A Reversible Steganography Scheme for 3D Mesh Models

I-Ling Chung[1,2], Chang-Min Chou[2] and Din-Chang Tseng[1]
[1] Institute of Computer Science and Information Engineering, National Central University,
No. 300, Jungda Rd., Jhongli City, Taoyuan, Taiwan 320.
E-mail: tsengdc@ip.csie.ncu.edu.tw Tel: +886-3-4227151- 57865
[2] Department of Electronic Engineering, Ching Yun University,
No. 229 Cheng-Shing Rd., Jhongli City, Taoyuan, Taiwan 320.
E-mail: ling@cyu.edu.tw Tel: +886-3-4581196-5127
E-mail: changmin@cyu.edu.tw Tel: +886-3-4581196-5108

Abstract—In this paper, we propose a reversible steganography scheme for 3D mesh models. The scheme first find a subset of vertices for data embedding, followed by encoding the prediction residues of these vertices and the secret bit string by arithmetic coding method. The mantissa part of these vertices is then substituted by the coded arithmetic real numbers. This scheme embeds the secret information into multiple layers such that it is high confidential and capacity flexible. In this scheme, the maximum distortion on the stego model is controlled by the setting of a real value thus it can be controlled to be imperceptible. Experimental results show the efficiency of this scheme.

Key words: steganography, data-hiding, watermarking, 3D authentication.

I. INTRODUCTION

Steganography, or known as data hiding, is the art of hiding a secret message (payload) in another (cover media), and converting the cover media into stego media without attracting attention of any malicious third party. Data hiding algorithms try to efficiently maximize the size of payload while introducing as little distortion as possible. Many data hiding techniques have been developed on images [1, 2], audios and videos. Relatively fewer researchers work on 3D models. However, with the growth of 3D models created and transmitted in the Internet, data hiding on 3D models receives an increasing interest.

Most existing 3D data hiding schemes introduce irreversible changes to the cover model. For some artistic or technical models, it is sometimes very important not to make any modification on the original mesh models. Hence there is a need to develop reversible steganography schemes which have the ability of recovering the original model in the data extraction phase. However, there has been little attention paid to the reversible data hiding techniques for 3D models. Our main goal here is to present a reversible steganography scheme for 3D mesh models.

II. RELATED WORKS

Data hiding and watermarking techniques for still images have been widely studied and investigated in recent years. On the other hand, data hiding and watermarking for 3D models get relatively less notice. Initially, Ohbuchi et al. [3, 4, 5] proposed a large variety of techniques for embedding data in 3D polygonal models. Benedens and Busch [6, 7] embedded private watermarks by altering 3D object normal distribution. Their watermarking systems achieved robustness against randomization of vertices, mesh altering (re-meshing), and polygon simplification operations. Praun et al. [8] proposed a sophisticated robust mesh watermarking scheme to resist common mesh attacks such as translation, rotation, scaling, cropping, smoothing, simplification, and re-sampling operations. These above researches all concentrated on the 3D watermarking techniques.

In the field of 3D data hiding, Cayre and Macq [9] proposed a 3D data hiding scheme based on a substitution procedure in the spatial domain. The key idea is to consider a triangle as a two-state geometrical object, depending on what bit value is to be hidden. Their scheme is robust against translation, rotation, and scaling operations. Wang and Cheng [10] presented a multilevel embedding procedure for expanding the hiding capacity. They propose three embedding levels called sliding, extending, and rotating to embed data based on slightly shifting the vertex position. This method can provide about three times the capacity of that in [9]. Recently, Chao et al [11] presented a very high-capacity and low-distortion 3D steganography scheme. Their steganography approach is based on a multilayered embedding scheme to hide secret messages in the vertices of 3D polygon models. By their scheme, the distortion is very small on the stego model as the number of hiding layers ranges from 7 to 13 layers. The above 3D steganography schemes are not reversible.

III. THE PROPOSED METHOD

Given a 3D mesh model, our goal is to hide information in this model by slightly modifying the model and make it possible for the receiver to retrieve the hidden information and to rebuild the original model. For a given mesh model M (V, C), where V is the vertex set and C is the connectivity relationship on M, we embed a set of secret bit-string S = (0100101…) by inducing a small displacement on a subset of

V. A vertex *v* is noted as $v(x_1, x_2, x_3)$ and $v'(x_1', x_2', x_3')$ before and after embedding, respectively. The basic embedding idea is stated as follows:

i. Find a subset of *V* for embedding. Let *E* represents this subset.

ii. Calculate the prediction residues of vertices in *E*. Encode the prediction residues by arithmetic coding scheme. The number of the encoded arithmetic values should be less than the number of vertices in *E*.

iii. Encode the bit-string *S* by arithmetic coding scheme.

iv. For vertices in *E*, substitute a certain part of the mantissa of the original vertex coordinates by the encoded arithmetic values. Part of the vertices are substituted by the prediction residue encoded values, the others are substituted by the bit-string encoded values.

v. Repeat steps *i* to *iv* several times for multi-layer embedding. Output the embedded mesh model.

Figure 1 illustrates the idea of the embedding steps. In the following, we describe the single layer embedding and decoding scheme in detail. The multi-layer embedding and decoding scheme can be achieved by repeating the single layer procedures.

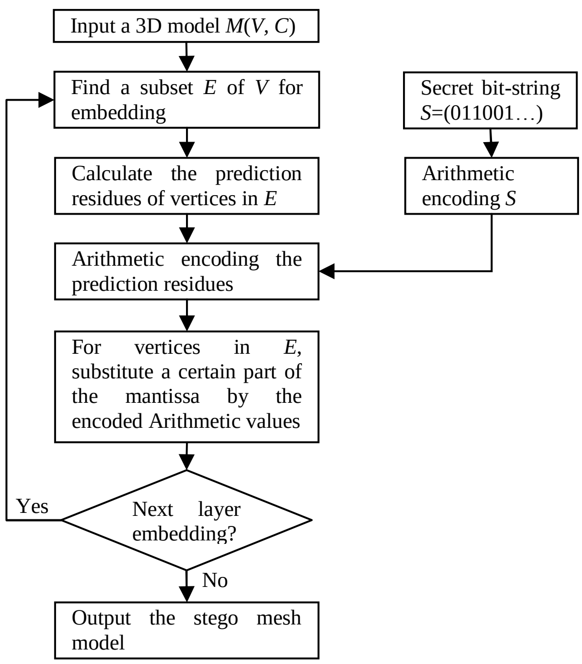


Fig. 1 Overview of the embedding scheme

A. *Finding a subset of vertices for embedding*

For a model *M*, we first set all its vertices as *fixed* vertices, then we traverse the whole model vertices starting from a predefined vertex v_s, for example, vertex v_1. The number *s* is saved as part of the key for decoding. A vertex surrounded by all *fixed* vertices is added into *E* and set to be a *non-fixed* vertex. Vertices in *E* are set for information embedding after a whole vertex traverse is completed.

B. *Prediction residues and bit-string encoding*

For each vertex in *E*, we first calculate its prediction coordinates v^p by averaging the coordinates of its neighboring vertices,

$$v^p = \frac{1}{|N(v)|} \sum_{v_j \in N(v)} v_j \tag{1}$$

where $N(v)$ is the set of *v*'s neighboring vertices and $|N(v)|$ is the size of $N(v)$. The prediction residues can be calculated by

$$pr = v - v^p \tag{2}$$

Figure 2 illustrates the concept of prediction residue.

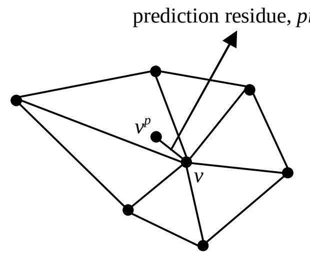


Fig. 2 The prediction residue *v* is the vertex for embedding v^p is predicted by *v*'s surrounding vertices.

Traditional arithmetic coding scheme [12] encodes datum into a series of real numbers between 0 and 1. In the proposed method, we encode the prediction residues into a series of real numbers between 0 and *Q*, where *Q* is the predefined maximum displacement on each coordinate. This modification can be achieved by changing the initial coding range from [0, 1) to [0, *Q*). This modification ensures that all coded real numbers are less than the maximum distortion *Q*. Another important issue is the precision level of the coded real numbers. We have to constrain the coded real numbers within a predefined precision level. Both the prediction residues and the bit-string are encoded by the arithmetic coding scheme to produce two sets of real numbers, *P* and *B*, respectively, for embedding.

C. *Mantissa substitution*

For each vertex $v(x_1, x_2, x_3)$ in *E*, we substitute part of its mantissa by the arithmetic coded real numbers,

$$x_i' = \begin{cases} x_i - (x_i \bmod Q) + a_j, & \text{if } x_i \ge 0; \\ x_i - (x_i \bmod Q) - a_j, & \text{otherwise.} \end{cases} \quad i = 1, 2, 3.\ a_j \in P, B \tag{3}$$

The vertices in *E* should provide enough space for embedding all real numbers in *P*, that is, $3|E| > |P|$. $|E|$ times by 3 since there are 3 coordinates in each vertex. The substitution procedure first substitutes all real numbers in *P*, and then real numbers in *B* for the rest vertices in $|E|$. For the single layer embedding, the mesh model M' now can be output as a secret model. Figure 3 illustrates the concept of mantissa substitution scheme.

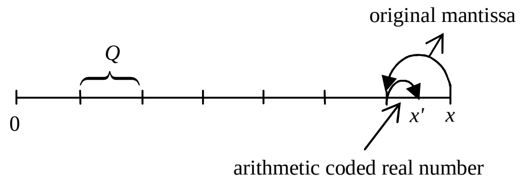


Fig. 3 The mantissa substitution scheme

D. Decoding and the reversibility

For a secret model M', we first traverse the whole model vertices starting from the predefined vertex v_s to get the embedding vertex set *E*. The rule is the same as described in the embedding side. For each vertex $v'(x_1', x_2', x_3')$ in *E*, we can extract the embedded real numbers,

$$a_j = \begin{cases} x_i' - (x_i' \bmod Q), & if\ x_i' \ge 0; \\ (x_i' \bmod Q) - x_i', & otherwise. \end{cases} \quad i = 1, 2, 3. \tag{4}$$

where a_j represents the elements of the decoded arithmetic real number sets, *P* and *B*. Then we can decode the prediction residues, *pr*, and the secret bit-string, *S*, from *P* and *B*, respectively. The original coordinates of vertices in *E* can be calculated by first applying (1) to get the predicted coordinates v^p, followed by adding the prediction residues back,

$$v = v^p + pr \tag{5}$$

Figure 4 provides an overview of the decoding scheme.

E. Multi-layer embedding

Since the proposed embedding scheme is reversible, the embedding procedure can be repeated for several times. The multi-layer embedding scheme can be described as following:

$$M_1 \leftarrow M_0 + S_1$$

$$M_2 \leftarrow M_1 + S_2$$

$$\cdots$$

$$M_n \leftarrow M_{n-1} + S_n$$

where M_0 is the original model, M_i represents the output of the *n*th-layer embedding, and S_i represents the secret bit-string embedded in each layer.

In the decoding stage, we can get all secret bit-strings and the original model by reversing the encoding procedures:

$$M_n \rightarrow M_{n-1} + S_n$$

$$M_{n-1} \rightarrow M_{n-2} + S_{n-1}$$

$$\cdots$$

$$M_1 \rightarrow M_0 + S_1$$

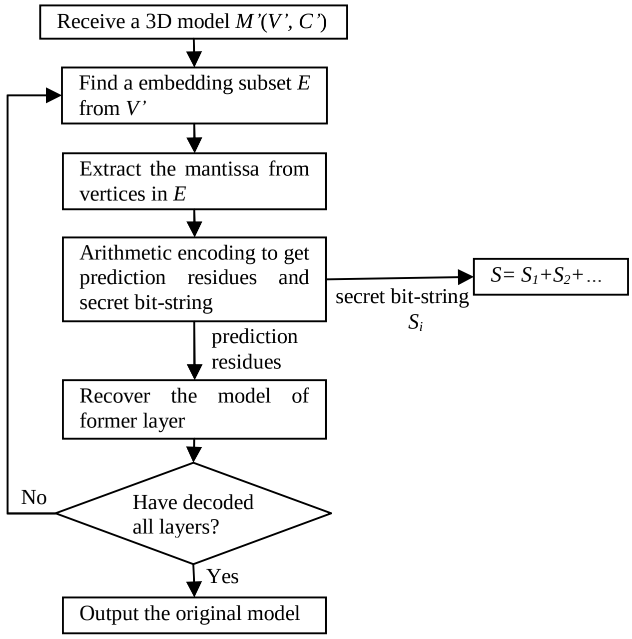


Fig. 4 Overview of the decoding scheme

F. Constraints and limitations

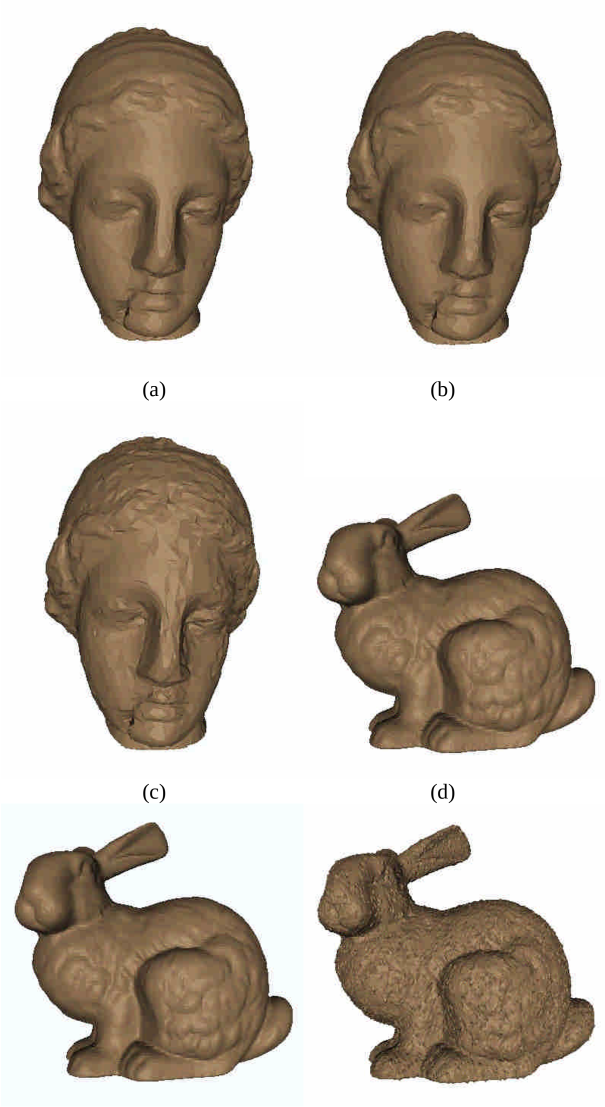
One important issue in the embedding stage is that the vertices in *E* should provide enough space for embedding all elements in *P*. In other words, we need to be sure that $3|E| > |P|$. $|E|$ times by 3 since there are 3 coordinates in each vertex. Since the number of vertices in *E* is fixed, we need to reduce the number of elements produced by arithmetic coding. From the nature of arithmetic coding, a large setting of *Q* will reduce the number of the coded elements. However, according to (3), a large setting of *Q* will increase the distortion of the stego model. If it is not achievable to make $3|E| > |P|$ under a certain setting of *Q*, another possible solution is to increase the precision degree of the embedding vertices for the stego model. For example, a vertex with coordinates (0.1234, 0.3456, 0.5678) in the original model may become (0.123623, 0.345246, 0.567431) in the stego model after embedding. In this example, *Q* is set to 0.001.

To avoid the precision level keeps increasing on some certain vertices; the selection of embedding vertices of the latter layers should eliminate the vertices that have been selected by the former layers till all vertices reaching the same precision level.

IV. Experimental Results and Discussion

We evaluated the proposed watermarking scheme on a set of 3D models. Table 1 shows a list of models and settings used in our experiments. Figure 5 illustrates the influence of the setting of *Q*. Fig. 5(a) shows an original Venus model. The Venus model has 8268 vertices and 16532 faces. 2328 vertices are chosen for embedding datum in the first layer. Fig. 5(b) shows the same view of the stego model with setting of Q = 0.001. This stego model is visually identical to the original model. Fig. 5(c) shows the same view of the stego model with setting of Q = 0.005. We may see the rigid surface on this stego model. Fig. 5(d) shows an original Bunny model. The Bunny model has 35947 vertices and 69451 faces. 9705 vertices are chosen for embedding datum in the first layer. Fig. 5(e) shows the same view of the stego model with setting of Q = 0.0001. This stego model is visually identical to the original model. Fig. 5(f) shows the same view of the stego model with setting of Q = 0.0005. Again, we may see the rigid surface on this stego model.

We have proposed a reversible steganography scheme for 3D models. This scheme embeds the secret information into multiple layers such that it is high confidential and capacity flexible. In this scheme, the maximum distortion on the stego model is controlled by the setting of Q, thus it can be controlled to be imperceptible. One disadvantage of this scheme is that it has to increase the precision level requirement for the stego model. We should pay more effort on this part for the future research.



(a) (b) (c) (d) (e) (f)

Fig. 5 The influence of the setting of Q. (a) The original *Venus* model. (b) Q = 0.001. (c) Q = 0.005. (d) The original *Bunny* model. (e) Q = 0.0001. (f) Q = 0.0005.

TABLE 1
THE MODELS AND SETTINGS USED IN OUR EXPERIMENTS.

Model	Vertex #	Face #	Vertices for embedding in the first layer	Q setting	Original precision level	Stego precision level
Beethoven	2655	5028	768	0.001	10^{-6}	10^{-8}
Bunny	35947	69451	9705	0.0001	10^{-6}	10^{-9}
Fan disk	11984	23964	3355	0.0001	10^{-6}	10^{-9}
Horse	19851	39698	5459	0.001	10^{-6}	10^{-8}
Pulley	25482	50964	7136	0.0001	10^{-6}	10^{-9}
Venus	8268	16532	2328	0.001	10^{-6}	10^{-8}

ACKNOWLEDGMENT

This work is partially funded by the National Science Council Taiwan, under the grant number NSC 97-2221-E-231 -018 -.

REFERENCES

[1] C.C. Chang and T.C. Lu, “A difference expansion oriented data hiding scheme for restoring the original host images,” *J. Syst. Software*, Vol. 79(12), pp. 1754-1766, 2006.
[2] C.C. Chang, J.Y. Hsiao, and C.S. Chan, “Finding optimal LSB substitution in image hiding by dynamic programming strategy,” *Pattern Recognition*, Vol. 36(7), pp. 1583-1595, 2003.
[3] R. Ohbuchi, H. Masuda, and M. Aono, “Watermarking three-dimensional polygonal models through geometric and topological modifications,” *IEEE Journal on Selected Areas in Communication*, Vol. 16(4), pp. 551-560, 1998.
[4] R. Ohbuchi, A. Mukaiyama, and S. Takahashi, “A frequency-domain approach to watermarking 3D shapes,” *Computer Graphics Forum*, Vol. 21(3), pp. 373-382, 2002.
[5] R. Ohbuchi, S. Takahashi, T. Miyazawa, and A. Mukaiyama, “Watermarking 3D polygonal meshes in the mesh spectral domain,” *In Proc. The Graphics Interface*, 2001.
[6] O. Benedens, “Geometry-based watermarking of 3-D models,” *IEEE Computer Graphics and Applications*, Vol. 19(1), pp. 46-55, 1999.
[7] O. Benedens and C. Busch, “Towards blind detection of robust watermarks in polygonal models,” *Computer Graphics Forum*, Vol. 19(3), pp. C199-208, 2000.
[8] E. Praun, H. Hoppe, and A. Finkelstein, “Robust Mesh Watermarking,” *Proc. ACM SIGGRAPH ’99*, pp. 49-56, 1999.
[9] F. Cayre and B. Macq, “Data hiding on 3-D triangle meshes,” *IEEE Trans. Signal Processing*, Vol. 51(4), pp. 939-949, 2003.
[10] C.M. Wang and Y.M. Cheng, “An efficient information hiding algorithm for polygon Models,” *Comput. Graph. Forum*, Vol. 24(3), pp. 591-600, 2005
[11] Min-Wen Chao, Chao-hung Lin, Cheng-Wei Yu, and Tong-Yee Lee, “A High Capacity 3D Steganography Algorithm,” *IEEE* Visualization and Computer Graphics, Vol. 15(2), pp. 274-284, 2009
[12] I.H. Witten, M. Radford, and J.G. Cleary, “Arithmetic coding for data compression,” *Communications of ACM*, Vol. 30(6), pp. 520-540, 1987.