Adaptive Data Hiding: A Hybrid Based High-capacity Approach for 3D Models

Shih-Chun Tu, Charlie Irawan Tan and Wen-Kai Tai
Department of Computer Science and Information Engineering
National Dong Hwa University, Taiwan, Republic of China.
tusjtu@ms01.dahan.edu.tw, d9421003@ems.ndhu.edu.tw, wktai@mail.ndhu.edu.tw

Abstract

A hybrid data hiding approach, combining permutation steganography and the spatial domain approach, is proposed in this paper. Message is partially embedded in the cover model by permutation steganography first, and then the rest of message is embedded in the vertex by modifying vertex position limited in a voxel. The number of bits to be embedded in the vertex is adaptive in light of the degree of model distortion and visual perception to the stego model preferred.

As compared with previous data hiding methods for 3D models, our capacity is up to twice as large as that of previous work. Also, with the adaptability, the proposed hybrid approach is flexible with a tradeoff between capacity and distortion. Both embedding and extraction procedures are simple to implement and running efficiently.

Keywords: data hiding, spatial domain, permutation steganography.

1. INTRODUCTION

Steganography, the art of hiding one message in another, has been used over many centuries. Steganography ranges from simple hidden messages that can be deciphered by shifting each letter by a number of positions in the alphabet to watermarks that can be extracted from an image by a specific method, and has been applied widely in many topics related to information security. Steganography has even been used recently in 3D data, with similar purposes to those of 2D data.

In this paper, we propose a hybrid approach for data hiding based on permutation steganography and the vertex modification in the spatial domain. The 3D model consists of the vertex set and face set. The permutation steganography by Tu et al. [1] is applied to embed message in the vertex and face. Namely, the vertex permutation and the face permutation represent the embedded message. Then, we modify the position of the vertex in the vertex permutation within a range inside a predefined voxel to further embed more bits in each component of the vertex. The axis aligned bounding volume composed of voxels is built for the cover model. The voxel is subdivided into a number of units in each dimension as well. The unit where a vertex is located at inside the voxel indicates the embedded message bitstream for each dimension. Eventually, one part of the message is embedded in the first embedding stage, and another in the second. When extracting the message, either the extraction procedure of permutation steganography [1], or modifying vertex position can be applied first. Message extracted by the procedure of modifying vertex position is the second part of embedding message, and, of course, by that of permutation steganography is the first part.

The permutation steganography is distortionless for the stego model, but modifying vertex position causes model distortion related to the number of bits to be embedded in and the scale of the cover model. Therefore, the capacity of our hybrid approach is adaptive by trading distortion for capacity. Model distortion is measured using normalized Hausdorff distance and visual perception. As experimental results show, our capacity is the highest than previous work of data hiding. Also, our approach is simple to implement with time complexity $O(n)$.

The rest of this paper is structured as follows. Section 2 surveys related work. Section 3 describes the proposed method. Experimental results are shown in Section 4. Finally, we conclude and point out possible future work in Section 5.

2. RELATED WORK

Steganography for 3D polygonal meshes was pioneered by Ohbuchi et al. [2], who introduced watermarking on 3D polygonal meshes. Since then many ideas to steganography have been proposed. Most methods are to slightly perturb the vertex positions of the mesh for hiding messages either in spatial domain [3, 4, 5, 6, 7, 8, 9, 2, 10, 11, 12, 13, 14, 15, 16], or in the spectral domain, [17, 18, 19, 20, 21, 22]. Spatial methods tend to have higher capacity and lower computation costs at the expense of weak robustness. Spectral methods are more robust but have limited capacity and involve serious computations. They are more appropriate for data protection applications, such as watermarking, than for data hiding. Recently, Chao et al. [23] presented a very high-capacity and low-distortion 3D steganography approach based on a novel multilayered embedding scheme to hide secret messages in the vertices of 3D polygonal models. Their approach can hide 21 to 39 bits/vertex.

Some methods, [17, 23, 20, 13], utilize 3D models defined as point set. Polygonal meshes provide fewer vertices than point set models, but have face information that can be used as the alternative medium. Higher vertex numbers allow a model to hide more information, but require more space and computing power to handle. Recently several methods, [6, 11], hide messages in the connectivity of the mesh by rearranging vertices and faces relative to a reference ordering derived from the mesh geometry. Their techniques are lossless because the cover and stego models are the same.

Permutation steganography [24, 25, 26, 27] gives optimal capacity for hiding information through reordering of $n$ primitives that have a known reference ordering. Permutation steganography is of the optimal capacity, up to $O(\log(n)) = O(n\log n)$ bits, which is much better than the results of the previous work for 3D polygonal meshes but at the expense of computation time $\Omega(n^2 \log^2 n \log \log n)$. Recently, two proposed methods, Bogomjakov et al. [28] and Tu et al. [1], are very simple to implement and perform efficiently, $O(n)$. Both methods guarantee the minimal capacity, one bit per element less than the theoretical optimum, and are robust and resistant to any kind of attacks on the polygonal mesh because the reference ordering is obtained by using the traversal of Edgebreaker mesh compression algorithm [29] based on the mesh connectivity alone. Obviously, those approaches are lossless.
The framework of our approach is shown in Figure 1. There are two stages both for message embedding and extraction procedures. When embedding messages for the cover model, we first apply permutation steganography [1], to hide message, and then modify the rearranged embedding vertices to hide more message. To extract message from the stego model, either the extraction procedure of [1] or modifying vertex position can be first used to extract message, and then another follows to extract message left.

### 3.1 Embedding Message

#### 3.1.1 Embedding by Permutation Steganography

Permutation steganography hides the message in a cover model by rearranging the order of vertices in the model with respect to a canonical reference ordering. We apply the method proposed by Tu et al. [1], which improves the work of Bogomjakov et al. [28], to hide the message in the first stage of the embedding procedure.

Given a cover model, \( M_c \equiv (V, F) \), where \( V \) is the set of vertices and \( F \) is the set of faces, the first edge of the first face in the model is selected as the initial vertex and edge to obtain two reference orderings by using Edgebreaker algorithm [29] respectively. Note that the binary trie [30] search structure is built for the embedding primitive, in which the internal node branches the search traversal by the message bitstream and the leaf node keeps the primitive each ordered by using Edgebreaker algorithm [29] respectively. Note that either stage can be used to extract message first, and then another.

After this first stage of embedding procedure, we have the quasi stego model \( M_{s*} = (V_{perm}, F_{perm}) \).

#### 3.1.2 Embedding by Modifying Vertex Position (MVP)

The vertex in the output primitive permutation \( V_{perm} \) can be embedded in more message bitstream by modifying its position. The axis aligned bounding volume of \( M_{s*} \) is determined. Given \( k_x, k_y, \) and \( k_z \) bits to be embedded in \( x, y, \) and \( z \) components of the vertex respectively, the volume is then subdivided into \( n_x^{voxel} \times n_y^{voxel} \times n_z^{voxel} \) voxels, where \( n_{voxel}^i = l_{BV}^i / (2^{k_i} \times l_v^i) \), \( l_{BV}^i \) is the side length of the bounding volume, and \( l_v^i \) is the unit length of \( 2^{k_i} \) units in the voxel for \( i = x, y, \) and \( z \). The next \( k_i \) bits in the embedding message are embedded in the vertex by modifying the coordinate of its \( i \) component to the unit with the index equal to the integer value of \( k_i \) in the component of the belonging voxel for all \( i = x, y, \) and \( z \), Figure 3 illustrates the voxelized bounding volume and the subdivision for a voxel. The embedding procedure of MVP is summarized in Algorithm 4.

At each step \( i \), a primitive at position \( p \) is chosen from the remaining \( n - i \) primitives of the reference ordering and output it as the next primitive of the permutation. The position \( p \) is the index of the leaf node reached by the binary trie traversal according to the next \( k + 1 \) bits, \( k = \lceil \log_2(n - i) \rceil \), in the embedding message. The binary trie is a complete binary tree so the leaf node is either at level \( \lceil \log_2(n - i) \rceil \) or \( \lceil \log_2(n - i) \rceil - 1 \). If the number of leaf nodes at the highest level, \( e = ((n - i) - 2^{k}) \times 2 \), is larger than the integer value of the next \( k + 1 \) bits, then the primitive at position \( p \) will be outputted as the next primitive of the permutation. Otherwise, the primitive at level \( \lceil \log_2(n - i) \rceil \) reached by next \( k \) bits in the embedding message will be outputted. Note that the output primitive is actually removed by replacing it with the last primitive in the remaining primitive so that the remaining \( n - i - 1 \) primitives can be still indexed sequentially.

### 3.2 Message Extraction

There are two stages of message extraction. First, the message extraction procedure of permutation steganography [1] is to extract the message. In the second stage, we extract the message which is embedded by MVP. Note that either stage can be used to extract message first, and then another.
4. EXPERIMENTAL RESULTS

All experiments were performed with several polygonal models of different sizes on a PC with an Intel Core 2 1.87GHz processor and 2GB main memory to verify and evaluate our proposed approach. In all experiments, unless otherwise specified, the decimal precision for the vertex coordinate of all testing models is about 6 decimal digits so the unit length $l_i$ is set to $1 \times 10^{-6}$, and $k_x = k_y = k_z = k$ bits.

Over 1000 randomly generated embedding messages are used to measure the average capacity and normalized Hausdorff distance of the testing models for permutation steganography [1], and MVP respectively. The statistics of the measured capacity is shown in Table 1. For permutation steganography, the average capacity of Bogomjakov et al. [28] is the highest, nearly optimum $log_2 n!$, than that of previous work. The average capacity improved by Tu et al. [1] is about 0.63 bits/vertex. For spatial domain, Cheng et al. [6] proposed a multilevel embedding procedure and a 3D model representation rearrangement procedure to hide 9 bits/vertex. Chao et al. [23] proposed a novel multilayered embedding scheme that can hide up to $3n_{layers}$ bits/vertex in normalized models, where $n_{layers}$ ranges from 7 to 13. The proposed MVP approach can hide 27 to 48 bits/vertex in 3D models. The capacity of MVP is much higher than that of Cheng et al. [6] in 3D models, but is less than that of Chao et al. [23] in normalized 3D models about 10 bits/vertex. However, as you can see, our hybrid data hiding approach produces much higher hiding capacity, $\approx 2 log_2 n!$, than all of the previous work.

The visual perception for the cover model and stego models of the all testing models were shown from Figure 7 to 12 in that subfigures (b) and (c) show the stego models with different $k$. Subfigure (b) shows that the stego model with the maximal $k$ bits embedded by using MVP has little distortion yet is almost unperceivable visually. As a model distortion comparison, subfigure (c) shows the stego model with distortion that can be easily detected when one likes to trade model distortion for embedding more bits. Subfigures (d), (e) and (f) show the close-up views of the subfigures right on the top. Regarding to the visual perception, subfigure (e) is similar to subfigure (d), but the distortion seen in subfigure (f) is manifest.

The number of bits to be embedded in the vertex at the second stage of message embedding is adaptive. But the more bits to be embedded in the vertex the more distortion to the stego model. The normalized Hausdorff distance [31] is commonly used to measure the model distortion such as Metro [32], M.E.S.H. [33], Cheng et al. [6], etc. We measure the average normalized Hausdorff distance (NHD) for all testing models and obtain the reasonable NHD for each of the testing model as shown in Table 1. As experiments show, the distortion is visually acceptable when the value of NHD is around $1 \times 10^{-4}$. Figure 5 shows the normalized Hausdorff distance as a function of embedding $k$ bits in each component of the vertex by MVP. Note that different model scale presents different sensitivity of model distortion to the increasing of embedding bits. The Armadillo is a large scale model and is of acceptable distortion when $k$ is up to 16 resulting totally high capacity 97.71 bpv. Figure 6 illustrates the adaptability using Armadillo model as an example. Our approach adapts the capacity by the model distortion preferred.
and permutation steganography [1], are both $O(n)$. Namely, the proposed hybrid approach is $O(n)$. Note that even for Armadillo model, the embedding and extraction procedures can be done in one second.

5. CONCLUSION AND FUTURE WORK

A hybrid data hiding approach has proposed, which combines permutation steganography and MVP. Embedding message by permutation steganography [1], for the vertex and face in the cover model and then modifying the vertex position by MVP, our method improves the capacity of data hiding on 3D models up to $2\log_2 n!$. Moreover, the capacity is adaptive regarding to the degree of model distortion making our method flexible. Our method is simple to implement and is efficient, $O(n)$, running in a second for all testing models.

The distortionless approach with high capacity is one of the key concerns for data hiding. In the future, it is worth exploring what optimal number of units in each component of the voxel is to make the model distortion less. Also, we like to embed more permutations in the cover model. Namely, the primitive arrangement represents more primitive permutations, hopefully, $\log_2 n! \geq 2$. MVP is not a robust approach. To improve the robustness, we would try to find a way for modifying vertex position on the basis of reference ordering.

6. REFERENCES


ABOUT THE AUTHOR
Shih-Chun Tu is currently a doctoral student of Computer Science and Information Engineering department at National Dong Hwa University, Taiwan, R.O.C.

Charlie Irawan Tan is currently a doctoral student of Computer Science and Information Engineering department at National Dong Hwa University, Taiwan, R.O.C.

Wen-Kai Tai is currently an associate professor of Computer Science and Information Engineering department at National Dong Hwa University, Taiwan, R.O.C. His current research interests include data hiding, real-time rendering, visibility, and techniques for gaming.
Figure 7: The visual perception for the cover model (a), Cow, its stego models for different value of $k$ ((b) and (c)), and three close-up views (bottom) at head regarding to the figure on the top.

Figure 8: The visual perception for the cover model (a), Fandisk, its stego models for different value of $k$ ((b) and (c)), and three close-up views (bottom) at middle part regarding to the figure on the top.
Figure 9: The visual perception for the cover model (a), Knots, its stego models for different value of $k$ ((b) and (c)), and three close-up views (bottom) at middle part regarding to the figure on the top.

Figure 10: The visual perception for the cover model (a), Horse, its stego models for different value of $k$ ((b) and (c)), and three close-up views (bottom) at head regarding to the figure on the top.
Figure 11: The visual perception for the cover model (a), Teeth, its stego models for different value of $k$ ((b) and (c)), and three close-up views (bottom) at middle part regarding to the figure on the top.

Figure 12: The visual perception for the cover model (a), Armadillo, its stego models for different value of $k$ ((b) and (c)), and three close-up views (bottom) at chest regarding to the figure on the top.