively.

3.3.4. Data Embedding. In order to embed secret data into the vertex, the vertex coordinates are required to be modified by using the direction vector and the embedding key. The weighted sum s_w of $w(w_0, w_1, \cdots, w_{n-1})$ is embedded in the vertex v_i' by using Equation (13).

$$v_i'' = v_i' + \varphi \vec{b}_{s_w}, \tag{13}$$

where v_i'' is the modified vertex coordinate of v_i', parameter φ is the embedding key, and $\vec{b}_{s_w} = (b_{s_w,x}, b_{s_w,y}, b_{s_w,z})$ is the corresponding direction vector of s_w.

Referring to Equation (13), data embedding in the encrypted domain is performed as

$$\begin{cases} c_{i,j}' = c_{i,j} \cdot c_{s_w} = E\left[v_{i,j}, r_{i,j}\right] \cdot E\left[-\varphi \cdot b_{s_w,j}, r_{s_w}\right]^{-1} \bmod N^2, & \text{if } \varphi \cdot b_{s_w,j} < 0, j \in \{1, 2, 3\}, \\ c_{i,j}' = c_{i,j} \cdot c_{s_w} = E\left[v_{i,j}, r_{i,j}\right] \cdot E\left[\varphi \cdot b_{s_w,j}, r_{s_w}\right] \bmod N^2, & \text{if } \varphi \cdot b_{s_w,j} > 0, j \in \{1, 2, 3\}, \\ c_{i,j}' = c_{i,j}, & \text{if } \varphi \cdot b_{s_w,j} = 0, j \in \{1, 2, 3\}, \end{cases} \tag{14}$$

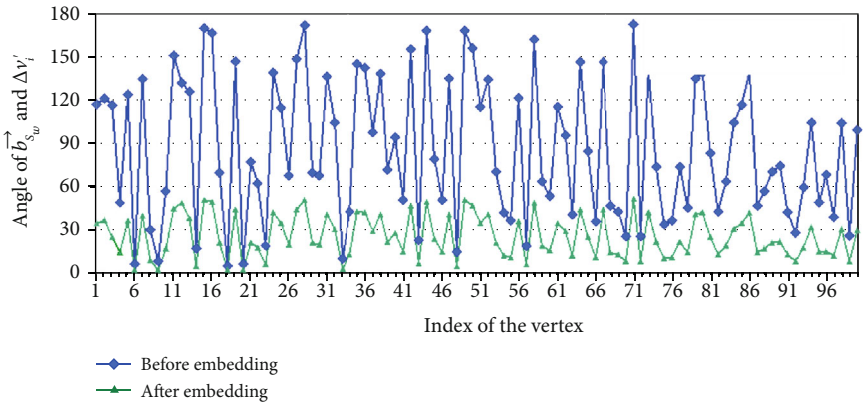

FIGURE 6: The changes of angle between prediction error $\Delta v_i'$ and direction vector $\vec{b}_{s_w}$ before and after data embedding.

where $c'_{i,j}$ is the ciphertext of $v''_{i,j}$, and c_{s_w} is the ciphertext of $\varphi \cdot b_{s_w,j}$. Moreover, if $\varphi \cdot b_{s_w,j} > 0$, addition homomorphism of Paillier cryptosystem is utilized to embed secret data. If $\varphi \cdot b_{s_w,j} < 0$, subtraction homomorphism expansion of Paillier cryptosystem is utilized to embed secret data.

After data embedding, the corresponding change of the prediction error in the plaintext is as follows.

$$\Delta v_i' = \Delta v_i + \varphi \cdot \vec{b}_{s_w}. \tag{15}$$

After Δv_i being changed, the angle between the direction of prediction error and direction vector will be small.

In order to improve accuracy of data extraction, two inferences about vector are provided to calculate the embedded key.

Inference 1. Suppose that $\vec{m}_1$ and $\vec{m}_2$ are two three-dimensional vectors. If the directions of $\vec{m}_1$ and $\vec{m}_2$ are the same, the modulus length $|\vec{m}_1 + \vec{m}_2|$ has the maximum. If the directions of $\vec{m}_1$ and $\vec{m}_2$ are the opposite, the modulus length $|\vec{m}_1 + \vec{m}_2|$ has the minimum. The proof is listed as follows.

$$\left|\vec{m}_1 + \vec{m}_2\right|^2 = \left|\vec{m}_1\right|^2 + \left|\vec{m}_2\right|^2 + 2\left|\vec{m}_1\right| \cdot \left|\vec{m}_1\right| \cdot \cos\theta\left(\vec{m}_1, \vec{m}_2\right), \tag{16}$$

where $\theta(\vec{m}_1, \vec{m}_2)$ is the angle between $\vec{m}_1$ and $\vec{m}_2$. According to Equation (16), since $|\vec{m}_1 + \vec{m}_2|$ is related to the angle, the above inference holds.

Inference 2. For two unit vectors $\vec{n}_1$ and $\vec{n}_2$, let $\theta(\vec{n}_1, \vec{n}_2)$ denote the angle between two vectors, and let $\theta(\vec{n}_1, \vec{n}_1 - \vec{n}_2)$ denote the angle between $\vec{n}_1$ and $\vec{n}_1 - \vec{n}_2$. $\theta(\vec{n}_1, \vec{n}_1 - \vec{n}_2)$ and $\theta(\vec{n}_1, \vec{n}_2)$ are positively related. The smaller $\theta(\vec{n}_1, \vec{n}_2)$, the smaller $\theta(\vec{n}_1, \vec{n}_1 - \vec{n}_2)$. In addition, $\theta(\vec{n}_1, \vec{n}_1 - \vec{n}_2) \in [0, \pi/2]$. The proof is listed as follows.

$$\begin{aligned}\cos\theta\left(\vec{n}_1, \vec{n}_1 - \vec{n}_2\right) &= \frac{\vec{n}_1 \times \left(\vec{n}_1 - \vec{n}_2\right)}{\left|\vec{n}_1\right| \cdot \left|\vec{n}_1 - \vec{n}_2\right|} \\ &= \frac{\left|\vec{n}_1\right|^2 - \left|\vec{n}_1\right| \cdot \left|\vec{n}_2\right| \cos\theta\left(\vec{n}_1, \vec{n}_2\right)}{\left|\vec{n}_1\right| \cdot \left|\vec{n}_1 - \vec{n}_2\right|},\end{aligned} \tag{17}$$

where $\times$ denotes cross product. Since $\vec{n}_1$ and $\vec{n}_2$ are unit vector, Equation (18) can be obtained according to Equation (17).

$$\cos\theta\left(\vec{n}_1, \vec{n}_1 - \vec{n}_2\right) = \sqrt{\left(1 - \cos\theta\left(\vec{n}_1, \vec{n}_2\right)\right)/2}. \tag{18}$$

According to Equation (18), $\cos\theta(\vec{n}_1, \vec{n}_1 - \vec{n}_2) > 0$, and $\theta(\vec{n}_1, \vec{n}_1 - \vec{n}_2) \in [0, \pi/2]$. The above inference holds.

3.3.5. Embedding Key Calculation. The embedding key influences the accuracy of data extraction. If the embedding key satisfies a certain condition, secret data can be extracted correctly.

Figure 5 shows the angle between prediction error and three direction vectors. Suppose that θ_{s_w} denotes the angle between prediction error Δv_i and direction vector $\vec{b}_{s_w}$ corresponding to the secret data s_w, and θ_k denotes the angle between prediction error Δv_i and other direction vector $\vec{b}_k (k \in [0, 2^n - 1], k \neq s_w)$. θ_{s_w} is computed as

$$\cos\theta_{s_w} = \frac{\Delta v_i' \times \vec{b}_{s_w}}{\left|\Delta v_i'\right| \left|\vec{b}_{s_w}\right|}. \tag{19}$$

Since the secret data is extracted by using the smallest angle between the prediction error and the direction vector, in order to improve the accuracy of data extraction, θ_{s_w} should be smaller than $\theta_k (k \in [0, 2^n - 1], k \neq s_w)$. After data embedding, Equation (20) should be satisfied.

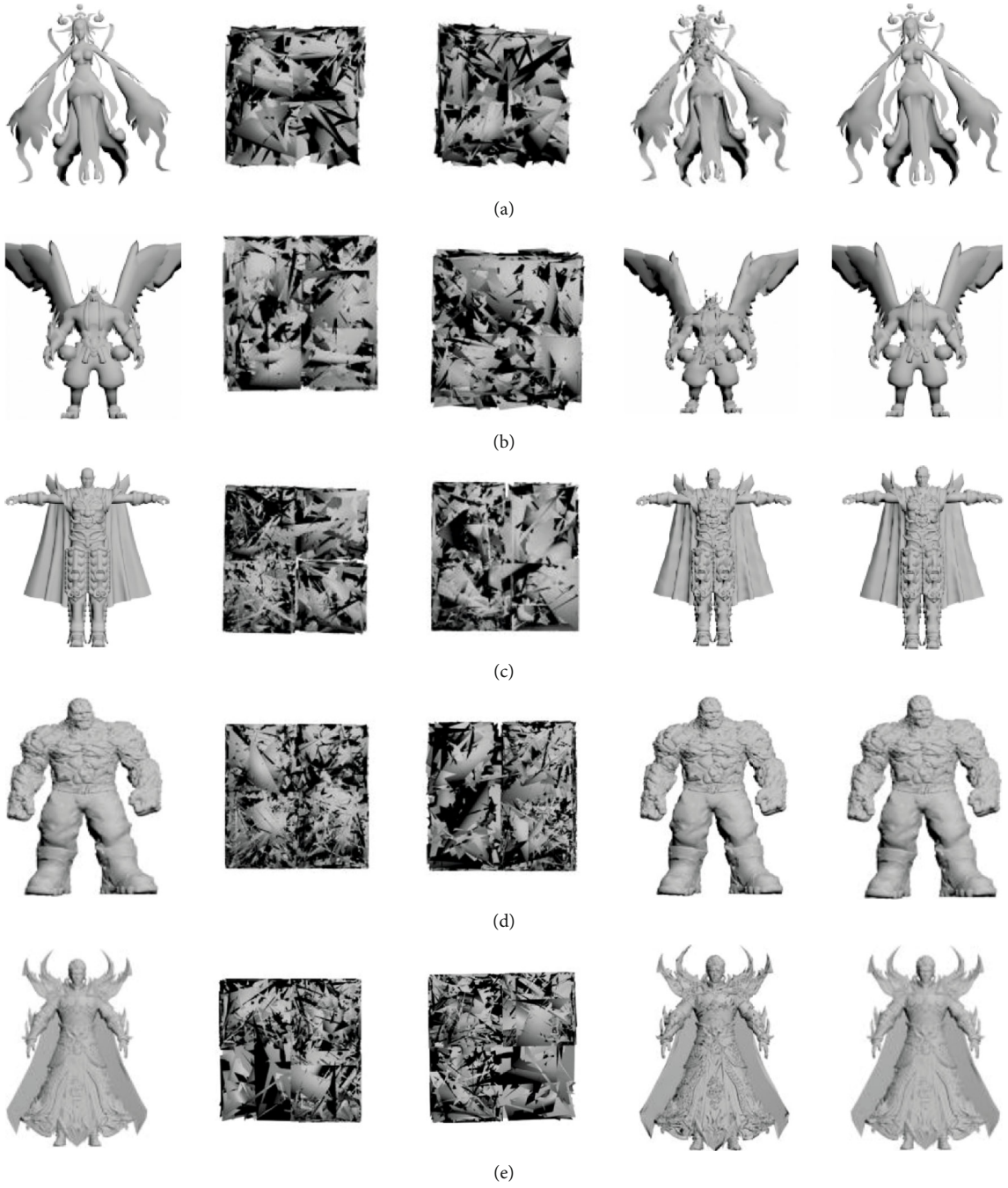

Figure 7: Illustrative examples showing the appearance of five models in different phases, include original 3D model, encrypted 3D model, data-embedded 3D model, directly decrypted 3D model, and recovery 3D model. (a) Fairy, (b) boss, (c) Devil, (d) Thing, and (e) Lord.

$$\forall k \in [0, 2^n - 1], k \neq s_w, \quad \cos\theta_{s_w} > \cos\theta_k. \tag{20}$$

The following equation can be derived by using Cosine Theorem.

$$\forall k \in [0, 2^n - 1], k \neq s_w, \quad \frac{\Delta v_i' \times \vec{b}_{s_w}}{|\Delta v_i'|\left|\vec{b}_{s_w}\right|} > \frac{\Delta v_i' \times \vec{b}_k}{|\Delta v_i'|\left|\vec{b}_k\right|}. \tag{21}$$

It can be derived to the following equation.

$$\left(\Delta v_i + \varphi \vec{b}_{s_w}\right) \times \left(\frac{\vec{b}_{s_w}}{\left|\vec{b}_{s_w}\right|} - \frac{\vec{b}_k}{\left|\vec{b}_k\right|}\right) > 0. \tag{22}$$

Suppose that the vector $\vec{M} = (\vec{b}_{s_w}/|\vec{b}_{s_w}| - \vec{b}_k/|\vec{b}_k|)$, and Equation (22) can be simplified as

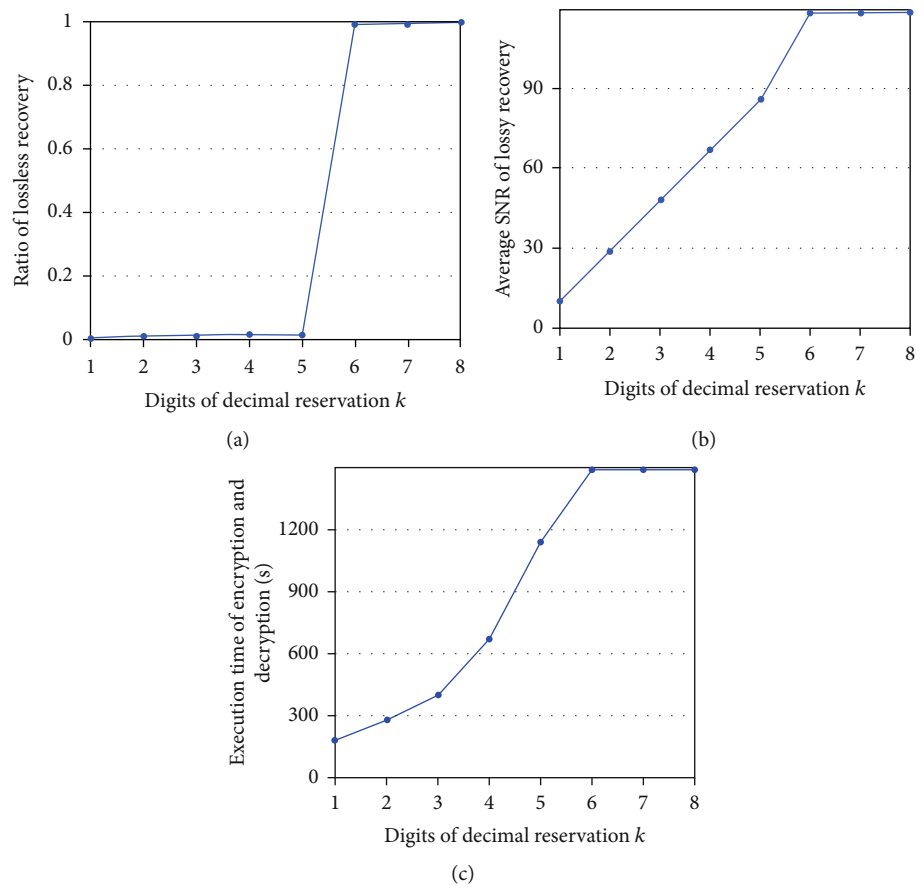

FIGURE 8: (a) Ratio of lossless recovery under different k. (b) Average SNR of decrypted model under different k. (c) Average execution time of encryption and decryption under different k.

$$\left(\Delta v_i + \varphi \vec{b}_{s_w}\right) \times \vec{M} > 0. \tag{23}$$

According to Inference 2, the angle between $\vec{b}_{s_w}$ and $\vec{M}$ is smaller than $\pi/2$, so $\vec{b}_{s_w} \times \vec{M} > 0$. Equation (24) can be derived.

$$\varphi > \frac{\Delta v \times \left(-\vec{M}\right)}{\vec{b}_{s_w} \times \vec{M}}. \tag{24}$$

Since the modulus length $|\Delta v| \in [0, D)$, Equation (25) can be derived according to Inference 1.

$$\Delta v = \frac{D}{\left|-\vec{M}\right|} \cdot \left(-\vec{M}\right), \quad \varphi > \left(\frac{\Delta v \times \left(-\vec{M}\right)}{\vec{b}_{s_w} \times \vec{M}}\right)_{\max} = \frac{D \cdot \left|\vec{M}\right|}{\vec{b}_{s_w} \times \vec{M}}. \tag{25}$$

It can be derived to the following equation by using Cosine Theorem.

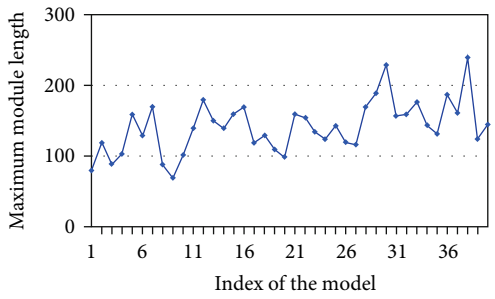

FIGURE 9: The maximum modulus length of forty 3D models.

$$\varphi > \frac{D \cdot \left|\vec{M}\right|}{\left|\vec{b}_{s_w}\right| \cdot \left|\vec{M}\right| \cdot \cos\theta\left(\vec{b}_{s_w}, \vec{M}\right)} \Leftrightarrow \varphi > \frac{D}{\left|\vec{b}_{s_w}\right| \cdot \cos\theta\left(\vec{b}_{s_w}, \vec{M}\right)}. \tag{26}$$

Suppose that $\theta(\vec{b}_{s_w}, \vec{M})$ denotes the angle between $\vec{M}$ and $\vec{b}_{s_w}$, in order to ensure that θ_{s_w} is smaller than θ_k, θ_{s_w} should satisfy the following equation.

$$\varphi = \left\lceil \frac{D}{\left|\vec{b}_{s_w}\right| \cdot \cos\theta\left(\vec{b}_{s_w}, \vec{M}\right)} \right\rceil. \quad (27)$$

If φ satisfies Equation (27), secret data can be extracted correctly. According to Inference 2 and Equation (27), φ has the maximum if the angle between $\vec{b}_{s_w}$ and $\vec{b}_k$ is smallest.

Since direction vector is related to the shared parameters n, the shared parameters influence the value of φ. According to Equation (27), the relationship between φ and n is obtained as follows.

$$\begin{cases} \varphi = D, & \text{if } n = 1, 2, 3, \\ \varphi = \left\lceil \sqrt{3}D \right\rceil, & \text{if } n = 4, \end{cases} \quad (28)$$

For example, for a clear description, let $n = 3$. In this condition, the angle between $\vec{b}_0 = (-1,-1,-1)$ and direction vector $\vec{b}_1 = (-1,-1,1)$ is the smallest. The embedding key can be calculated by the following equation.

$$\vec{M} = \left(\frac{\vec{b}_0}{\left|\vec{b}_0\right|} - \frac{\vec{b}_1}{\left|\vec{b}_1\right|}\right) = \frac{1}{\sqrt{3}}(-1,-1,2), \quad \cos\left(\vec{b}_0, \vec{M}\right) = \frac{1}{\sqrt{3}}, \varphi = \left\lceil \frac{D}{\left|\vec{b}_0\right| \cos\left(\vec{b}_0, \vec{M}\right)} \right\rceil = D. \quad (29)$$

In order to explain the whole processes, the data embedding example is given as follows. For convenience, suppose that $n = 2$, the ith vertex $v_i' = (1410, 2120, 790)$, the prediction value $\bar{v}_i = (1430, 2280, 750)$, and the secrete data $w' = (1, 0)$. Since $n = 2$, the weighted sum $s_w \in [0, 3]$ according to Equation (11), and four direction vectors can be obtained according to the mapping table. Four direction vectors $\vec{b}_0 = (-1,-1,-1)$, $\vec{b}_1 = (-1,-1,1)$, $\vec{b}_2 = (-1,1,-1)$, and $\vec{b}_3 = (-1,1,1)$ correspond to $s_w = 0$, $s_w = 1$, $s_w = 2$, and $s_w = 3$, respectively. Since $w' = (1, 0)$, then $s_{w'} = 2$ can be computed, and $s_{w'} = 2$ corresponds to $\vec{b}_2 = (-1,1,-1)$. In addition, $\varphi = 250$ can be obtained by Equation (28). The prediction error can be computed, and $\Delta v_i = (-20,-160,40)$ according to Equation (9). Initially, the angle between Δv_i and $\vec{b}_0$, $\vec{b}_1$, $\vec{b}_2$, $\vec{b}_3$ can be computed, and $\theta_0 = 60.88°$, $\theta_1 = 40.12°$, $\theta_2 = 128.7°$, and $\theta_3 = 110.3°$. After data hiding, θ_2 between Δv_i and $\vec{b}_2$ should be the smallest angle. During data hiding, $v_i'' = v_i' + \varphi\vec{b}_2$ and $\Delta v_i' = \Delta v_i + \varphi\vec{b}_2$. Hence, $v_i'' = (1160, 2370, 540)$ and $\Delta v_i' = (-270, 90, -210)$. Then, $\theta_0 = 50.49°$, $\theta_1 = 91.87°$, $\theta_2 = 21.58°$, and $\theta_3 = 75.83°$ can be computed. The result shows that θ_2 between $\Delta v_i'$ and $\vec{b}_2$ will be smallest after data hiding, and the secret data can be extracted by finding the smallest angle.

Table 3: The effect of the embedding key and shared parameter on bit error rate of data extraction.

n	φ					
	1	2	3	4	5	6
50	14.2%	17.3%	24.8%	33.5%	46.4%	46.7%
70	7.34%	9.12%	12.6%	23.4%	38.9%	38.9%
90	3.21%	4.25%	5.79%	16.5%	32.9%	32.9%
110	1.54%	2.25%	2.76%	10.8%	25.4%	25.5%
130	0.79%	1.22%	1.63%	6.86%	18.2%	18.3%
150	0.32%	0.63%	0.98%	4.25%	13.2%	13.3%
170	0.14%	0.26%	0.53%	2.84%	9.47%	9.47%
190	0.08%	0.11%	0.24%	1.77%	6.24%	6.25%
210	0.03%	0.05%	0.09%	1.25%	4.83%	4.83%
230	0.01%	0.02%	0.04%	0.84%	3.05%	3.06%
250	0	0	0	0.53%	2.24%	2.25%
270	0	0	0	0.32%	1.76%	1.76%
290	0	0	0	0.21%	1.43%	1.46%
210	0	0	0	0.15%	1.09%	1.09%
330	0	0	0	0.09%	0.79%	0.79%
350	0	0	0	0.04%	0.42%	0.42%

Figure 6 shows the changes of angles θ_{s_w} between prediction error $\Delta v_i'$ of 100 vertices and direction vector $\vec{b}_{s_w}$ corresponding to secret data s_w before and after data embedding when the shared parameter is 3. The result shows that the angle will become smaller after data embedding.

In addition, a large embedding key will make 3D model disturbed obviously. Hence, the embedding key will be discussed specifically in Section 4 to balance the distortion of the directly decrypted model and the accuracy of data extraction.

3.4. Data Extraction and Model Recovery. After receiving the encrypted model with secret data, receiver can decrypt 3D model with private key λ and obtain the directly decrypted 3D model. The decryption of 3D model is as follows.

$$v_{i,j}'' = D\left[c_{i,j}'\right] = \frac{L\left(\left(c_{i,j}'\right)^{\lambda} \bmod N^2\right)}{L\left(g^{\lambda} \bmod N^2\right)} \bmod N. \quad (30)$$

The directly decrypted 3D model is similar to the original model because only the coordinates of some vertices are modified slightly during data embedding.

After decrypting 3D model, all vertices are first classified into the embedded set and the referenced set using the dyeing algorithm. Secondly, prediction error of the vertex in embedded set is computed and the mapping table is constructed with the shared parameter n. Then, the angles between prediction error and all direction vector are computed, and the smallest angle θ_{s_w} is selected, which is the angle between

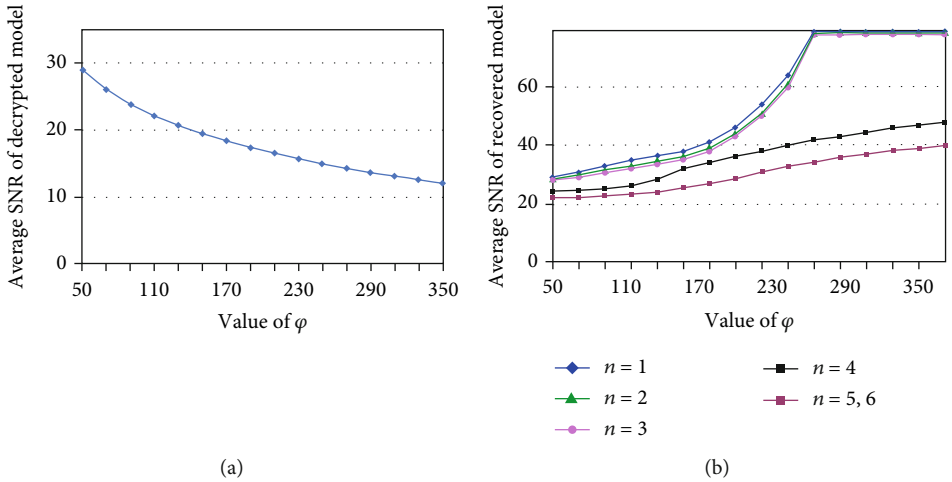

FIGURE 10: (a) The effect of the embedding key and shared parameter on SNR_D of decrypted model. (b) The effect of the embedding key and shared parameter on SNR_R of decrypted model.

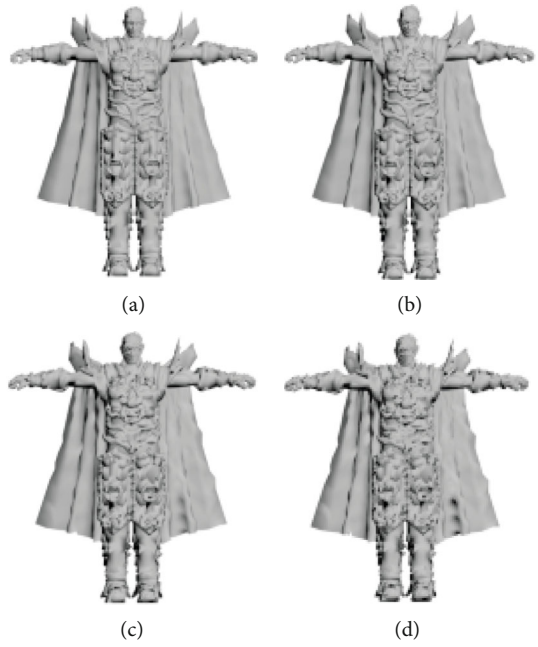

FIGURE 11: Four decrypted models devil when $n = 3$, and φ changes from 90 to 150. (a) $\varphi = 90$, (b) $\varphi = 110$, (c) $\varphi = 130$, and (d) $\varphi = 150$.

$\Delta v_i'$ and $\vec{b}_{s_w}$. At last, $\vec{b}_{s_w}$ can be obtained, and the corresponding s_w can be found by using the mapping table.

s_w is converted into n bit $w(w_0, w_1, \cdots w_{n-1})$ by using

$$w_i = \left\lfloor \frac{s_w'}{2^i} \right\rfloor \bmod 2, \quad i = 0, 1, 2, \cdots, n-1. \tag{31}$$

After data extraction, the embedding key and direction vector can be used to recovery the original 3D model by using

$$v_i' = v_i'' - \varphi \cdot \vec{b}_{s_w}. \tag{32}$$

4. Experiment Results and Discussion

The proposed method was implemented in Matlab 2016b under Window 7. We implemented the following experiment on 100 3D models and calculated the average of 100 3D models. Figure 7 shows a group of experiment results of five 3D models in different parses. The phases from left to right are original 3D model, encrypted 3D model, data-embedded 3D model, directly decrypted 3D model, and recovered 3D model. Directly decrypted 3D models have low distortion, and recovered 3D models have high similarity compared to original 3D model.

The similarity of disturbed 3D models is evaluated by the signal-to-noise ratio (SNR). The higher the SNR, the better the imperceptibility after embedding watermark. SNR is computed as

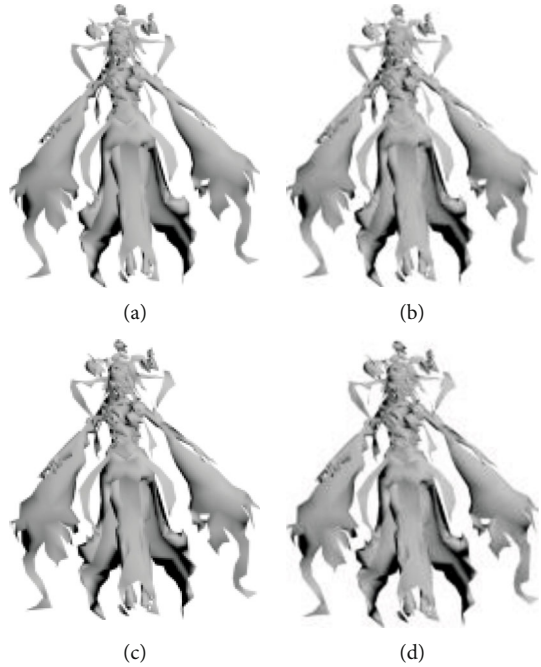

Figure 12: Four decrypted models fairy when n changes from 1 to 4. (a) $n = 1$, $\varphi = 110$; (b) $n = 2$, $\varphi = 110$; (c) $n = 3$, $\varphi = 110$; and (d) $n = 4$, $\varphi = 170$.

Table 4: The relationship between the number of the vertices and SNR_R.

Model	Vertices	D	Embedding rate	Error rate	SNR_D	SNR_R
Fairy	4252	205	83.7%	2.79%	18.43	34.79
Boss	10663	184	75.9%	2.54%	19.85	34.52
Devil	27872	169	77.8%	2.36%	21.32	35.16
Thing	110812	137	76.8%	2.88%	22.87	35.84
Lord	250343	118	77.4%	1.94%	24.16	35.32

$$\mathrm{SNR} = 10 \lg \frac{\sum_{i=1}^{N_V} \left[\left(v_{i,x} - \bar{v}_x\right)^2 + \left(v_{i,y} - \bar{v}_y\right)^2 + \left(v_{i,z} - \bar{v}_z\right)^2 \right]}{\sum_{i=1}^{N_V} \left[\left(g_{i,x} - v_{i,x}\right)^2 + \left(g_{i,y} - v_{i,y}\right)^2 + \left(g_{i,z} - v_{i,z}\right)^2 \right]}, \tag{33}$$

where $\bar{v}_x, \bar{v}_y, \bar{v}_z$ are the mean of vertex coordinates, $v_i(v_{i,x}, v_{i,y}, v_{i,z})$ are the original vertex coordinates, and $g_i(g_{i,x}, g_{i,y}, g_{i,z})$ are the coordinates of disturbed 3D model.

SNR_D is used to evaluate the similarity of directly decrypted models, and SNR_R is used to evaluate the similarity of recovered models. In addition, the bit error rate (BER) is used to measure the error rate of data extraction. The lower the BER, the higher the accuracy of data extraction.

4.1. Decimal Reservation Digits k. In order to observe the effect of decimal reservation digits k on the quality of the decrypted model and time cost, we changed the value of k from 1 to 8 and perform encrypting and decrypting on 3D models. As shown in Figure 8(a), $k = 6$ is a threshold, which enables 3D models recovery losslessly or limits recovery losslessly. Since the significant digits of vertex coordinates is 6, it is easy to cause permanent distortion if $k < 6$. The distortion of directly decrypted model (SNR_D) is calculated, which is shown in Figure 8(b). The time cost of encryption and decryption is related to the value k, which is shown in Figure 8(c). In order to obtain high quality of decrypted models and reduce the time cost, k is set to 4 in the next experiment.

4.2. The Maximum of the Modulus Length D. D is the maximum of the modulus length of the prediction error. Due to the spatial correlation of natural 3D model, the actual value and its prediction value of the vertex coordinates are relatively close. Hence, the modulus length of the prediction error always is small. In order to calculate the range of the prediction error, the experiment is performed on 40 3D models. Figure 9 shows the maximum modulus length of 40 3D models. For all 3D models, it can be observed that all maximum modulus lengths are less than 250. Hence, D is set to 250 in the next experiment.

4.3. The Choice of the Embedding Key and the Shared Parameter. According to Equation (13), the embedding key φ directly affects the distortion of decrypted model. If φ is large and satisfies Equation (27), secret data will be extracted correctly, but the quality of 3D models will be decreased obviously. If φ is small, the change of prediction error also will be small, and the accuracy of data extraction will reduce. However, there are several existing methods that can improve the accuracy of data extraction, such as ECC code and BCD code. Hence, in order to balance the accuracy of data extraction and the decrypted model, the experiment is carried out by select φ from 50 to 350 with the interval of 20.

Table 5: The performance of the proposed method compared with the existing method.

Methods	Embedding key	BER	SNR_D	SNR_R	Capacity (bpv)
The proposed method	$\varphi = 110$	2.76%	22.13	35.12	0.872
	$\varphi = 250$	0	15.24	∞	0.872
Method in [26]		4.22%	5.35	31.97	0.369
Method in [28]		0	30.08	∞	0.396

The larger the shared parameter, the greater the embedding capacity. However, the larger the shared parameter leads to that, more direction vectors are constructed, which affects the accuracy of data extraction. The experiment is carried out by selecting n from 1 to 6.

Table 3 shows the effect of the embedding key and shared parameter on BER of data extraction. Figure 10(a) shows the effect of the embedding key and shared parameter on SNR_R of decrypted model. Figure 10(b) shows the effect of the embedding key and shared parameter on SNR_D of decrypted model. With the change of n from 1 to 4, there are corresponding values of φ with high accuracy and low distortion. When $n = 1$ and $\varphi = 110$, BER = 1.54%, $SNR_D = 22.13$, and $SNR_R = 37.63$. When $n = 2$ and $\varphi = 110$, BER = 2.25%, $SNR_D = 22.13$, and $SNR_R = 35.83$. When $n = 3$ and $\varphi = 110$, BER = 2.76%, $SNR_D = 22.13$, and $SNR_R = 35.12$. When $n = 4$ and $\varphi = 170$, BER = 2.88%, $SNR_D = 17.28$, and $SNR_R = 34.85$. It is observed that the proposed method has high accuracy, high embedding capacity, and low distortion when $n = 3$ and $\varphi = 110$.

Figure 11 shows decrypted models when $n = 3$, and φ changes from 90 to 150. Figure 12 shows decrypted model when n changes from 1 to 4.

4.4. The Effect of the Number of the Vertices on SNR_R. Table 4 shows the relationship between the number of the vertices and SNR_R. If a 3D model has a large number of vertices, then the distance between the adjacent vertices will become small, which makes 3D model has a small D. In addition, D directly influence SNR_R because of the relationship between D and ϕ. Hence, if a 3D model has more vertices, then the value of SNR_R will be small, which means that the decrypted model has low distortion.

4.5. Performance Comparison. In order to show the performance of the proposed method, we compare the proposed method with the existing method in [26] and in [28], as shown in Table 5. Compared with method in [26], the proposed method has three times the capacity and lower distortion than method in [26]. Moreover, the proposed has a lower BER, which can be reduced to zero by making the embedding key satisfy Equation (27).

Compared to method in [28], the embedding capacity of the proposed method is smaller. However, since several bits can be mapped to a direction vector using the mapping table, the capacity can be improved by embedding several bits into a vertex. Hence, the proposed method has a higher embedding capacity.

In the proposed method, the distortion and the accuracy can be adjusted by using the embedding key. When the embedding key is 110, 3D models after data hiding has lower distortion. When the embedding key is 250, the BER of data extraction can be reduced to zero. In addition, the distortion will increase as the embedding key increases, the BER will decrease as the embedding key decreases.

5. Conclusion

The method is proposed to preserve privacy and protect copyright of 3D models. Moreover, the proposed method has very good potential for practical applications since the directly decrypted models have lower distortion than the original models. Original 3D model is preprocessed, and the vertex of 3D model is encrypted by using Paillier cryptosystem. In the cloud, the dyeing method is designed to classify all vertices into the embedded set and the referenced set. After that, secret data is embedded by expanding direction of prediction error with direction vector. The prediction error of the vertex in the embedded set is computed by using the referenced set, and the direction vector is obtained according to the mapping table. Secret data can be extracted by comparing the angle between the direction of prediction error and direction vector, and the original model can be restored using the referenced set. The proposed method is efficient to protect copyright of 3D models in the cloud when the cloud administrator does not know the content of the 3D models. Moreover, the proposed has higher capacity and lower distortion than the existing methods.

For the future work of RDH-ED method, we will investigate the following two possible research directions. (1) Extracting information from plaintext is expanded to extracting from plaintext and ciphertext. (2) Further improve the similarity between the directly decrypted model and the original model.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflict of interest.

Authors' Contributions

The conceptualization and funding acquisition are credited to Li Li. The methodology and writing-original draft are due to Shengxian Wang. The conceptualization and supervision are credited to Ting Luo. The writing-review and editing are credited to Ching-Chun Chang. English grammar is credited to Qili Zhou. Formal analysis and investigation are originated by Hui Li. All authors read and approved the final manuscript.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (No. 61370218 and No. 61971247), Public Welfare Technology and Industry Project of Zhejiang Provincial Science Technology Department (No. LGG19F0-20016), and Ningbo Natural Science Foundation (No. 2019A610100).

References

[1] X. Gao, L. An, X. Li, and D. Tao, "Reversibility improved lossless data hiding," *Signal Processing*, vol. 89, no. 10, pp. 2053–2065, 2009.

[2] W. Liang, J. Long, C. Li, and J. Xu, "A fast defogging image recognition algorithm based on bilateral hybrid filtering," *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2020.

[3] W. Liang, W. Huang, J. Long, and K. Zhang, "Deep reinforcement learning for resource protection and real-time detection in IoT environment," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6392–6401, 2020.

[4] J. Chen, F. Peng, J. Li, and M. Long, "A lossless watermarking for 3D STL model based on entity rearrangement and bit mapping," *International Journal of Digital Crime and Forensics*, vol. 9, no. 2, pp. 25–37, 2017.

[5] Y. H. Huang and Y. Y. Tsai, "A reversible data hiding scheme for 3D polygonal models based on histogram shifting with high embedding capacity," *3D Research*, vol. 6, no. 2, pp. 1–12, 2015.

[6] I. C. Dragoi and D. Coltuc, "Local-prediction-based difference expansion reversible watermarking," *IEEE Transaction on Image Processing*, vol. 23, no. 4, pp. 1779–1790, 2014.

[7] X. Zhang and S. Wang, "Fragile watermarking with error-free restoration capability," *IEEE Transaction on Multimedia*, vol. 10, no. 8, pp. 1490–1499, 2008.

[8] C. De Vleeschouwer, J.-F. Delaigle, and B. Macq, "Circular interpretation of bijective transformations in lossless watermarking for media asset management," *IEEE Transactions on Multimedia*, vol. 5, no. 1, pp. 97–105, 2003.

[9] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Transaction on Circuits and Systems*, vol. 13, no. 8, pp. 890–896, 2003.

[10] Y. Hu, H. K. Lee, K. Chen, and J. Li, "Difference expansion based reversible data hiding using two embedding directions," *IEEE Transaction on Multimedia*, vol. 10, no. 8, pp. 1500–1512, 2008.

[11] A. Poljicak, L. Mandic, and D. Agic, "Discrete Fourier transform–based watermarking method with an optimal implementation radius," *Journal of Electronic Imaging*, vol. 20, no. 3, 2011.

[12] B. Chen, G. Coatrieux, G. Chen, X. Sun, J. L. Coatrieux, and H. Shu, "Full 4- D quaternion discrete Fourier transform based watermarking for color images," *Digital Signal Processing*, vol. 28, pp. 106–119, 2014.

[13] B. Ou, X. Li, Y. Zhao, R. Ni, and Y. Q. Shi, "Pairwise prediction-error expansion for efficient reversible data hiding," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5010–5021, 2013.

[14] X. W. Li and S. T. Kim, "Optical 3D watermark based digital image watermarking for telemedicine," *Optics and Lasers in Engineering*, vol. 51, no. 12, pp. 1310–1320, 2013.

[15] Z. Ni, Y. Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Transaction on Circuits Systems for Video Technology*, vol. 16, no. 3, pp. 354–362, 2006.

[16] P. Tsai, Y. C. Hu, and H. L. Yeh, "Reversible image hiding scheme using predictive coding and histogram shifting," *Signal Processing*, vol. 89, no. 6, pp. 1129–1143, 2009.

[17] C. Y. Lin and S. F. Chang, "Semi-fragile watermarking for authenticating JPEG visual content," *Proceeding of Electronic Imaging*, vol. 3971, pp. 140–151, 2000.

[18] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," *IEEE Transaction on Information Forensics and Security*, vol. 8, no. 3, pp. 553–562, 2013.

[19] X. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Processing Letters*, vol. 18, no. 4, pp. 255–258, 2011.

[20] X. Cao, L. Du, and X. Wei, "High capacity reversible data hiding in encrypted images by patch-level sparse representation," *IEEE Transaction on Cybernetics*, vol. 46, no. 5, pp. 1132–1143, 2016.

[21] M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran, "On compressing encrypted data," *IEEE Transaction on Signal Processing*, vol. 52, no. 10, pp. 2992–3006, 2004.

[22] W. Liu, W. Zeng, and L. Dong, "Efficient compression of encrypted grayscale images," *IEEE Transaction on Image Processing*, vol. 19, no. 4, pp. 1097–1102, 2010.

[23] W. Hong, T. S. Chen, and H. Y. Wu, "An improved reversible data hiding in encrypted images using side match," *IEEE Signal Processing Letters*, vol. 19, no. 4, pp. 199–202, 2012.

[24] H. T. Wu and Y. M. Cheung, "A reversible data hiding approach to mesh authentication," in *Proceedings of International Conference on Web Intelligence*, pp. 774–777, 2005.

[25] C. Y. Jhou, J. S. Pan, and D. Chou, "Reversible data hiding base on histogram shift for 3D vertex," *International Conference on International Information Hiding and Multimedia Signal Process*, 2007, pp. 365–368, Kaohsiung, Taiwan, 2007.

[26] R. Q. Jiang, W. M. Zhang, and N. H. Yu, "Reversible data hiding in encrypted three-dimensional mesh models," *IEEE Transaction on Multimedia*, vol. 20, no. 1, pp. 55–67, 2018.

[27] M. Shah and W. M. Zhang, "Homomorphic encryption-based reversible data hiding for 3D mesh models," *Arabian Journal for Science and Engineering*, vol. 43, no. 12, pp. 8145–8157, 2018.

[28] L. Li, S. X. Wang, S. Q. Zhang, and T. Luo, "Homomorphic encryption-based robust reversible watermarking for 3D model," *Symmetry*, vol. 12, no. 3, pp. 347–365, 2020.

[29] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of the International Conference on Advance in Cryptology-Eurocrypt*, pp. 233–238, 1999.

[30] K. Donald, *The Art of Computer Programming*, Addison-Wesley, Massachusetts, USA, 3rd edition, 1997.