High Capacity Reversible Data Hiding in Encrypted 3D Mesh Models Based on Dynamic Prediction and Virtual Connection

Ke Wang, Ye Yao, Yanzhao Shen, Fengjun Xiao, Yizhi Ren, and Weizhi Meng, *Senior Member, IEEE*

***Abstract*—In recent years, reversible data hiding in encrypted domain (RDH-ED) has garnered considerable interest among researchers, resulting in the development of high-performance methods based on various carriers. However, the challenge of enhancing the data embedding capacity while ensuring reversibility becomes increasingly pronounced when the carrier is a three-dimensional (3D) model. In this paper, a high capacity RDH-ED method based on dynamic prediction and virtual connection for 3D models is proposed. Unlike existing methods that partition the vertices in the model into embeddable and prediction sets, where each vertex can only serve one function, the proposed dynamic prediction mechanism constructs a data embedding order set by leveraging the connectivity relationships between vertices. This allows each vertex within the set to both embed data and provide predictions, significantly increasing the proportion of embeddable vertices. Moreover, the proposed method is the first work to consider independent vertices within the model and integrates a novel virtual connection approach with the dynamic prediction process, enabling all independent vertices to participate in data embedding and prediction, thereby further enhancing the data embedding capacity. Experimental results demonstrated that the proposed method significantly outperforms other state-of-the-art methods in terms of data embedding capacity while ensuring reversibility.**

***Index Terms*—Reversible data hiding, encrypted domain, three-dimensional model, dynamic prediction, virtual connection.**

I. Introduction

REVERSIBLE data hiding (RDH) is a technology that conceals private information within multimedia carriers, enabling both error-free data extraction and lossless carrier restoration. Due to distinctive features, it is frequently applied in crucial domains such as copyright protection [1], medical image processing [2], military intelligence transmission [3], etc. RDH methods can be primarily categorized into four types according to their implementation techniques, i.e., histogram shifting (HS) [4, 5], lossless compression [6, 7], prediction error (PE) expansion [8, 9], and multi-histogram modification (MHM) [10–13]. The pursuit of RDH methods that simultaneously achieve high data embedding capacity and low distortion of recovered image has always been a goal for numerous researchers.

With the rapid development of cloud technology in recent years, the transmission of images in a cloud environment may pose privacy leakage risks. When content owners send images to recipients, the transmitted images pass through third-party cloud servers for storage and forwarding, making the image content vulnerable to potential leaks. To address this issue, reversible data hiding in encrypted images (RDHEI) which requires image owners to encrypt images before transmission has been proposed. The third-party cloud servers cannot discern the original content of the images and can also embed necessary data, such as timestamps, user information, etc., into the encrypted images. Upon receiving encrypted images containing data, the recipients can execute data extraction or image recovery operations based on respective permissions. Based on the way of allocating embedding room, most existing RDHEI methods can be classified into two categories, i.e., reserving room before encryption (RRBE) [14–19] and vacating room after encryption (VRAE) [20–23]. For RRBE, researchers typically leverage strong correlations between image pixels to allocate sufficient embedding space. However, because the pixel values in encrypted images are uncertain, the data embedding capacity achievable by VRAE may be constrained compared to what RRBE can achieve.

As the demand for higher visual fidelity in images continues to rise, three-dimensional (3D) models have emerged to play a significant role in virtual reality, engineering applications, education, training, etc. RDH on encrypted 3D models in a cloud environment offers an effective means to secure content and embed essential data such as copyrights, thereby garnering increasing attention from researchers. Jiang *et al.* [24] were the first to propose an RDH method for encrypted 3D models, embedding data in the least significant bit (LSB) of vertex coordinate values. However, this approach has limited data embedding capacity and cannot guarantee error-free data extraction. Subsequently, Shah *et al.* [25] utilized homomorphic encryption technology to achieve higher embedding capacity, but this resulted in data expansion and significant computational cost. To address the data expansion issue observed in [25], a new RDH method [26] based on Paillier encryption

This work was supported in part by the Humanities and Social Sciences Foundation of Ministry of Education of China under Grant Number 23YJA870013, in part by the National Natural Science Foundation of China under Grant Number 62471167 and 72374058. The authors acknowledge the Supercomputing Center of Hangzhou Dianzi University for providing computing resources. *(Corresponding author: Ye Yao; Weizhi Meng.)*

Ke Wang, Ye Yao and Yizhi Ren are with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China (e-mail: yaoye@hdu.edu.cn).

Yanzhao Shen is with the Shandong Institute of Blockchain, Jinan 250102, China. Fengjun Xiao is with the Zhejiang Informatization Development Institute, Hangzhou Dianzi University, Hangzhou 310018, China.

Weizhi Meng is with the School of Computing and Communications, Lancaster University, United Kingdom, and also with the Department of Applied Mathematics and Computer Science, Technical University of Denmark, Denmark (e-mail: weme@dtu.dk).

for 3D models was proposed. The length of the encrypted bits corresponding to the vertex coordinates is reasonably controlled, but the recovered model is not entirely accurate. In [27], a separable RDH approach based on spatial subdivision and space encoding was designed to yield larger embedding capacity, and proper selection of multiple thresholds is required to avoid errors during data extraction phase. Unlike Jiang *et al.*'s method [24], which used the LSB for data embedding, Xu *et al.* [28] and Yin *et al.* [29] attempted to explore the relationship between the most significant bit (MSB) of original coordinate values and their predictive values, resulting in improved vertex utilization and higher data embedding capacity compared to [24]. Furthermore, Lyu *et al.* [30] and Tang *et al.* [31] employed an odd-even strategy to divide vertices into embeddable and prediction sets, generating the redundant room for each embeddable vertex based on comparisons of multi-MSB. Recent studies [32–35] have increasingly complicated the partitioning strategy to enhance vertex utilization, Tsai *et al.* [32] used a strategy of random sampling to reduce the number of reference vertices; A new approach that adaptively partitions a 3D model into multiple sub-blocks and uses the most central vertex in each sub-block to predict other vertices was proposed in [33] and [35]; Gao *et al.* [34] introduced a vertex grouping method in which each group contains a single predictive vertex, and the encryption of 3D models is based on secret sharing over Galois field.

Many of the methods mentioned above aim to minimize the number of reference vertices without significantly affecting prediction accuracy, thereby increasing embedding capacity. While Hou *et al.*'s method [33] has already achieved a high level of vertex utilization, there is still room for improvement. In this paper, we introduce a dynamic prediction-based method where vertices can serve for both data embedding and predicting adjacent vertices. Additionally, to utilize vertices not situated on the face, we establish virtual connections between them and other vertices. Finally, the original vertex coordinate values are compared with corresponding predictive values through multi-MSB prediction, and the resulting labels are further reduced in length through entropy encoding. In the proposed method, the proportion of embeddable vertices approaches nearly 100%. Experimental results demonstrate that the data embedding capacity of our scheme significantly outperforms existing state-of-the-art methods.

The main contributions of this paper are the following:

1) A dynamic prediction mechanism has been proposed to construct a sequential set of embeddable vertices within a model. The vertices of this collection serve a dual purpose: they are utilized not only for embedding data but also for providing predictions for other vertices. The proportion of embeddable vertices approaches nearly 100%.

2) In order to endow the independent vertices within the model with capabilities for data embedding and prediction, a novel approach called virtual connection has been integrated into the dynamic prediction process, thereby enhancing the utilization of embeddable vertices.

3) The multi-MSB prediction strategy is employed at each embeddable vertex and new vertex which is generated by integrating the average features of all vertices in the corresponding prediction set.

4) We adaptively apply arithmetic coding or different Huffman indicators based on the occurrence frequency of embedding labels so as to further reduce the length of auxiliary information.

The rest of this paper is organized as follows. Section II introduces three related works [30, 32, 33] based on vertices division and multi-MSB prediction. The proposed method using dynamic prediction and virtual connection is described in Section III. Then, the experimental results including settings analysis and embedding performance comparison are given in Section IV. Finally, Section V concludes this paper.

II. Related work

In this section, seven related studies, categorized into MSB prediction-based [28–31] and partition strategy-based [32–34] methods, are introduced in detail. All these methods begin by partitioning all vertices in the 3D model into embeddable and prediction sets based on different rules. The MSB prediction-based methods employ simple vertex partitioning strategies, with capacity performance improvements primarily relying on MSB comparison. To further enhance vertex utilization, increasingly complex partitioning strategy-based methods have emerged and we will only focus on introducing vertex partitioning rules across various methods.

A. MSB Prediction-based Methods

A 3D model M is composed of two fundamental sets, with the vertices set denoted as $V = \{v_i\}_{i=1}^{p}$, and the face set denoted as $F = \{f_c\}_{c=1}^{q}$. Here, p and q represent the number of vertices and faces, respectively. Each vertex v_i consists of three coordinate values, namely $v_{i,x}$, $v_{i,y}$, and $v_{i,z}$. Additionally, each face f_c is composed of three vertices. Since the original vertex coordinate values $v_{i,x}$, $v_{i,y}$, and $v_{i,z}$ are all in decimal format, they should be initially converted into integers for ease of processing. In MSB prediction-based methods [28–31], the decimal point of the original vertex coordinate values is shifted to the right by u positions, and the remaining decimal part is truncated. Assuming that the integer coordinate value $\tilde{v}_{i,x}$ is derived from the decimal value $v_{i,x}$, it can be represented using the following equation:

$$\tilde{v}_{i,x} = \lfloor v_{i,x} \times 10^u \rfloor \tag{1}$$

where $\lfloor . \rfloor$ represents the floor function. When performing a reversible operation, the recovered coordinate value $\hat{v}_{i,x}$ is given by:

$$\hat{v}_{i,x} = \tilde{v}_{i,x} / 10^u \tag{2}$$

In comparison to $v_{i,x}$, $\hat{v}_{i,x}$ experiences a reduction in precision. Thus, the aforementioned conversion is inherently lossy, with the extent of precision loss being contingent upon the parameter u. The value of u also dictates the bits length l

required for the computer to represent the integer coordinate value $\tilde{v}_{i,x}$:

$$l = \begin{cases} 8, & 1 \leq u \leq 2 \\ 16, & 3 \leq u \leq 4 \\ 32, & 5 \leq u \leq 9 \\ 64, & 10 \leq u \leq 33 \end{cases} \tag{3}$$

All methods first initialize an empty embedding set S_e and an empty prediction set S_p. In [28] and [29], the face set F is traversed in ascending order and for each face f_c, if none of the vertices in f_c are in S_e or S_p, the first vertex of f_c is added to S_e, and the remaining two vertices are added to S_p. Then the coordinate values of all vertices are converted into signed binary sequences of length l. For each vertex v_i in the S_e set, assuming the vertex has an integer coordinate value $\tilde{v}_{i,x}$ along the x-axis, its transformed binary sequence is $\{\tilde{v}^t_{i,x}\}^l_{t=1}$. [28] uses the MSBs of the binary x-axis coordinates of all vertices connected to v_i in S_p for comparison and prediction of $\tilde{v}^1_{i,x}$. The index i of embeddable vertex v_i will be recorded as auxiliary information. However, the comparison of single MSB significantly limits the payload capacity of 3D mesh models. Thus [29] compares $\{\tilde{v}^t_{i,x}\}^l_{t=1}$ and multi-MSB of binary predictive vertices coordinates along x-axis. If the predicted results for the current bit differ, stop the comparisons and the predicted length r_1 along the x-axis can be determined. Following the same procedure, compare the y- and z-axis coordinate values to obtain corresponding predicted lengths r_2 and r_3. Finally, the minimum value of r_1, r_2 and r_3 is taken as the label r_m which indicates the current vertex v_i can embed $3 \times r_m$ bits of data and needs to be embedded into the 3D model as auxiliary information to ensure accurate data extraction and reversible model recovery. Although multi-MSB prediction can enhance vertex embedding capacity, the limited number of embeddable vertices restricts the algorithm's performance. Lyu *et al.* [30] improved method [29] by simply classifying vertices with odd (even) indices into the embedding set S_e and vertices with even (odd) indices into the prediction set S_p. This approach allowed half of the vertices in the 3D mesh model to be used for data embedding based on multi-MSB prediction, significantly improving embedding performance. Building on [30], Tang *et al.* [31] added some vertices from S_p that were not suitable for prediction into S_e, enabling multi-MSB prediction to be applied to more embeddable vertices. Since then, researchers have increasingly focused on improving the proportion of embeddable vertices without compromising the effectiveness of multi-MSB prediction. More complex vertex partitioning strategies have been proposed, showing more promise and advantages in enhancing embedding compared to MSB prediction-based methods.

B. Partition Strategy-based Methods

In previous MSB prediction-based scheme [28–31], vertices with negative coordinate values cannot be effectively used for data embedding. To address this issue, Tsai *et al.* [32] proposed a coordinate transformation process. Firstly, calculate the minimum value of each axis coordinate among all vertices using the following equation:

$$\begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix} = \begin{bmatrix} \min_{i\in\{1,2,\ldots,p\}}(v_{i,x}) \\ \min_{i\in\{1,2,\ldots,p\}}(v_{i,y}) \\ \min_{i\in\{1,2,\ldots,p\}}(v_{i,z}) \end{bmatrix} \tag{4}$$

Then, to ensure that the coordinate values of each vertex fall within the range of 0 to 1, a coordinate transformation process is given by:

$$\begin{bmatrix} \overline{v}_{i,x} \\ \overline{v}_{i,y} \\ \overline{v}_{i,z} \end{bmatrix} = \begin{bmatrix} (v_{i,x} - x_m)/10^b \\ (v_{i,y} - y_m)/10^b \\ (v_{i,z} - z_m)/10^b \end{bmatrix} \tag{5}$$

where b is the longest integer digit of all the shifted vertex coordinate values. Finally, all non-negative coordinate values are converted to integers using parameter u and the Eq. (1). Tsai *et al.* further utilized selection key K_s and random sampling to select $\lfloor T_s \times N_{V_k} \rfloor$ neighboring vertices for each embeddable vertex V_k as references, where T_s is a selection threshold and N_{V_k} represents the number of neighboring vertices around vertex V_k. The embeddable length was then constructed by using multi-MSB prediction between the embeddable vertex and its corresponding reference vertices. The proposed random sampling strategy eliminates the need for specific partitioning rules, thereby increasing the number of embeddable vertices that meet the conditions. However, the proportion of predictive vertices remains relatively large and requires further optimization.

Hou *et al.* [33] employed the same coordinate transformation process as described in [32] to convert the decimal form of vertex coordinate values into integers. Subsequently, octree spatial subdivision was proposed to divide a 3D model M into n sub-blocks from top to bottom. The result of the partitioning can be described as follows:

$$M = \left\{ \begin{matrix} sb_1, sb_2, \cdots, sb_a, \cdots, sb_n | Depth(sb_a) \leq \\ MaxD \quad or \quad Size(sb_a) \leq MaxS \end{matrix} \right\} \tag{6}$$

where sb_a represents the a^{th} sub-block, $Depth(sb_a) \leq MaxD$ indicates that the tree depth cannot exceed the boundary volume $MaxD$ and $Size(sb_a) \leq MaxS$ signifies the maximum number of vertices in each sub-block should be $MaxS$. For a sub-block, if the embedding room vacated by the vertices is less than the embedding threshold T_D, the sub-block will be considered as a non-embeddable block. All embeddable sub-blocks will be sorted in descending order according to the number of vertices, while the ordinal index of vertices will be updated after reordering. In an embeddable sub-block, one vertex is arbitrarily selected as a reference and compared with other vertices using multi-MSB prediction for vacating room. The octree spatial subdivision allows the proportion of embeddable vertices to reach up to 95%, leaving relatively few predictive vertices. However, it is still regrettable that these few vertices are overlooked and not used for data embedding.

The adaptive vertex grouping strategy was proposed in [34]. It begins by sorting the faces F in ascending order of vertex indices both row wise and column wise to form F'. Subsequently, all independent triangular faces in F' are collected to form the initial group set $G = \{g_1, g_2, \ldots, g_t\}$.

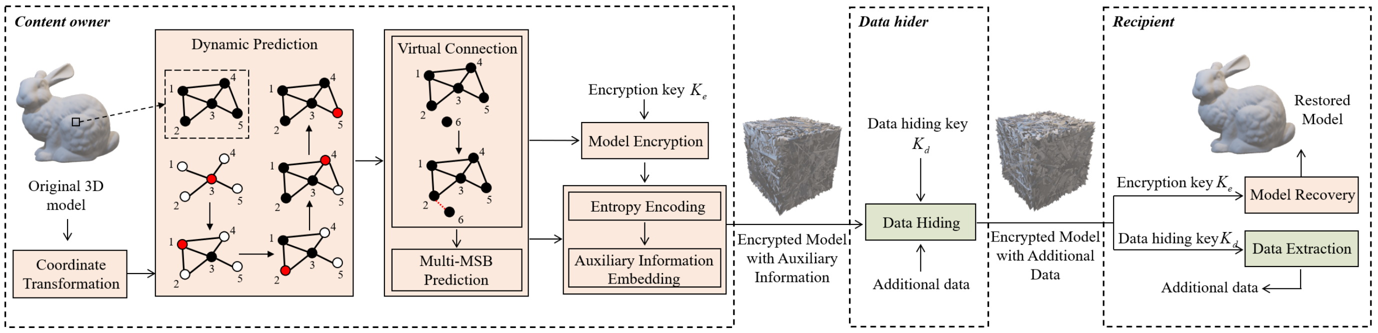


Fig. 1. The framework of the proposed method.

For vertices not included in any group, connected vertices that are already in G are identified in F' in sequential order, and the ungrouped vertices are added to the groups of their connected vertices. After grouping the vertices, the first vertex in each group serves as a reference for the other remaining vertices and is not used for data embedding. The number of groups determines the number of predictive vertices, so this partitioning mechanism still presents a bottleneck in enhancing the embedding capacity of 3D models.

Based on the above analysis, existing vertex partitioning strategies can still be further optimized to ultimately achieve a nearly 100% utilization rate of embeddable vertices. The more embeddable vertices there are, the greater the embedding capacity of the model, provided that the effectiveness of multi-MSB prediction is not compromised. To address this consideration, dynamic prediction and virtual connection mechanisms have been proposed to increase the payload upper bound of the RDHED method.

III. Proposed Method

In this paper, we propose a high capacity reversible data hiding method that combines dynamic prediction and virtual connection applied to encrypted 3D models. The framework of our proposed method is illustrated in Fig. 1, consisting of three main parts: 1) The content owner frees up redundant space through coordinate transformation, dynamic prediction, virtual connection, and Multi-MSB prediction. Subsequently, the encrypted model is embedded with auxiliary information which is constructed by entropy encoding and sent to the data hider. During the dynamic prediction process, a vertex serves as the starting point and gradually spreads to the surrounding vertices. The transition of red vertices indicates the order of dynamic prediction. Besides, the virtual connection is applied between the independent vertex and the nearest vertex in space, allowing the independent vertex to participate in dynamic prediction for data embedding and prediction. 2) Upon receiving the encrypted model, the data hider performs pre-encrypted data embedding and transfers the resulting model to recipients with different access privileges. 3) Different keys determine whether the recipient can successfully execute data extraction or model recovery processes. Modules of the same color in the diagram represent reversible processes.

A. Coordinate Transformation Process

To facilitate subsequent data processing, the coordinate transformation process involves converting the decimal-format coordinate values of vertices in a 3D model M into integers. Following the approach similar to Tsai *et al.*'s method [32], the minimum boundary values x_m, y_m, and z_m of each axis are firstly derived through Eq. (4). Subsequently, the original coordinate values are constrained within the 0 to 1 range using the edge values in accordance with the transformation process defined by Eq. (5), resulting in $\overline{v}_{i,x}$, $\overline{v}_{i,y}$, and $\overline{v}_{i,z}$. Eventually, these decimal coordinate values are converted into integers by Eq. (1) for different values of the parameter u.

Assuming that the value of (x_m,y_m,z_m) is $(-0.37,-0.58,-0.52)$ after the calculation of Eq. (4), and the maximum values of $v_{i,x}$, $v_{i,y}$ and $v_{i,z}$ are 0.77, 0.61 and 0.66 respectively, then b is obtained to be 1 by Eq. (5). When the parameter u is 3, a vertex with the original coordinate value of $(0.72, 0, -0.11)$ is transformed into fractional and integer coordinates of $(0.109, 0.058, 0.041)$ and $(109, 58, 41)$, respectively.

B. Dynamic Prediction

Under typical circumstances, the content owner needs to divide the vertices in a 3D model into two distinct sets: the embeddable set S_e and the prediction set S_p. Vertices within the set S_e are utilized for data embedding, whereas those in the set S_p are employed to predict the surrounding embeddable vertices. To ensure reversibility, the coordinate values of the predictive vertices remain unaltered and thus do not participate in data embedding. Numerous previous methods have aimed to minimize the number of predictive vertices as much as possible without significantly affecting prediction accuracy. This reduction has facilitated an increase in the number of embeddable vertices, thereby improving the data embedding capacity of the entire model. However, to maintain the accuracy of predictions, the number of predicted vertices cannot be reduced indefinitely, so it becomes increasingly difficult to augment the model's embedding capacity. To address this issue, this paper proposes a dynamic prediction method wherein the number of predictive vertices surrounding a given vertex changes dynamically. The vertices in the model can be used for both prediction and data embedding. The specific steps will be described in the following:

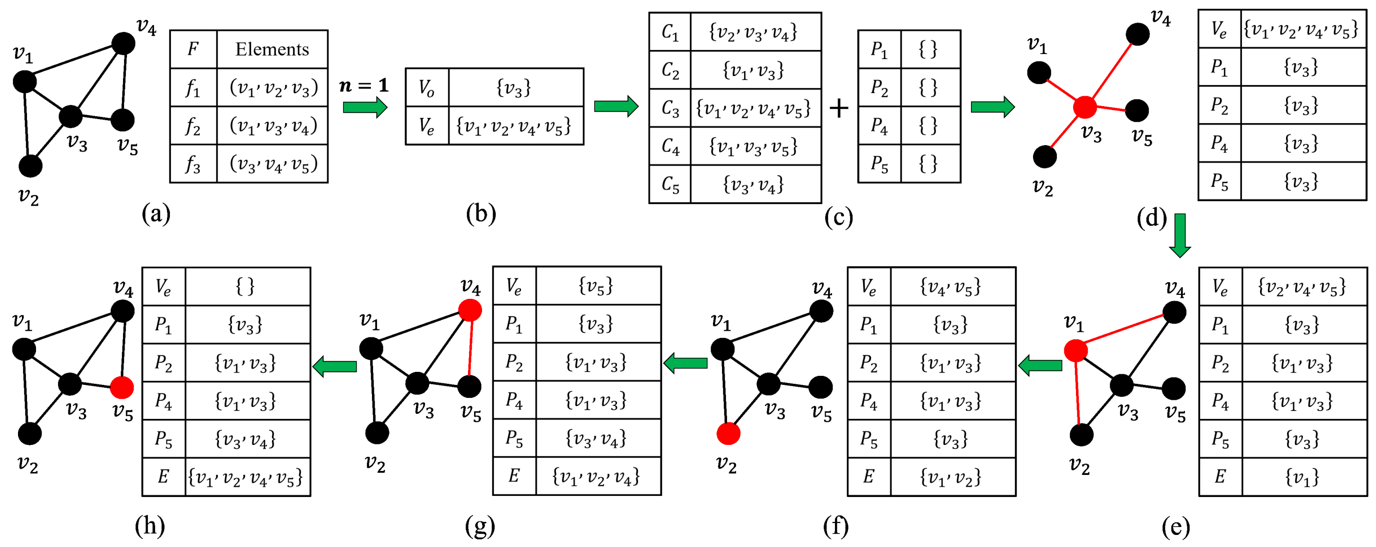


Fig. 2. An example of dynamic prediction in a simple model.

(1) **Sort vertices:** Through the face set $F = \{f_c\}_{c=1}^{q}$, the number of connected vertices around each vertex v_i in the model can be calculated sequentially. Then all p vertices are sorted in descending order based on the counts. The top n vertices are selected to form the set V_o, while the remaining vertices constitute the set V_e. Here, n is a threshold with a minimum value of 1 and all vertices in V_e will be used for data embedding.

(2) **Initialize the vertices in $\boldsymbol{V_o}$ and $\boldsymbol{V_e}$:** For each vertex v_i in the model, initialize a set C_i containing all the neighboring vertices connected to v_i. Additionally, assuming that the vertices in sets V_o and V_e are represented by v_j and v_k respectively, create a corresponding set P_k for each vertex v_k. The set P_k is intended to include other vertices that will be used to predict v_k, and it is initially an empty set.

(3) **Update set $\boldsymbol{P_k}$:** Iterate over each vertex v_j in the set V_o to get its corresponding set C_j. For every vertex denoted as $v_{j'}$ in the set C_j, if it belongs to the set V_e, then add the vertex v_j to the corresponding set $P_{j'}$ of vertex $v_{j'}$. As a result, some vertices v_k in V_e will have non-empty corresponding sets P_k.

(4) **Construct embedding vertex sequences:** First an empty set E is created for representing the embedding order of all vertices in the set V_e. Second, each vertex in V_e are sorted according to the number of vertices in corresponding set P_k from largest to smallest, and the first vertex v_k in the sorted set V_e is deleted as well as added to the set E. Then iterate over each vertex denoted as $v_{k'}$ in the set C_k, and if the current vertex belongs to the set V_e, the vertex v_k is added to the corresponding set $P_{k'}$ of vertex $v_{k'}$. Finally, the revised set V_e is sorted again and the subsequent processes described above are repeated until there is no more vertex in the set V_e. The final generated set E not only contains all the vertices in the original set V_e, but also further represents the order in which vertices are embedded into the data. Besides, all the vertices in the set P_k are used for predicting the vertex in E.

In order to better understand the steps described above, Fig. 2 shows the complete processes of dynamic prediction using a simple model as an example. As shown in Fig. 2(a), the model consists of 5 vertices and 3 faces. When the threshold value n is set to 1, it can be observed that the vertex v_3 has the maximum number of neighboring vertices connected to it. Therefore, the resulting set V_o is equal to $\{v_3\}$, and the set V_e contains the remaining vertices. Subsequently, the vertices are initialized, where $C_1 \sim C_5$ respectively represent the sets of vertices connected to $v_1 \sim v_5$. Each vertex v_k in the set V_e corresponds to an initially empty prediction set P_k. As shown in Fig. 2(d), the unique vertex v_3 in the set V_o is traversed, highlighted in red. Based on the set C_3, it can be inferred that the vertices v_1, v_2, v_4, and v_5 connected to vertex v_3 belong to set V_e, with the connecting edges marked in red, thus the prediction sets P_1 P_2, P_4, and P_5 for the four connected vertices all include vertex v_3. Then the vertices in the set V_e are sorted in descending order based on the number of vertices in the set P_k. Since P_1, P_2, P_4, and P_5 have the same number of vertices, vertex v_1 is selected first. As shown in Fig. 2(e), vertices v_2 and v_4 connected to v_1 belong to set V_e, therefore vertex v_1 is added to the sets P_2 and P_4, and it is removed from the set V_e while being added to the set E. V_e is further sorted, and since both P_2 and P_4 have the maximum of 2 vertices, vertex v_2 is selected next in order. Vertices v_1 and v_3 connected to v_2 do not belong to set V_e, so vertex v_2 only needs to be removed from the set V_e and added to the set E, the resulting configuration is recorded in Fig. 2(f). Next taking vertex v_4 as the observation object in Fig. 2(g), vertex v_5 is connected to vertex v_4 in set V_e. Therefore, vertex v_4 is added to the sets P_5 and E, and the set V_e is left with only vertex v_5 after deleting v_4. Finally, the sole vertex v_5 is added to the set E, and the set V_e becomes empty. The final result is shown in Fig. 2(h), where the arrangement of vertices in set E represents the order of embedded data, and P_1, P_2, P_4, and P_5 are the corresponding prediction sets for these embedding vertices.

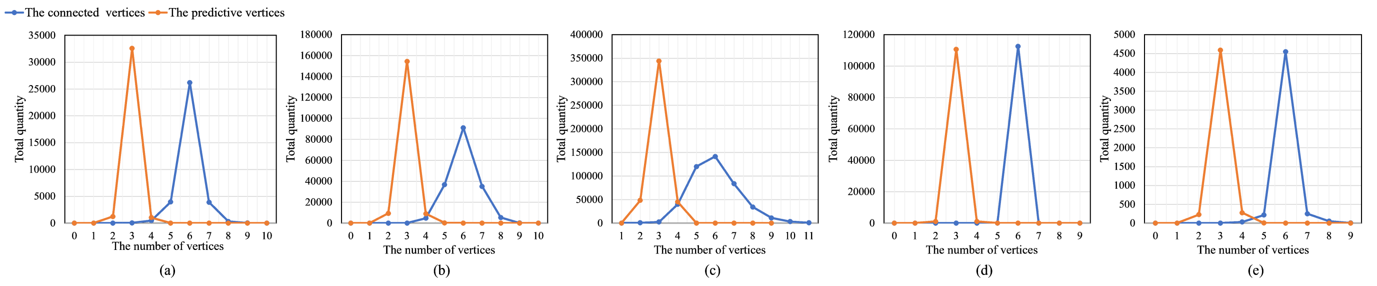


Fig. 3. The number distribution of the connected vertices and predictive vertices for each embeddable vertex in five test models ($n = 3$) : (a) Bunny; (b) Armadillo; (c) Dragon; (d) Horse; (e) Casting.

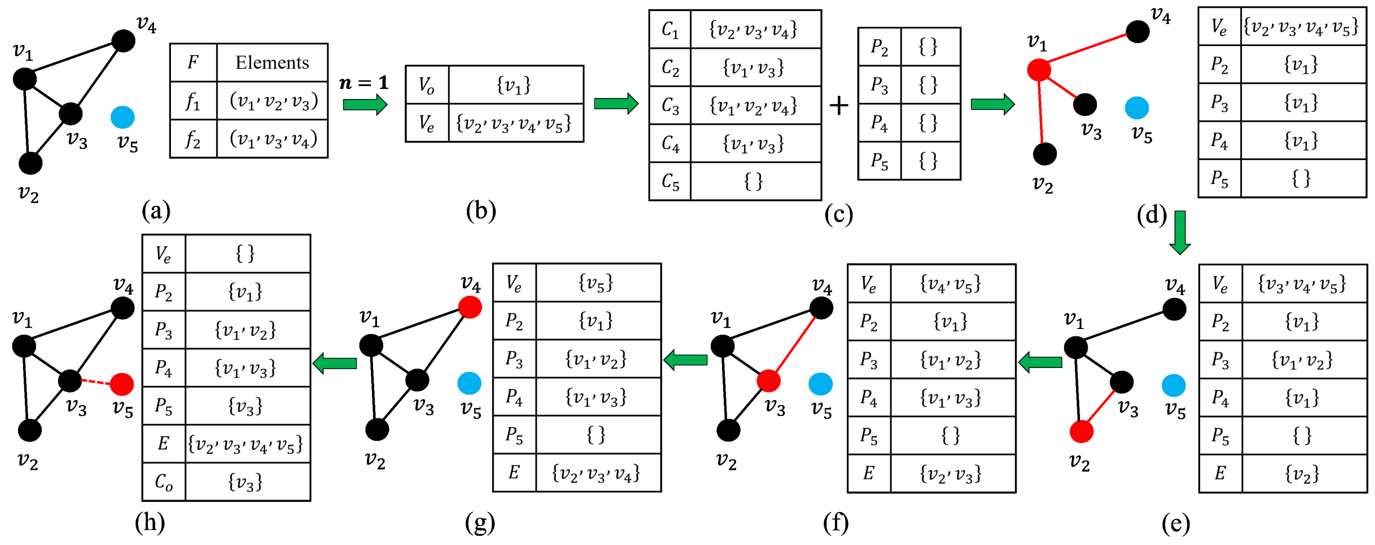


Fig. 4. Illustration of using virtual connection combined with dynamic prediction in a model.

The mechanism of dynamic prediction relies on providing predictions for the connected vertices based on an initial set V_o of n vertices, then these embeddable vertices are further utilized to predict adjacent vertices. The priority of vertex v_k being added to the embedding sequence set E increases as the number of predictive vertices around it grows dynamically. Taking five test models, ‘Bunny’, ‘Armadillo’, ‘Dragon’, ‘Horse’, and ‘Casting’ as examples, Fig. 3 shows the number distribution of the connected vertices and used predictive vertices for each embeddable vertex when the threshold n is set to 3. From the five subplots, it can be seen that the majority of embeddable vertices have 5, 6, or 7 connected vertices, with 3 usable predictive vertices. The distribution trend indicates that despite the differences in scale and number of vertices for each model, nearly half of the connected vertices around the embeddable vertices can be utilized as predictive vertices. This demonstrates that the dynamic prediction method can ensure an ample number of predictive vertices for each embeddable vertex, without compromising prediction accuracy as the proportion of embeddable vertices increases.

C. Virtual Connection

Considering that some models may contain independent vertices that are not connected to any other vertices, the original dynamic prediction method would result in empty prediction sets for these independent vertices, as there are no vertices to provide predictions. In order to utilize all independent vertices and increase the vertex utilization rate to maximize the embedding capacity of the model, we propose a virtual connection method based on dynamic prediction. Assuming a 3D model contains independent vertices, the dynamic prediction will only detect these vertices in the final stage within the set V_e while constructing the embeddable vertex sequence. Subsequently, all independent vertices will be added to the end of sequence E. In step 4 of dynamic prediction, when the set C_k corresponding to the traversed vertex v_k is empty, the current vertex is identified as an independent vertex. At this point, we calculate the squared distance value D^2 between the current independent vertex v_k and each vertex v_d within set $V_o \cup E$ in 3D space using the following equation:

$$D^2 = (v_{(k,x)} - v_{(d,x)})^2 + (v_{(k,y)} - v_{(d,y)})^2 + (v_{(k,z)} - v_{(d,z)})^2 \tag{7}$$

The vertex v_d with the smallest distance D^2 will be added to set P_k as the predictive vertex for vertex v_k, thereby establishing a virtual connection between the two vertices. The reason for selecting vertex from the set $V_o \cup E$ is that

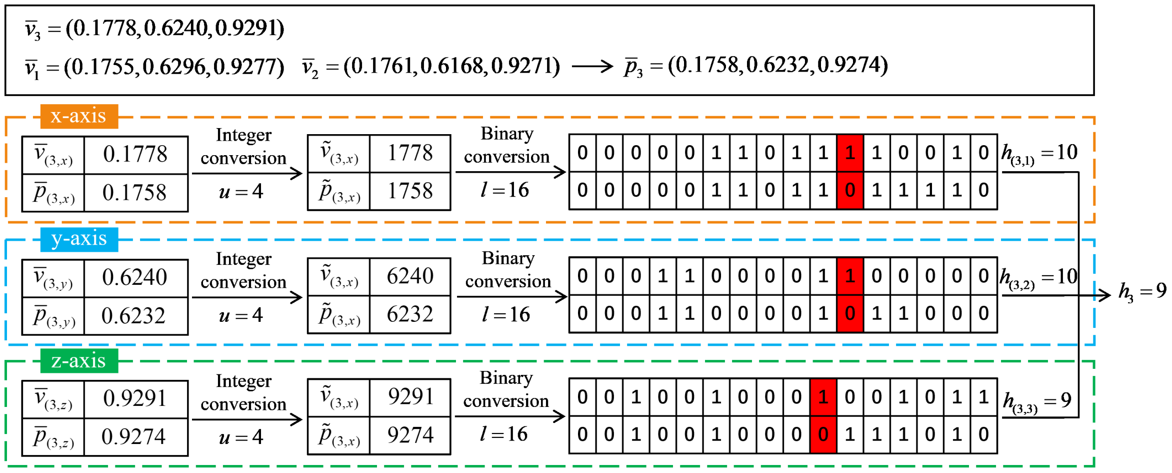


Fig. 5. An example of multi-MSB prediction for vertex v_3 in Fig. 4.

the coordinates of all vertices within this set are already known before predicting v_k. Additionally, a new set C_o is created to record all the vertices that are virtually connected to independent vertices in order to maintain reversibility. Each independent vertex in the model has only one corresponding virtual connection object, making the number of elements in the set C_o determinable.

Fig. 4 illustrates an example where the virtual connection method is applied in the model, with the vertex v_5 highlighted in blue being an independent vertex. Figs. 4(b)-(g) represent the processes of direct dynamic prediction, which do not specifically involve the vertex v_5, and therefore will not be further elaborated. As depicted in Fig. 4(h), the squared distances between the vertex v_5 and the four vertices v_1, v_2, v_3, and v_4 in the set $V_o \cup E$ are computed using Eq. (7). Assuming that vertex v_3 is the closest to vertex v_5, a virtual connection is then established between v_3 and v_5, resulting in v_3 being added to sets P_5 and C_o. Finally, v_5 is removed from the set V_e and included in the set E. The entire process combining dynamic prediction and virtual connection is outlined in Alg. 1.

Algorithm 1 Dynamic prediction and virtual connection.

```
Input: Vertices $V$, Faces $F$, Threshold $n$.
Output: Prediction set $P_k$, Embedding sequence set $E$, Virtual connection set $C_o$.
1:  Construct the set of connected vertices $C_i$ corresponding to each vertex $v_i$ from $F$;
2:  $V_o \leftarrow$ the top $n$ vertices with the highest number of elements in $C_i$;
3:  $V_e \leftarrow$ the remaining $p - n$ vertices;
4:  $P_k, E, C_o \leftarrow \emptyset$;
5:  for all $V_o$ do
6:      for all $C_j \cap V_e$ do
7:          $P_{j'} \leftarrow P_{j'} \cup \{v_j\}$;
8:      end for
9:  end for
10: while $V_e \neq \emptyset$ do
11:     $v_k \leftarrow$ the first vertex with the highest number of elements in $P_k$;
12:     $V_e \leftarrow V_e - \{v_k\}$; $E \leftarrow E \cup \{v_k\}$;
13:     if $C_k = \emptyset$ then
14:         $v_d \leftarrow$ the closest vertex to $v_k$ in set $V_o \cup E - \{v_k\}$;
15:         $P_k \leftarrow \{v_d\}$; $C_o \leftarrow C_o \cup \{v_d\}$;
16:         Continue;
17:     end if
18:     for all $C_k \cap V_e$ do
19:         $P_{k'} \leftarrow P_{k'} \cup \{v_k\}$;
20:     end for
21: end while
22: Return $P_k$, $E$, $C_o$.
```

D. Multi-MSB Prediction

For each embeddable vertex v_k in the set E, the corresponding embeddable bit length can be determined through multi-MSB prediction with all predictive vertices in set P_k. First, we obtain all vertices in set P_k and calculate their average coordinate values on the x-axis, y-axis, and z-axis to obtain a new vertex p_k. Using the coordinate transformation process mentioned in Section III-A, the original coordinate values of vertices v_k and p_k are transformed into non-negative integers, and the modified vertices are denoted as $\tilde{v}_k$ and $\tilde{p}_k$. Then convert the integer coordinate values $\tilde{v}_{k,x}$, $\tilde{v}_{k,y}$, and $\tilde{v}_{k,z}$ into l-bit binary sequences $\{\tilde{v}^t_{k,x}\}^l_{t=1}$, $\{\tilde{v}^t_{k,y}\}^l_{t=1}$, and $\{\tilde{v}^t_{k,z}\}^l_{t=1}$ by the following formula:

$$\tilde{v}^t_{k,s} = \left\lfloor \frac{\tilde{v}_{k,s} \bmod 2^{l+1-t}}{2^{l-t}} \right\rfloor, s = x, y, z \tag{8}$$

The same conversion is applied to vertex $\tilde{p}_k$. Furthermore, the binary sequences $\{\tilde{v}^t_{k,x}\}^l_{t=1}$ and $\{\tilde{p}^t_{k,x}\}^l_{t=1}$ are compared in the order of MSB to LSB. If the current compared bits are not equal, the comparison stops and the number of previous identical bits is recorded as $h_{(k,1)}$. The same process is repeated for the y-axis and z-axis binary coordinate values to obtain corresponding identical bit lengths $h_{(k,2)}$ and $h_{(k,3)}$. The minimum value among $h_{(k,1)}$, $h_{(k,2)}$, and $h_{(k,3)}$, denoted as h_k, is selected as the embedding label for vertex v_k. This signifies that the coordinate values of current vertex can

TABLE I
NUMBER AND HUFFMAN INDICATOR OF EACH POSSIBLE VALUE OF h_k FOR BUNNY MODEL.

Bunny	h_k							
	1	2	3	4	5	6	7	8
Number	0	1	15	1104	970	1609	3070	29177
Indicators	11110	00001	1110	110	0001	001	10	01

provide a redundant space of $3 \times h_k$ bits for data embedding.

Fig. 5 illustrates the process of calculating the embedding label of vertex v_3 in Fig. 4 using the multi-MSB prediction. Set P_3 contains two vertices, v_1 and v_2, used for predicting the vertex v_3. First, v_1, v_2, and v_3 are transformed into $\overline{v}_1$, $\overline{v}_2$, and $\overline{v}_3$ respectively through coordinate conversion. The coordinate values of $\overline{v}_1$, $\overline{v}_2$, and $\overline{v}_3$ range from 0 to 1, with specific values given in the figure. Next, the average coordinate values of $\overline{v}_1$ and $\overline{v}_2$ on different axes are calculated to form $\overline{p}_3$, which is used directly for multi-MSB prediction on $\overline{v}_3$. For the comparison on the x-axis, $\overline{v}_{(3,x)}$ and $\overline{p}_{(3,x)}$ are converted into integers using Eq. (1) with u set to 4, and the generated integers are further transformed into two binary sequences of length 16 using Eq. (3) and Eq. (8). The two binary sequences are compared from the MSB to LSB, and when the 11^{th} bit is reached, the two highlighted differing bits in the figure indicate an inconsistency. Therefore the comparison is stopped at this point, and the embeddable length $h_{(3,1)}$ for the x-axis coordinate value is determined to be 10. The same process is applied to the y-axis and z-axis comparisons, resulting in $h_{(3,2)} = 10$ and $h_{(3,3)} = 9$ respectively. Finally h_3 is determined to be 9 by taking the minimum value of $h_{(3,1)}$, $h_{(3,2)}$, and $h_{(3,3)}$, which implies that the current embeddable vertex v_3 can accommodate an additional 27 bits of data.

E. Model Encryption

After completing coordinate conversion, dynamic prediction, virtual connection, and multi-MSB prediction, the content owner needs to convert the integer representation $\tilde{v}_i$ of all vertices in the original model M into binary sequences of length l, denoted as $\{\tilde{v}^t_{1,x}\}^l_{t=1}, \ldots, \{\tilde{v}^t_{i,s}\}^l_{t=1}, \ldots, \{\tilde{v}^t_{p,z}\}^l_{t=1}$ respectively, according to Eq. (8). Subsequently, a binary sequence r_b of length $3 \times p \times l$ is randomly generated using the encryption key K_e. Then, r_b is evenly divided into $3 \times p$ binary sequences of length l, labeled as $\{r^t_{1,x}\}^l_{t=1}, \ldots, \{r^t_{i,s}\}^l_{t=1}, \ldots, \{r^t_{p,z}\}^l_{t=1}$. Each $\{r^t_{i,s}\}^l_{t=1}$ is used to encrypt the corresponding $\{\tilde{v}^t_{i,s}\}^l_{t=1}$ by performing a bitwise XOR operation:

$$e^t_{i,s} = \tilde{v}^t_{i,s} \oplus r^t_{i,s} \tag{9}$$

where $t = 1, 2, \ldots, l, i = 1, 2, \ldots, p, s = x, y, z$. $\{e^t_{i,s}\}^l_{t=1}$ represents the generated encrypted sequence and can be used to calculate the encrypted integer coordinate values denoted as $\tilde{v}'_{(i,s)}$ using the following formula:

$$\tilde{v}'_{(i,s)} = \sum_{t=1}^{l} e^t_{i,s} \cdot 2^{l-t} \tag{10}$$

Based on the generated $\tilde{v}'_{(i,s)}$, a new encrypted 3D model M_e can be fully constructed.

F. Entropy Coding and Auxiliary Information

To ensure the complete recovery of the original model, some certain auxiliary information needs to be embedded into the encrypted model in advance. After obtaining the label h_k for each embeddable vertex v_k in the set E through multi-MSB prediction, these labels need to be converted into a binary sequence and included as part of the auxiliary information. When the parameters u and n are both set to 2, the distribution of h_k for the test model 'Bunny' is presented in Table I. Since a mere 7-bit binary sequence can represent every integer coordinate value, the minimum value of h_k is 1. The table reveals an uneven frequency distribution of each possible value of h_k. In an effort to minimize the length of the auxiliary information, thus allocating more space for data embedding, we use common lossless entropy coding techniques including arithmetic coding and Huffman coding to compress all h_k. Here, taking Huffman coding as an example, shorter Huffman indicators are applied to values of h_k that are more prevalent. In this manner, each value of h_k in the table corresponds to a specific encoding, and the Huffman encoding rule $H =$'1111000001111011000010011001', formed by these indicators, are also included as auxiliary information for recovering the encoded h_k. In the model recovery stage, the rule H will be first extracted and then split into 8 Huffman codes, each corresponding to different h_k, facilitating the decoding of all h_k. Since arithmetic coding typically achieves better compression rates than Huffman coding, the choice of entropy coding method should depend on the final size of the auxiliary information to maximize the embedding space. For simplicity, the remainder of this section will assume the use of Huffman coding.

To obtain the objects of independent vertices in the virtual connection process, the auxiliary information should also include all vertices in set C_o. The index of each vertex is converted into a binary sequence of length $\lceil \log_2 p \rceil$ and arranged in order after the Huffman coding rules H and all encoded h_k. Subsequently, all auxiliary information needs to be embedded in specific positions within the encrypted model M_e. As discussed in Section III-E, the vertices of the encrypted model can sequentially be converted into $3 \times p$ encrypted binary sequences denoted as $\{e^t_{1,x}\}^l_{t=1}, \ldots, \{e^t_{i,s}\}^l_{t=1}, \ldots, \{e^t_{p,z}\}^l_{t=1}$. For ease of subsequent embedding operations, the first $h_i(h_k)$ embeddable bits of each $\{e^t_{i,s}\}^l_{t=1}$ are combined to form sequence L_1, while the last $l - h_i$ bits are combined to form sequence L_2. All auxiliary information will be used to replace some bits in L_1 sequentially. The modified L_1 is then concatenated with L_2 and converted back into vertex coordinate values in order. As illustrated in steps a, b, and c of Fig. 6, the encrypted vertex coordinate values are transformed into binary sequences according to different axes. The embeddable bits and the remaining bits are combined to form a long sequence $L_1 + L_2$ of length $3 \times p \times l$. Finally all auxiliary information is embedded into the sequence starting from the first bit, and the changed sequence is recursively

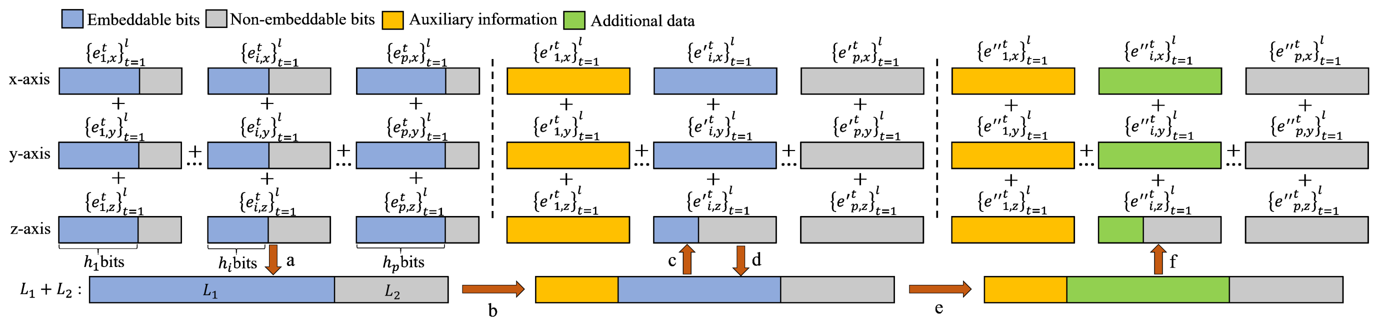


Fig. 6. Steps a, b, and c are the room distribution of auxiliary information embedding; Steps d, e, and f are the room distribution of data embedding.

decomposed into $\{e'^{t}_{1,x}\}_{t=1}^{l}$, …, $\{e'^{t}_{i,s}\}_{t=1}^{l}$, …, $\{e'^{t}_{p,z}\}_{t=1}^{l}$ to form a final encrypted model M_f containing the auxiliary information, which is sent by the content owner to the data hider.

G. Data Hiding

Upon receiving the model M_f sent by the content owner, the data hider first determines the number of independent vertices, denoted as $\hat{n}$, based on the face F. Then, all vertex coordinate values in the model are converted into binary form and recombined into a sequence of length $3 \times p \times l$. Starting from the first bit of the obtained sequence, the data hider extracts the Huffman coding rule H and decodes $p - n$ labels h_k of the embeddable vertices. Following this, the remaining part of the auxiliary information is extracted, which consists of $\hat{n} \times \lceil \log_2 p \rceil$ bits representing the indexes of $\hat{n}$ vertices in the set C_o. Additionally, the remaining bits in sequence L_1 that are not replaced by the auxiliary information, as discussed in Section III-F, can be identified and used to embed additional data. As shown in steps d, e, and f of Fig. 6, the remaining embeddable bits in the sequence are replaced with the data encrypted by a data hiding key K_d, and the changed sequence is recursively decomposed into $\{e''^{t}_{1,x}\}_{t=1}^{l}$, …, $\{e''^{t}_{i,s}\}_{t=1}^{l}$, …, $\{e''^{t}_{p,z}\}_{t=1}^{l}$ to form a marked model M_m containing the data, which is then sent to the recipient.

H. Data Extraction and Model Recovery

After receiving the model containing data, the recipient can extract all auxiliary information and encrypted data following the same procedures as the data hider. The possession of different keys determines whether the recipient can successfully extract the data or recover the model.

If the recipient only has the data hiding key K_d, the extracted encrypted data is directly decrypted to obtain the original data.

When only the encryption key K_e is available, the recipient first uses the face F for dynamic prediction to obtain the sets V_o, V_e, and E. Then, each vertex in set E corresponds with the tag value h_k, allowing the $3 \times p \times l$-length sequence to be decomposed into each l-length binary sequence combining embeddable bits and non-embeddable bits, decrypt each l-length binary sequence in order through $\{r^{t}_{i,s}\}_{t=1}^{l}$ generated by the encryption key K_e. Subsequently, by traversing each current coordinate value of the vertices v_k in set E and using h_k, the original vertex v_k can be restored according to all vertices in set P_k. If the current traversed vertex is an independent vertex, the only vertex virtually connected in set P_k can be obtained from the extracted auxiliary information. Finally, the coordinate values of all vertices are restored one by one, resulting in the final recovery of the original model.

If the recipient possesses both the data hiding key K_d and the encryption key K_e, not only the data, but the original model can be obtained correctly. Through the processes described above, it is evident that the proposed method is fully separable, allowing the recipient, upon obtaining the necessary authorization, to perform data extraction and model recovery in an arbitrary order.

IV. Experimental Results and Analysis

In order to demonstrate the performance of the proposed method, this section conducts an analysis of multiple performance parameters of the model, including security, separability, reversibility, and embedding capacity. In addition, we also compare the present method with state-of-the-art methods to fully demonstrate the superiority. All experiments were conducted using the Matlab R2021a software on the Windows 11 operating system. The test models used are displayed in Fig. 7, including Bunny, Armadillo, Dragon, Horse, and Casting. Furthermore, the Princeton Segmentation Benchmark (PSB) dataset [36] with 380 models will be utilized to further validate the universality of the performance. Two metrics, signal-to-noise ratio (SNR) and Hausdorff distance (HD), are used to measure the difference between the restored model and original model. Here, SNR primarily evaluates the geometric distortion and a higher SNR value indicates smaller geometric deformation of the 3D model. It can be calculated using the following formula:

$$SNR = 10 \times \lg \frac{\sum_{i=1}^{p}[(v_{i,x} - \overline{v}_x)^2 + (v_{i,y} - \overline{v}_y)^2 + (v_{i,z} - \overline{v}_z)^2]}{\sum_{i=1}^{p}[(v'_{i,x} - v_{i,x})^2 + (v'_{i,y} - v_{i,y})^2 + (v'_{i,z} - v_{i,z})^2]} \tag{11}$$

where $\overline{v}_x, \overline{v}_y, \overline{v}_z$ are the mean values of the original coordinates $v_{i,x}, v_{i,y}, v_{i,z}$, and $v'_{i,x}, v'_{i,y}, v'_{i,z}$ are the modified coordinates of the restored model. HD describes the similarity between the vertices sets of two models; the smaller the HD is, the more similar the two sets of vertices are. Assuming

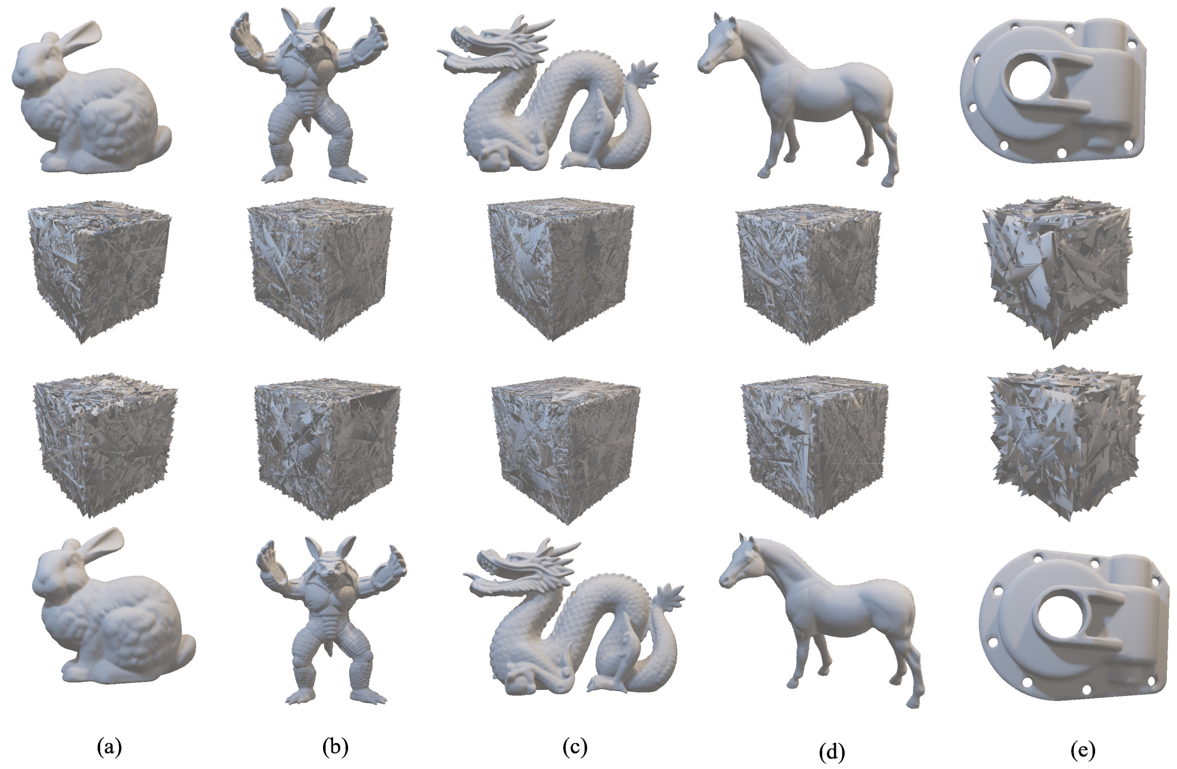


Fig. 7. The original models, final encrypted models, marked models, and restored models, arranged from top to bottom. (a) Bunny; (b) Armadillo; (c) Dragon; (d) Horse; (e) Casting.

the vertices sets of the two models being compared are $A = \{a_1, a_2, \ldots, a_p\}$ and $B = \{b_1, b_2, \ldots, b_p\}$, HD is defined as follows:

$$\begin{aligned} HD(A,B) &= \max(ed(A,B), ed(B,A)), \\ ed(A,B) &= \max_{a \epsilon A} \min_{b \epsilon B} \|a-b\|, \\ ed(B,A) &= \max_{b \epsilon B} \min_{a \epsilon A} \|b-a\| \end{aligned} \tag{12}$$

where $\|.\|$ calculates the Euclidean distance of the two sets of vertices. The embedding performance of our method is measured by the vertex embedding rate (ER), which indicates the average number of bits embedded per vertex. The unit is denoted as bits per vertex (bpv).

A. Security Discussion

In the entire process of the proposed method, the original model M is transformed by the content owner through vacating room and encryption to generate the final encrypted model M_f containing auxiliary information. After embedding data by the data-hider, the final encrypted model is further transformed into the marked model M_m. Both of these newly generated models replace the original model for transmission over the network, emphasizing the critical significance of their security. Fig. 7 shows the final encrypted and marked models generated by five test models. Visually, these models all exhibit good imperceptibility, meaning that attackers intercepting these models in the network would not be able to discern the original content. Furthermore, using Bunny as the test model, Fig. 8 illustrates the distribution of vertex coordinates for the models at the three aforementioned stages, where $\tilde{v}_{(i,s)}$, $\tilde{v}'_{(i,s)}$, and $\tilde{v}''_{(i,s)}$ represent the integer coordinate values of the original model M, the final encrypted model M_f, and the marked model M_m, respectively. In Fig. 8(a), the histograms display

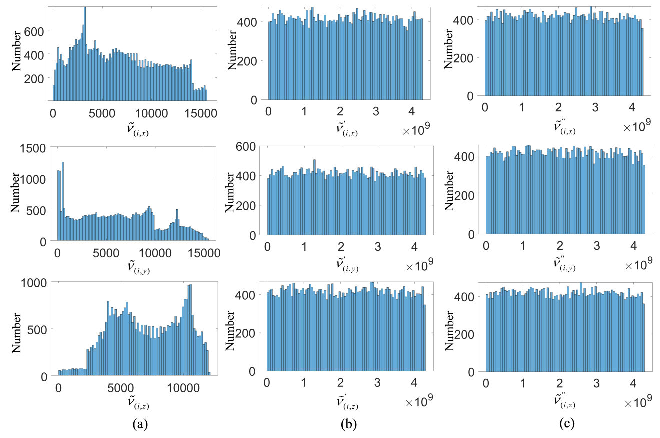


Fig. 8. Illustration of undetectability of the Bunny model in three stages: (a) is the distribution of integer coordinate values $\tilde{v}_{(i,x)}$, $\tilde{v}_{(i,y)}$, and $\tilde{v}_{(i,z)}$ in the original model; (b) is the distribution of integer coordinate values $\tilde{v}'_{(i,x)}$, $\tilde{v}'_{(i,y)}$, and $\tilde{v}'_{(i,z)}$ in the final encrypted model; (c) is the distribution of integer coordinate values $\tilde{v}''_{(i,x)}$, $\tilde{v}''_{(i,y)}$, and $\tilde{v}''_{(i,z)}$ in the marked model.

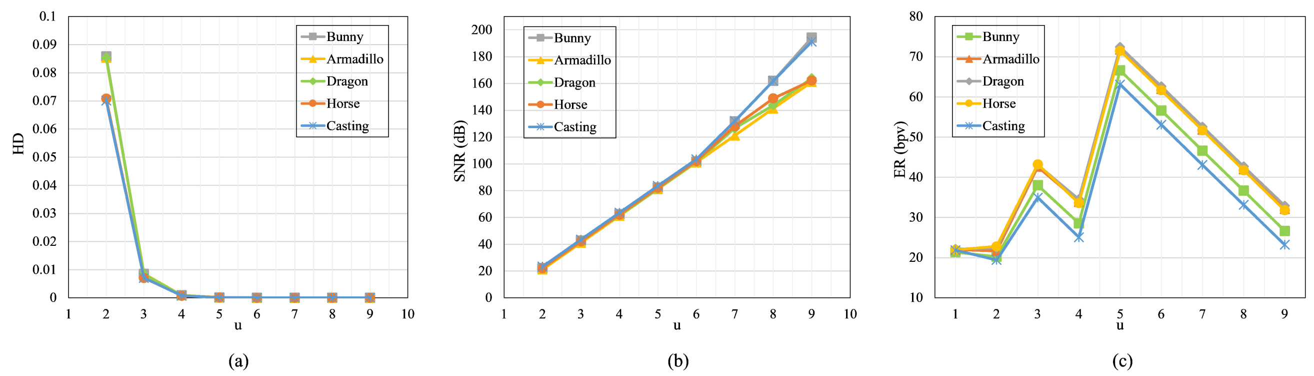


Fig. 9. The results using different u of the five test models: (a) HD; (b) SNR; (c) ER.

TABLE II
THE COMPARISON RESULTS UNDER DIFFERENT STATISTICAL ANALYSIS COEFFICIENTS USING THE TEST MODELS.

Test models		SE↑ (0 ∼ 9)	NPCR↑ (0% ∼ 100%)	HD↑ (0 ∼ +∞)	SNR↓ (−∞ ∼ +∞)
Bunny	M	7.4914	-	0	+∞
	M_f	8.9899	100%	72627.46	-116.42
	M_m	8.9887	100%	73071.66	-116.39
Armadillo	M	7.0897	-	0	+∞
	M_f	8.9981	100%	73859121	-117.55
	M_m	8.9978	100%	73985040	-117.53
Dragon	M	7.6239	-	0	+∞
	M_f	8.9992	100%	73859.34	-116.06
	M_m	8.9991	100%	73684.25	-116.04
Horse	M	7.1454	-	0	+∞
	M_f	8.9970	100%	738589.40	-118.36
	M_m	8.9965	100%	733611.83	-118.35
Casting	M	7.7011	-	0	+∞
	M_f	8.9225	100%	719006.99	-116.93
	M_m	8.9237	100%	728376.77	-116.96

↑ denotes higher value is better, and vice versa.

the distributions of the integer coordinate values $\tilde{v}_{(i,x)}$, $\tilde{v}_{(i,y)}$, and $\tilde{v}_{(i,z)}$ in the original model, revealing uneven distributions along the x-, y-, and z-axis. Hence, the original model M lacks sufficient undetectability and its content is easily perceptible, making it unsuitable for transmission over public channels. In contrast, the histograms in Figs. 8(b)(c) exhibit more uniform distributions, therefore the final encrypted model M_f and the marked model M_m produced by the proposed method both possess good imperceptibility, making them suitable for transmission without the risk of content leakage.

We further evaluate the visual security level of the models through statistical analysis coefficients, e.g., Shannon entropy (SE), the number of changing vertex rate (NCVR), HD, and SNR. In calculating SE, the number of discretization intervals is fixed at 8, resulting in a maximum entropy value of 9. NCVR represents the proportion of vertex coordinates in the target model that have changed compared to the original model. Table II presents a comparison of the original model M, the final encrypted model M_f, and the marked model M_m across different statistical coefficients, each with corresponding boundary values. The results show that, compared to the model M, the SE of models M_f and M_m is very close to the maximum value of 9, highlighting that the vertex distribution in these generated models is highly uniform and random, without the discernible patterns or regularities present in the original model. Each NPCR of 100% further underscores that the vertex coordinate distributions in the final encrypted model M_f and marked model M_m are completely distinct from those in the original model. Additionally, the high HD and negative SNR values reinforce that the two models generated by the proposed method differ significantly from the original model, making it challenging to infer M from M_f or M_m. The table also reveals that models M_f and M_m have similar values across the same metrics, as model M_m is generated by embedding encrypted data into model M_f, where the original ciphertext is replaced with new ciphertext. These statistical analyses effectively demonstrate that the security of the models is not compromised by data embedding, and both models M_f and M_m exhibit excellent visual security.

B. Separability and Reversibility Analysis

The bottom of Fig. 7 illustrates the restored models corresponding to the five tested models, which, to the naked eye, appear indistinguishable from the original models. A receiver possessing only the encryption key K_e can decrypt the marked models into the restored models. If the receiver possesses both the data hiding key K_d and the encryption key K_e, he or she can retrieve the original embedded data from the marked models and also obtain the restored models at the bottom of Fig. 7. The restored models in both cases are identical, which proves the separability of the proposed algorithm. Furthermore, in order to objectively evaluate the discrepancies between the reconstructed models and the original models during the restoration phase, we employed the SNR and HD as metrics under varying values of u. SNR and HD are computed based on comparisons between the original and reconstructed models. An SNR approaching $+\infty$ suggests a high degree of similarity between the compared models, whereas a HD approaching 0 conveys a similar implication. Figs. 9(a)(b) present the variations in SNR and HD for five test models as the value of u increases from 1 to 9. The graphs reveal that, with the increment of u, the SNR for all

TABLE III
THE ER OF THE FIVE TEST MODELS WHEN $u = 5$ AND $n = 1 \sim 20$.

Model name	Threshold n																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Bunny	66.543	66.661	66.630	66.629	**66.662**	66.661	66.659	66.657	66.655	66.654	66.653	66.653	66.651	66.651	66.647	66.592	66.594	66.592	66.592	66.590
Armadillo	71.833	71.832	71.832	**71.846**	**71.846**	**71.846**	71.845	71.845	71.845	71.844	71.844	71.845	71.844	71.839	71.838	71.838	71.838	71.837	71.837	71.836
Dragon	72.283	72.323	72.352	**72.417**	72.416	72.415	72.415	72.414	72.414	72.412	72.411	72.410	72.410	72.409	72.409	72.408	72.407	72.407	72.406	72.405
Horse	71.443	**71.535**	71.533	71.533	71.528	71.529	71.527	71.526	71.526	71.525	71.525	71.523	71.413	71.412	71.411	71.411	71.410	71.410	71.410	71.408
Casting	**63.179**	63.177	63.120	63.118	63.109	63.106	63.093	63.084	63.072	63.062	63.049	63.036	63.026	63.016	62.996	62.987	62.981	62.968	62.956	62.948

TABLE IV
EMBEDDING CAPACITY AND AUXILIARY INFORMATION LENGTH FOR FIVE TEST MODELS $(u = 5, n = 4)$.

Model name	Number of		Total embedding	Auxiliary information	ER (bpv)	Increased capacity (bit) using	
	Vertices	Faces	capacity (bit)	length (bit)		Virtual connection	Entropy encoding
Bunny	35947	69451	2507451	112355	66.63	+43208	+85168
Armadillo	172974	345944	12887016	459573	71.85	-	+405272
Dragon	437645	871414	32948688	1255830	72.42	-	+931625
Horse	112642	225280	8354049	296148	71.53	-	+267042
Casting	5096	10224	335199	13552	63.12	-	+11908

five test models progressively ascends towards infinity and the HD correspondingly descends towards zero. This indicates that when the value of u is sufficiently large to encompass all significant portions of the model's vertex coordinate values, the SNR will reach infinity and the HD will approach zero, allowing for a complete and lossless reversible restoration of the 3D model in our method.

C. Embedding Capacity Analysis

The data embedding capacity of the model is defined as the total embeddable capacity minus the bit length of the auxiliary information and subsequently represented by the embedding rate (ER) in bits per vertex (bpv). As the parameter u varies from 1 to 9, we recorded the average ER of different test models with the threshold n set from 1 to 10, the results are presented in Fig. 9(c). The patterns depicted in the figure indicate that when u changes from 2 to 3 or from 4 to 5, there is an increase in the model's ER. This is due to the length l of the binary coordinate values l changes from 8 to 16 and then to 32, leading to an increase in the embeddable bit length for each vertex. Conversely, when u increases from 1 to 2, from 3 to 4, or from 5 to 9, the ER of the model gradually decreases. This is attributed to the fact that when the length l remains fixed, an increasing number of bits are utilized to represent the significant part of the coordinate values and to be predicted.

Table III presents the ER of five test models under the parameter setting of $u = 5$ and threshold value n ranging from 1 to 20. The highlighted values indicate the highest ER for the respective models. Statistical analysis reveals that for the majority of models, an initial threshold value of n set between 1 to 6 can maximize the model's embedding capacity. Then as the value of n increases, the number of vertices available for data embedding decreases, resulting in a diminishing trend in the ER.

Table IV shows the fundamental data regarding the embedding capacity and the length of auxiliary information for five test models with $u = 5$ and $n = 4$. When the total embedding capacity is fixed, a shorter length of auxiliary information results in a higher data embedding capacity for the model. The entropy coding employed in the proposed method capitalizes on this principle. Thus, the table includes the capacity enhancement using entropy encoding as opposed to conventional binary encoding, demonstrating the efficacy of arithmetic or Huffman encoding. Additionally, when the length of auxiliary information is fixed, a greater total embeddable capacity translates to a higher ER. The virtual connections utilized in our method are based on this principle. Table IV enumerates the increased capacity achieved by models with independent vertices using virtual connections compared to those without, effectively validating the advantageous impact of our proposed method. Even though the proportion of models with independent vertices is relatively small, the proposed virtual connection technique effectively utilizes these often overlooked vertices and incorporates them into the dynamic prediction process, ensuring a vertex utilization rate close to 100%. Additionally, with the rapid development of 3D technology, more complex and high-precision 3D mesh models with independent vertices will inevitably be used across various industries. Virtual connections ensure that the embedding performance of such models remains unaffected.

D. Performance Comparison

The performance of the proposed method was compared with several state-of-the-art works [27, 29–34], with the experiment setup uniformly setting the parameter u to 5. In [27], Tsai *et al.* introduced a technique involving spatial partitioning and space coding for data embedding, where the achievable ER depends on the number of subdivided blocks $T_P \times T_P \times T_P$, reaching up to 7.68 bpv. Unlike [27], Yin *et al.* [29] relied on the connection between embeddable vertices and their surrounding vertices through multi-MSB to vacate redundant space, requiring different embeddable lengths to achieve the maximum ER for various models. In [30], half of the vertices in the model were used to predict the other

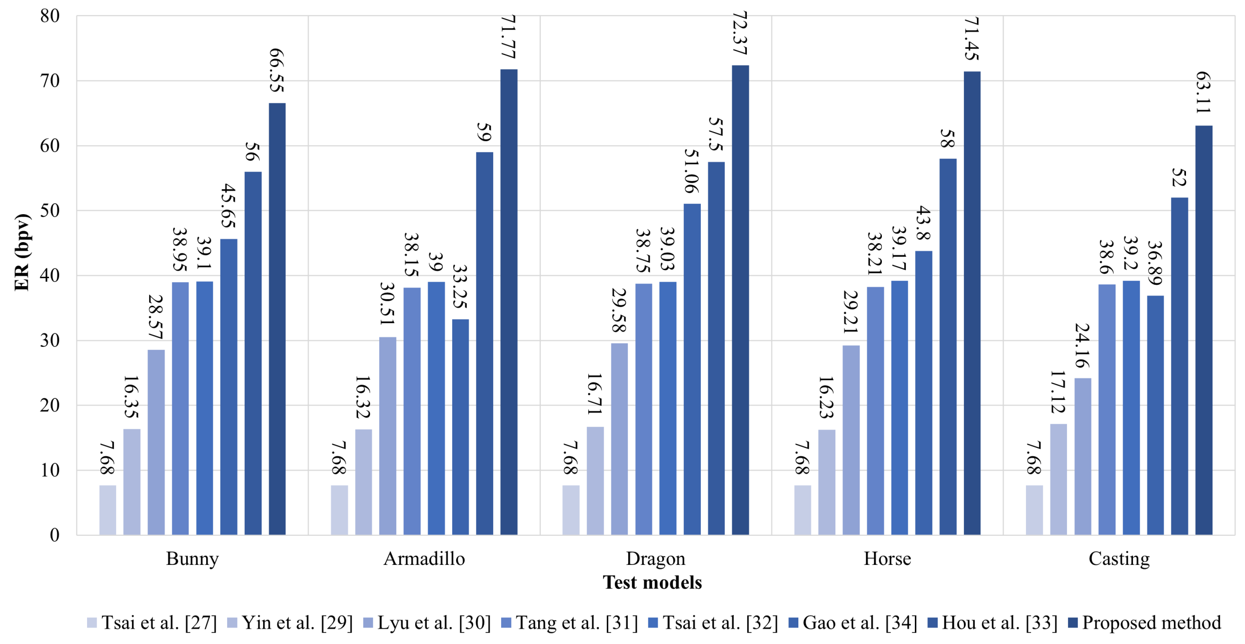


Fig. 10. Comparison of ER (bpv) among several compared methods for five test models.

half through multi-MSB prediction, and the embeddable length for each vertex could be directly obtained. Building on [30], Tang *et al.* [31] transformed some vertices from predictors to data embedders. In [32], Tsai *et al.* employed a selection threshold T_s and a selection key K_s to randomly select a proportion T_s of vertices connected to the embeddable vertex for prediction. Typically, when T_s is set to 0%, meaning one embeddable vertex corresponds to one predictor, the model's ER can reach its maximum. Hou *et al.* [33] utilized an octree to partition the model into multiple sub-blocks, with relevant parameters set to $MaxD = 6$, $MaxS = 40$, and $T_D = 3$. The model encryption in [34] uses secret sharing. In the current method, the threshold n typically ranges from 1 to 6 to maximize the model's ER. The maximum ER achieved by the aforementioned advanced methods and the current method on five test models were compared and displayed in Fig. 10 using bar charts. It is evident from the figure that the proposed method surpasses the others in ER across all test models, particularly for models with a larger number of vertices, such as Armadillo, Dragon, and Horse, where the proposed method exceeds the best-performing scheme by more than 12 bpv. To demonstrate the proposed method's capability of high-capacity data embedding across all models, the PSB database were employed to calculate and compare the average ER of the current and other methods, as illustrated in Fig. 11. The results indicate that the proposed method consistently outperforms the others, with the average ER of the models nearly 14 bpv higher than the most advanced method. This signifies that our method significantly enhances the data embedding capacity of the models compared with other methods, and the achieved effects are applicable to all models.

Table V presents the features comparison between the

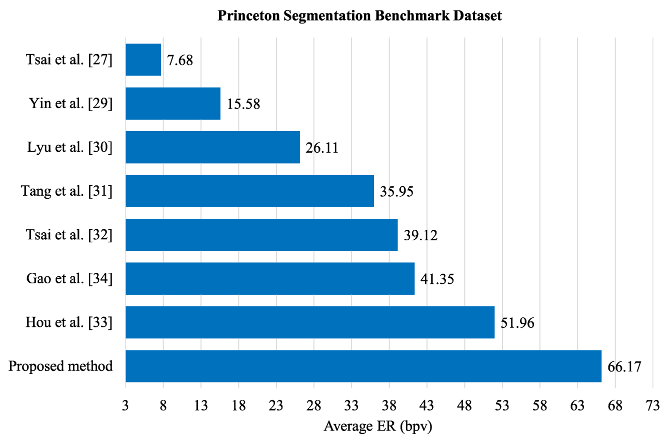


Fig. 11. Comparison of average ER (bpv) among several compared methods on PSB dataset.

proposed method and state-of-the-art methods. The utilization rate of vertices indicates the proportion of embeddable vertices to the total number of vertices, while the ER is calculated based on models from the PSB database. In [27], although nearly all vertices are available for data embedding, the ER still depends on the number of partitioned sub-blocks T_P. However, incorrect T_P can lead to errors in data extraction. Yin *et al.*'s method [29] constructs the embeddable room using multi-MSB between vertices, which improves the ER, but the vertex utilization rate remains low. Lyu *et al.* [30] improved upon Yin *et al.*'s method [29] by allocating odd- or even-numbered vertices as embeddable, thus achieving a vertex utilization rate of 50%. In [31], Tang *et al.* further refined the vertex partitioning mechanism in Lyu *et al.*'s method

TABLE V
FEATURES COMPARISON BETWEEN THE PROPOSED METHOD AND OTHER METHODS.

Method	Vertex utilization rate	Embedding rate	Independent vertices embedding	Computational complexity	Separability	Extraction errors
Tsai *et al.* [27]	≈100%	$\log_2(T_P \times T_P \times T_P)$	No	Low	Yes	Yes
Yin *et al.* [29]	27.48%	1.78 ∼ 12.48	No	Low	Yes	No
Lyu *et al.* [30]	50%	17.84 ∼ 25.65	No	Low	Yes	No
Tang *et al.* [31]	73%	22.53 ∼ 38.95	No	Low	Yes	No
Tsai *et al.* [32]	27.48% ∼ 62.74%	13.49 ∼ 27.36	No	Low	Yes	No
Hou *et al.* [33]	95.49%	16.56 ∼ 55.02	No	High	Yes	No
Gao *et al.* [34]	69.93%	33.55 ∼ 51.61	No	Low	Yes	No
Proposed method	≈100%	62.21 ∼ 70.69	Yes	Low	Yes	No

[30], allowing some vertices initially used for prediction to embed data, thereby enhancing the ER. Tsai *et al.*'s method [32] employs thresholds to randomly select prediction vertices around embeddable vertices, with both the vertex utilization rate and ER varying as the thresholds change. In [33], Hou *et al.* utilized the octree and multiple thresholds to divide the model into multiple embeddable sub-blocks, with only one vertex per sub-block used for prediction, resulting in a maximum vertex utilization rate of approximately 95.49%. However, this method performs poorly in models with fewer vertices, with a significant difference between the lowest and highest ERs, and the method has a high computational complexity. Gao *et al.*'s method [34] proposed a grouping rule that allowed the vertex utilization to reach nearly 70%, the ER of the model is not high, and significant disparities can still occur. In stark contrast to the aforementioned works, the proposed method in this paper uses dynamic prediction and virtual connection to allow all vertices, apart from n initial vertices, to be used for both data embedding and prediction. As the threshold n is often a single-digit number when the method performs optimally, the vertex utilization rate can approach nearly 100%, ensuring a high ER within every model. Moreover, our method is unique in its ability to embed data using independent vertices. The algorithmic procedures within this method are straightforward and separable, allowing for the complete and accurate extraction of data.

V. CONCLUSION

In this work, we proposed a high capacity RDH method for encrypted 3D mesh models based on dynamic prediction and virtual connection. The dynamic prediction mechanism allows vertices to embed data while also serving predictive functions. Subsequently, virtual connection enables the further utilization of independent vertices within the model. The synergy of two ways nearly maximizes the vertex utilization rate. Experimental results demonstrated that the proposed method achieves a higher embedding capacity on 3D models compared to the existing state-of-the-art methods while ensuring reversibility.

In the future, we will explore novel methods aimed at enhancing the precision of predictions between vertices and reducing the length of auxiliary information, thereby further improving the data embedding capacity of 3D models. Additionally, we will focus on the security aspect of the generated models within complex environments, investigating their potential to withstand various forms of attacks.

REFERENCES

[1] J. Wang, X. Chen, J. Ni, N. Mao, and Y. Shi, "Multiple histograms-based reversible data hiding: Framework and realization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 8, pp. 2313–2328, 2020.

[2] G. Coatrieux, C. Le Guillou, J.-M. Cauvin, and C. Roux, "Reversible watermarking for knowledge digest embedding and reliability control in medical images," *IEEE Trans. Inf. Technol. Biomed.*, vol. 13, no. 2, pp. 158–165, 2008.

[3] S. Lee, C. D. Yoo, and T. Kalker, "Reversible image watermarking based on integer-to-integer wavelet transform," *IEEE Trans. Inf. Forensic Secur.*, vol. 2, no. 3, pp. 321–330, 2007.

[4] Z. Ni, Y. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354–362, 2006.

[5] G. Coatrieux, W. Pan, N. Cuppens-Boulahia, F. Cuppens, and C. Roux, "Reversible watermarking based on invariant image classification and dynamic histogram shifting," *IEEE Trans. Inf. Forensics Secur.*, vol. 8, no. 1, pp. 111–120, 2013.

[6] M. U. Celik, G. Sharma, and A. M. Tekalp, "Lossless watermarking for image authentication: a new framework and an implementation," *IEEE Trans. Image Process.*, vol. 15, no. 4, pp. 1042–1049, 2006.

[7] W. Zhang, X. Hu, X. Li, and N. Yu, "Recursive histogram modification: establishing equivalency between reversible data hiding and lossless data compression," *IEEE Trans. Image Process.*, vol. 22, no. 7, pp. 2775–2785, 2013.

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

[9] Q. Chang, X. Li, and Y. Zhao, "Reversible data hiding for color images based on adaptive three-dimensional histogram modification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 9, pp. 5725–5735, 2022.

[10] Q. Chang, X. Li, Y. Zhao, and R. Ni, "Adaptive pairwise prediction-error expansion and multiple histograms modification for reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 12, pp. 4850–4863, 2021.

[11] T. Zhang, T. Hou, S. Weng, F. Zou, H. Zhang, and C.-C. Chang, "Adaptive reversible data hiding with contrast enhancement based on multi-histogram modification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 8, pp. 5041–5054, 2022.

[12] S. Weng, Y. Zhou, T. Zhang, M. Xiao, and Y. Zhao, "General framework to reversible data hiding for JPEG images with multiple two-dimensional histograms," *IEEE Trans. Multim.*, pp. 1–16, 2022.

[13] S. Weng, Y. Zhou, T. Zhang, M. Xiao, and Y. Zhao, "Reversible data hiding for JPEG images with adaptive multiple two-dimensional histogram and mapping generation," *IEEE Trans. Multim.*, pp. 1–15, 2023.

[14] S. Yi and Y. Zhou, "Separable and reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Trans. Multim.*, vol. 21, no. 1, pp. 51–64, 2019.

[15] Y. Wu, Y. Xiang, Y. Guo, J. Tang, and Z. Yin, "An improved reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Trans. Multim.*, vol. 22, no. 8, pp. 1929–1938, 2020.

[16] Z. Yin, Y. Xiang, and X. Zhang, “Reversible data hiding in encrypted images based on multi-MSB prediction and huffman coding,” *IEEE Trans. Multim.*, vol. 22, no. 4, pp. 874–884, 2020.
[17] C. Yu, X. Zhang, X. Zhang, G. Li, and Z. Tang, “Reversible data hiding with hierarchical embedding for encrypted images,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 2, pp. 451–466, 2022.
[18] S. Xu, J.-H. Horng, C.-C. Chang, and C.-C. Chang, “Reversible data hiding with hierarchical block variable length coding for cloud security,” *IEEE Trans. Dependable Secur. Comput.*, pp. 1–14, 2022.
[19] Y. Yao, K. Wang, Q. Chang, and S. Weng, “Reversible data hiding in encrypted images using global compression of zero-valued high bit-planes and block rearrangement,” *IEEE Trans. Multim.*, 2023.
[20] X. Zhang, “Separable reversible data hiding in encrypted image,” *IEEE Trans. Inf. Forensic Secur.*, vol. 7, no. 2, pp. 826–832, 2011.
[21] X. Liao and C. Shu, “Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels,” *J. Vis. Commun. Image Represent.*, vol. 28, pp. 21–27, 2015.
[22] B. Chen, W. Lu, J. Huang, J. Weng, and Y. Zhou, “Secret sharing based reversible data hiding in encrypted images with multiple data-hiders,” *IEEE Trans. Dependable Secur. Comput.*, vol. 19, no. 2, pp. 978–991, 2022.
[23] C. Yu, X. Zhang, G. Li, S. Zhan, and Z. Tang, “Reversible data hiding with adaptive difference recovery for encrypted images,” *Inf. Sci.*, vol. 584, pp. 89–110, 2022.
[24] R. Jiang, H. Zhou, W. Zhang, and N. Yu, “Reversible data hiding in encrypted three-dimensional mesh models,” *IEEE Trans. Multim.*, vol. 20, no. 1, pp. 55–67, 2017.
[25] M. Shah, W. Zhang, H. Hu, H. Zhou, and T. Mahmood, “Homomorphic encryption-based reversible data hiding for 3D mesh models,” *Arabian Journal for Science and Engineering*, vol. 43, pp. 8145–8157, 2018.
[26] B. J. Van Rensburg, P. Puteaux, W. Puech, and J.-P. Pedeboy, “Homomorphic two tier reversible data hiding in encrypted 3D objects,” in *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 3068–3072, IEEE, 2021.
[27] Y.-Y. Tsai, “Separable reversible data hiding for encrypted three-dimensional models based on spatial subdivision and space encoding,” *IEEE Trans. Multim.*, vol. 23, pp. 2286–2296, 2020.
[28] N. Xu, J. Tang, B. Luo, and Z. Yin, “Separable reversible data hiding based on integer mapping and MSB prediction for encrypted 3D mesh models,” *Cogn. Comput.*, vol. 14, no. 3, pp. 1172–1181, 2022.
[29] Z. Yin, N. Xu, F. Wang, L. Cheng, and B. Luo, “Separable reversible data hiding based on integer mapping and multi-MSB prediction for encrypted 3D mesh models,” in *Chinese Conference on Pattern Recognition and Computer Vision (PRCV).*, pp. 336–348, Springer, 2021.
[30] W.-L. Lyu, L. Cheng, and Z. Yin, “High-capacity reversible data hiding in encrypted 3D mesh models based on multi-MSB prediction,” *Signal Process.*, vol. 201, p. 108686, 2022.
[31] Y. Tang, L. Cheng, W. Lyu, and Z. Yin, “High capacity reversible data hiding for encrypted 3D mesh models based on topology,” in *International Workshop on Digital Watermarking*, pp. 205–218, Springer, 2022.
[32] Y.-Y. Tsai and H.-L. Liu, “Integrating coordinate transformation and random sampling into high-capacity reversible data hiding in encrypted polygonal models,” *IEEE Trans. Dependable Secur. Comput.*, 2023.
[33] G. Hou, B. Ou, M. Long, and F. Peng, “Separable reversible data hiding for encrypted 3D mesh models based on octree subdivision and multi-MSB prediction,” *IEEE Trans. Multim.*, 2023.
[34] K. Gao, J.-H. Horng, and C.-C. Chang, “Reversible data hiding for encrypted 3D mesh models with secret sharing over galois field,” *IEEE Transactions on Multimedia*, vol. 26, pp. 5499–5510, 2024.
[35] X. Wu, J. Zhang, Q. Yao, and W. Lyu, “Reversible data hiding based on octree partitioning and arithmetic coding in encrypted three-dimensional mesh models,” in *International Conference on Intelligent Computing*, pp. 49–60, Springer, 2024.
[36] X. Chen, A. Golovinskiy, and T. A. Funkhouser, “A benchmark for 3d mesh segmentation,” *ACM Trans. Graph.*, vol. 28, no. 3, p. 73, 2009.

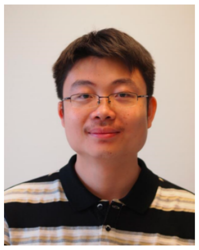
Ke Wang received the B.S. degree from Southwest Minzu University, Chengdu, Sichuan, China, in 2022. He is currently pursuing the master’s degree with the School of Cyberspace, Hangzhou Dianzi University. His current research interests include multimedia forensics and information hiding.

Ye Yao received the M.S. degree in computer science and the Ph.D. degree in communication and information systems from Wuhan University, Wuhan, China, in 2005 and 2008, respectively. He was a Visiting Scholar with New Jersey Institute of Technology, Newark, NJ, USA, from December 2016 to December 2017. He is currently an Associate Professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou. His research interests include multimedia forensics and information security.

Yanzhao Shen is currently a researcher at Shandong Institute of Blockchain. In 2012 he received the Master degree in Computer Science and Technology at Donghua University and in 2018 he obtained the Ph.D degree from School of Mathematics at Shandong University. His research interests involve security and privacy, cryptography, blockchain, and internet of things.

Fengjun Xiao received the M.S. degree from the Institute for Advanced Studies in Humanities and Social Sciences, Beihang University, Beijing, China, in 2014, and the Ph.D. degree from the School of Public Administration, Beihang University, Beijing, China, in 2020. He is currently a researcher with the Zhejiang Informatization Development Institute, Hangzhou Dianzi University, Hangzhou. His research interests include information management, sustainable development, and research innovation.

Yizhi Ren received the Ph.D. degree in Computer software and theory from Dalian University of Technology, China in 2011. From 2008 to 2010, he was a research fellow at Kyushu University, Japan. He is currently a full professor with School of Cyberspace, Hangzhou Dianzi University, China. His research interests include data security, privacy preserving, and trust management. He has served as the local chairs/PC members of more than 20 International conferences, such as, ICCCN 2018, ICC 2018, and DSC 2019.



Weizhi Meng is a Full Professor in the School of Computing and Communications, Lancaster University, United Kingdom, and an adjunct faculty in the Department of Applied Mathematics and Computer Science, Technical University of Denmark, Denmark. He obtained his Ph.D. degree in Computer Science from the City University of Hong Kong. He was a recipient of the Hong Kong Institution of Engineers (HKIE) Outstanding Paper Award for Young Engineers/Researchers in both 2014 and 2017. He also received the IEEE ComSoc Best Young Researcher Award for Europe, Middle East, Africa Region (EMEA) in 2020 and the IEEE ComSoc Communications Information Security (CISTC) Early Career Award in 2023. His primary research interests are blockchain technology, cyber security and artificial intelligence in security including intrusion detection, blockchain applications, smartphone security, biometric authentication, and IoT security. He is senior member of IEEE.