symmetry

Article

A Vertex-Based 3D Authentication Algorithm Based on Spatial Subdivision

Yuan-Yu Tsai [1,2], Yu-Shiou Tsai [1], I-Ting Chi [1] and Chi-Shiang Chan [1,2,*]

1 Department of M-Commerce and Multimedia Applications, Asia University, No. 500, Lioufeng Rd., Wufeng District, Taichung 41354, Taiwan; yytsai@asia.edu.tw (Y.-Y.T.); s0917251664@gmail.com (Y.-S.T.); aa982273@gmail.com (I.-T.C.)

2 Department of Medical Research, China Medical University Hospital, China Medical University, No. 2, Yude Road, North District, Taichung 40447, Taiwan

* Correspondence: CSChan@asia.edu.tw; Tel.: +886-4-23323456 (ext. 20035)

Received: 10 August 2018; Accepted: 18 September 2018; Published: 20 September 2018

Abstract: The study proposed a vertex-based authentication algorithm based on spatial subdivision. A binary space partitioning tree was employed to subdivide the bounding volume of the input model into voxels. Each vertex could then be encoded into a series of binary digits, denoted as its authentication code, by traversing the constructed tree. Finally, the above authentication code was embedded into the corresponding reference vertex by modulating its position within the located subspace. Extensive experimental results demonstrated that the proposed algorithm provided high embedding capacity and high robustness. Furthermore, the proposed algorithm supported controllable distortion and self-recovery.

Keywords: spatial subdivision; fragile watermarking; tamper detection; authentication; 3D models

1. Introduction

The rapid development of computer technology and the Internet, has resulted in most information to now be stored electronically as digital data. Computer networks have replaced conventional communication methods, such as written letters, for information transmission between people. However, the Internet is an open platform. Without proper protection, anyone can steal, destroy, or modify transmitted data. Therefore, how to effectively claim copyrights and protect the integrity of digital data is a popular topic amongst researchers. One of the solutions to the aforementioned problems is copyright marking techniques.

According to various levels of application, copyright marking techniques [1,2] can be categorized into fragile watermarking and robust watermarking, according to the robustness of the embedded watermark. Fragile watermarking technique aims to protect the integrity of multimedia data through tamper detection; this technique can verify the integrity of multimedia content, even if the data have been only slightly modified. Robust watermarking technique is used to protect the copyright of multimedia content. It detects the copyright information embedded in multimedia data, when the data have been maliciously or non-maliciously attacked. The difference in practical scenarios between the above two techniques can be illustrated as follows. A 3D model designer creates a fantastic work and wants to share and disperse it all over the world. Before this, the designer should adopt 'robust' watermarking algorithms to embed their confidential personal information into the 3D model. Thus, the designer can claim rightful ownership when an adversary steals or plagiarizes the work. In another scenario, a designer completes a 3D model with all the functionality requested by the customer and then sends it out for their approval. The functionality of the 3D model may be compromised, even after inadvertent modification by a third party during transmission.

Therefore, the designer can adopt 'fragile' watermarking algorithms to embed the authentication code into the 3D model in advance, for integrity checking and tamper detection. Relating to industrial applications for 3D copyright protection, Treatstock introduced the 'Watermark3D' software solution [3] for 3D printable files. This software can allow creators to adopt different watermarks to files given to different users. Thus, the illegal user can be detected by extracted watermark. Figure 1 shows its system flowchart.

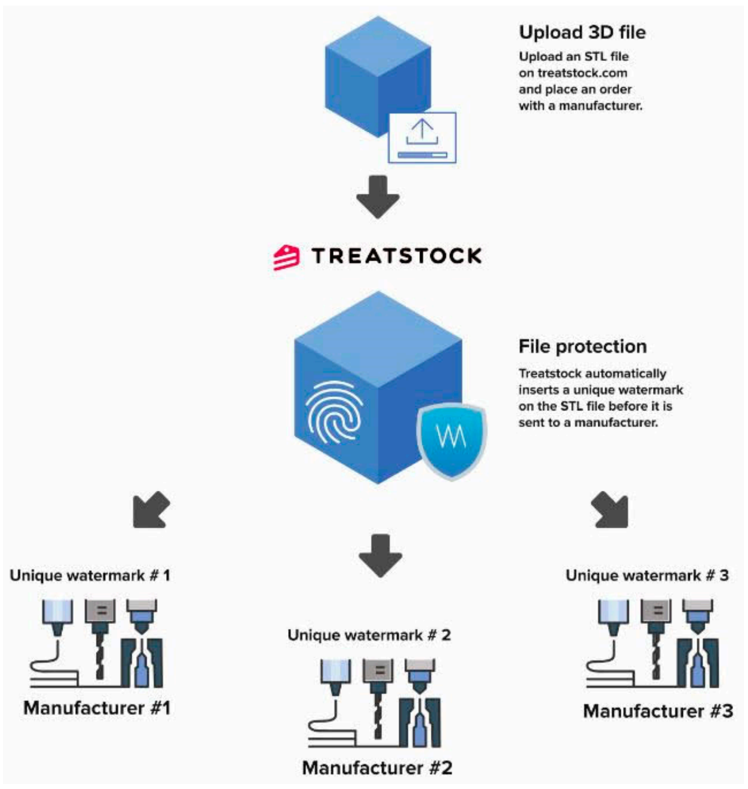

Upload 3D file
Upload an STL file on treatstock.com and place an order with a manufacturer.
TREATSTOCK
File protection
Treatstock automatically inserts a unique watermark on the STL file before it is sent to a manufacturer.
Unique watermark # 1
Manufacturer #1
Unique watermark # 2
Manufacturer #2
Unique watermark # 3
Manufacturer #3

Figure 1. The system flowchart for Watermark3D software.

Widely used in digital images and videos [4–15], fragile watermarking technique has been a popular research topic, and numerous studies have obtained favorable results. The research and development of 3D model protection mechanisms, which have numerous applications, have gradually received scholarly attention. Based on the size of the tampered area, fragile watermarking technique can be categorized into two types: Region-based and Vertex-based. Region-based fragile watermarking technique detects tampered areas relatively loosely, and the detected area may contain vertices that have not been tampered with; however, this technique effectively detects the tampered topology. In contrast, vertex-based fragile watermarking technique accurately detects the tampered vertices but cannot detect changes in topology because of the limitation of the algorithm used.

According to the latest survey [16] and our understanding about the 3D authentication algorithm, there are currently eight vertex-based 3D model authentication algorithms [17–24], which are implemented in the spatial domain. Despite these algorithms obtaining favorable experimental results, there is still room for improvement. Some techniques ignore geometric operations, such as translation, rotation, and uniform scaling, which are common for 3D models, and this prevents the authentication code from being correctly extracted. Furthermore, the length of the authentication code used in some approaches is insufficient, resulting in a high rate of false alarms after a vertex has been tampered with. Finally, some technologies do not support blind extraction schemes, limiting their practicality.

Therefore, this paper proposes a vertex-based 3D model authentication algorithm based on spatial subdivision and provides solutions to the aforementioned deficiencies. First, we use a binary space partitioning tree (BSP tree) and an embedding threshold to decompose the bounding volume of the

3D model into numerous subspaces, and then we employ a tree structure to encode each vertex into a series of binary data, denoted as its authentication code. Subsequently, we use the secret key to determine the reference vertex corresponding to each processing vertex. Finally, the position of the reference vertices in the subspace is altered, and the encoding result of the processing vertex is embedded to complete authentication code embedding, enabling self-recovery. Compared with previously developed algorithms, the proposed technique has high embedding capacity and ensures highly robust authentication codes. In addition, this technique effectively and accurately detects tampered areas, executing self-recovery to restore vertex values. The proposed technique can also be applied to polygonal models and point-cloud models, to implement new concepts in the field of 3D model authentication.

The remainder of the paper is organized as follows: Section 2 introduces eight vertex-based 3D model authentication algorithms implemented in the spatial domain. This section also illustrates Tsai et al.'s tree-based information hiding algorithm [25] for 3D models. Section 3 details the method proposed by this study. Section 4 presents the experimental results and discussion; and Section 5 presents the conclusions and future research directions.

2. Related Works

2.1. Vertex-Based 3D Authentication Algorithms

Chen et al.'s proposed algorithm [17] can be regarded as the first vertex-based 3D authentication for point-cloud models, if the vertex degree is discarded during the encoding/decoding processes. The encoding process is initialized by feeding vertex's x- and y-coordinates into the hash function to obtain the hash value. The above value is then embedded into its z-coordinate based on the quantization index modification method [26]. The authentication process is finally achieved by comparing the reconstructed, with the pre-embedded, hash value to detect if the vertex has been maliciously altered.

Wang et al. [18] proposed a reversible fragile watermarking scheme for 3D models, with the vertex coordinate represented by the IEEE 754 single precision format. A reversible quantization index modification embedding scheme is employed to modulate the distance from the vertex to the gravity center of the input model. The authors claimed that the proposed algorithm was immune to the causality and convergence problems. Furthermore, the spread spectrum technique made their proposed algorithm robust against vertex reordering and similarity transformation attacks. Huang et al. [20] also adopted the quantization index modification technique to embed the watermark into the r coordinate in the spherical coordinate system for authentication and verification. Model distortion can be controlled by quantization step setting.

Xu and Cai [19] adopted principal component analysis to obtain the spherical coordinates mapping the square-matrix, and a binary image watermark is embedded into the square-matrix with embedding parameter α. Although the proposed algorithm is robust against similarity transformation attacks, the original mapping square-matrix is needed for correct watermark extraction. Wang et al. [21] proposed a vertex-based authentication algorithm based on hamming code. The coordinates of each vertex are firstly normalized into the range 0 to 1, and then represented as a series of binary digits. The fourth to seventh least significant bits are used to generate three parity check bits, and subsequently replace the eighth to tenth bits to complete embedding process. The proposed algorithm is efficient but with low embedding capacity and low robustness. Chang et al. [23] also proposed a hamming code based fragile watermarking scheme for 3D models. They first transformed the input model from a Cartesian coordinate system to a Spherical one. The watermark is then embedded into the r component based on the least-significant-bit substitution scheme.

Wang et al. [22] proposed a novel chaos sequence-based 3D fragile watermarking algorithm for 3D models. Instead of the hash function, they adopted chaotic systems to generate chaotic sequences to enhance the security of the embedded watermark. The proposed method can provide the capability of accurately verifying and locating the tampered region, to protect the integrity of 3D objects. Finally, Wang et al. [24] employed output feedback (OFB) to generate the embedded watermark, and the watermark is embedded based on the least-significant-bit substitution scheme. The proposed algorithm can have smaller distortions.

2.2. Tsai et al.'s Tree-Based Information Hiding Algorithm

Tsai et al. proposed an information hiding algorithm for 3D models based on a BSP tree. The approach uses principal component analysis to analyze the input 3D model and to resist similarity transformation attacks. The principal component analysis produces a coordinate system with three orthogonal coordinate axes; the origin is the center of gravity of the 3D model, and the coordinate axes are the eigenvectors of the vertex of the 3D model. Subsequently, the bounding volume of the 3D model is established on this coordinate system, and a threshold value is introduced. Through the BSP tree, the bounding volume is divided into numerous subspaces according to the distribution of the vertex; these subspaces are known as voxels (Figure 2).

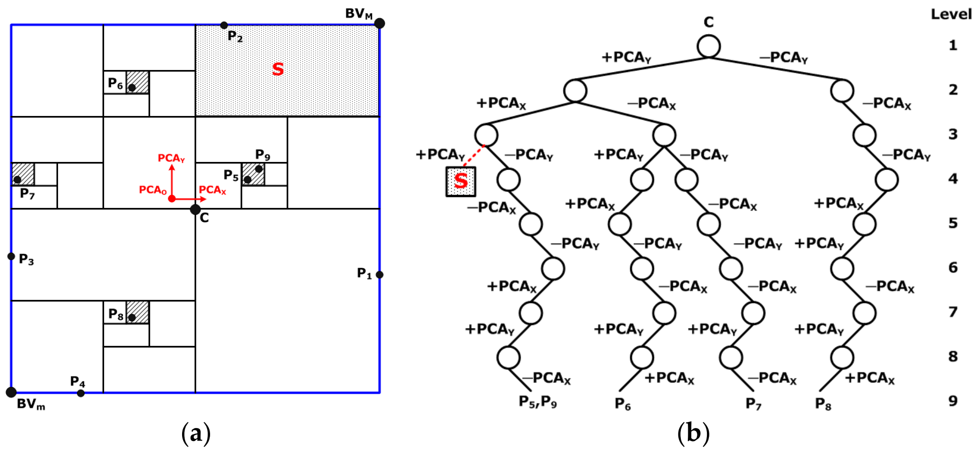


Figure 2. Examples of Tsai et al.'s algorithm: (**a**) the BSP spatial subdivision result; (**b**) the BSP tree structure.

In the embedding process, each voxel is visited using the tree structure. If there is more than one vertex within a voxel, the vertices do not undergo message embedding. For a voxel with only a single vertex, the subspace is decomposed into 2^m subspaces, and the secret key is employed to number each subspace. According to the secret message of m bits, the vertex is moved into the corresponding subspace, completing message embedding.

3. The Proposed Algorithm

This paper extends the information hiding algorithm of the 3D model based on a BSP tree proposed by Tsai et al., to an authentication algorithm. In the original algorithm, the visiting process of the tree structure exactly represents the encoding result of each vertex. Given a bounding volume equivalent to the original model in size, the vertex positions can be located using the aforementioned encoding results. When the threshold value in the subdivision process is lower, the vertex can be accurately located. However, the threshold value, which is limited by the number of decimal places represented by the vertex coordinate value in the file, should not be too low; otherwise, the accurate secret message cannot be extracted.

The proposed approach is divided into two procedures: Authentication code embedding and extraction (Figure 3). The authentication code embedding procedure comprises four processes: Preprocessing, vertex encoding, authentication code generation, and authentication code embedding. The authentication code extraction procedure is composed of five processes: Preprocessing, vertex encoding, authentication code reconstruction, authentication code extraction, and tamper detection and self-recovery.

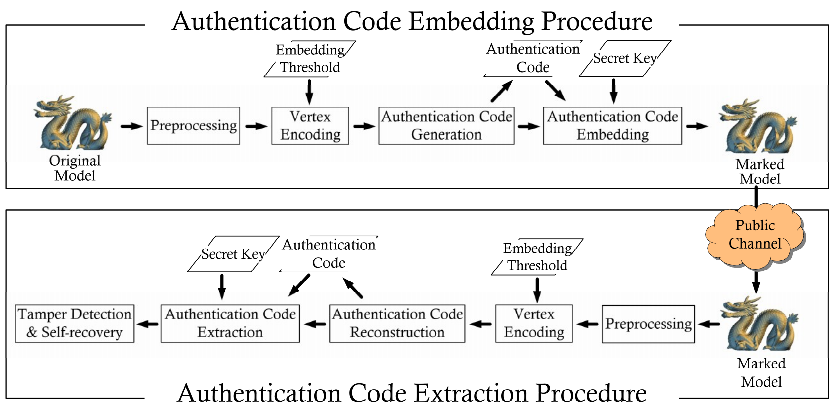

Authentication Code Embedding Procedure
Embedding Threshold
Authentication Code
Secret Key
Original Model
Preprocessing
Vertex Encoding
Authentication Code Generation
Authentication Code Embedding
Marked Model
Public Channel
Secret Key
Authentication Code
Embedding Threshold
Tamper Detection & Self-recovery
Authentication Code Extraction
Authentication Code Reconstruction
Vertex Encoding
Preprocessing
Marked Model
Authentication Code Extraction Procedure

Figure 3. Flowchart of the proposed method.

3.1. Preprocessing Process

In this process, the 3D model is first read to obtain the related information of the original model. Let the number of vertices be V_N and the number of polygons be P_N. After reading the information of vertex V_i and polygon P_K, a bounding volume of the 3D model is established on the basis of Cartesian coordinates. The magnitude of the bounding volume, DL_{BV}, is determined using the distance between the boundary points BP_M and BP_m as follows:

$$\begin{cases} BP_{M_d} = max\left(V_{i_d}\right) \\ BP_{m_d} = min\left(V_{i_d}\right) \end{cases}, \quad d = x, y, z; \; i = 1, 2, \ldots, V_N \tag{1}$$

$$DL_{BV} = \sqrt{\left(BP_{M_x} - BP_{m_x}\right)^2 + \left(BP_{M_y} - BP_{m_y}\right)^2 + \left(BP_{M_z} - BP_{m_z}\right)^2} \tag{2}$$

3.2. Vertex Encoding Process

We use a BSP tree with an embedding threshold to decompose the bounding volume into numerous subspaces according to the distribution of vertices, and each vertex must fall within a certain subspace. Using the tree structure's visiting process, each vertex can be encoded. Assuming the leftward visit is 0 and the rightward visit is 1, taking vertices P_5 and P_8 in Figure 2 as examples, P_5 is encoded as 00111001 and P_8 is encoded as 11100100.

3.3. Authentication Code Generation Process

This process converts the aforementioned encoding result into three weights w_x, w_y, and w_z, between 0 and 1. Each weight will be used for each corresponding coordinate axis in the authentication code embedding process. First, three-bit data is grabbed each time from the encoding result to generate a decimal digit. Thus, a string of decimal digits is generated. Second, we break the above number into three groups, working from left to right, and insufficient places are filled with 0. Finally, three weights are obtained by converting the number of each group into decimals between 0 and 1. Taking the encoding results of vertices P_5 and P_8 from the previous process as examples, the encoding result of P_5 becomes $(\underline{001}\,\underline{110}\,\underline{01})_2 = (\underline{1}\,\underline{6}\,\underline{1})_{10}$, generating the decimals 0.1, 0.6, and 0.1; and the encoding result of P_8 is converted to $(\underline{111}\,\underline{001}\,\underline{00})_2 = (\underline{7}\,\underline{1}\,\underline{0})_{10}$, producing the decimals 0.7, 0.1, and 0.0.

3.4. Authentication Code Embedding Process

To be able to execute self-recovery in the authentication process, a secret key is used to obtain the reference vertex corresponding to each embedded vertex; in addition to increasing the security and privacy of the authentication code, this enables the technique to support self-recovery. For each embedding vertex V, its authentication code is obtained in the previous process, and the boundary vertex $BV_m^{R_V}$ of the subspace in which the corresponding reference vertex R_V is located can be found. The authentication code is embedded into the reference vertex, as indicated in the following:

$$R'_{V_d} = BV_{m_d}^{R_V} + L_d^{R_V} \times w_d, \;\; d = x, y, z \tag{3}$$

where R'_V is the generated data-embedded reference vertex which carries the authentication code information, and $L_d^{R_V}$ is the length, width, and height of the subspace in which R_V is located.

The procedure presented in the flowchart (Figure 3) should be followed when the integrity of a model must be verified. First, the information related to the marked model is obtained in the preprocessing process, and the encoding result of each vertex is extracted during the vertex encoding process. Then, in the authentication code reconstruction process, the authentication code of a vertex is reconstructed using the scheme illustrated in Section 3.3. Whether the vertex has been tampered with can be determined by comparing the three weights derived from the subspace where the reference vertex is located. If a vertex is confirmed to have been tampered with, its authentication code is extracted from its corresponding reference vertex, and the BSP tree is used to locate the vertex in a certain subspace to complete self-recovery.

4. Experimental Evaluations

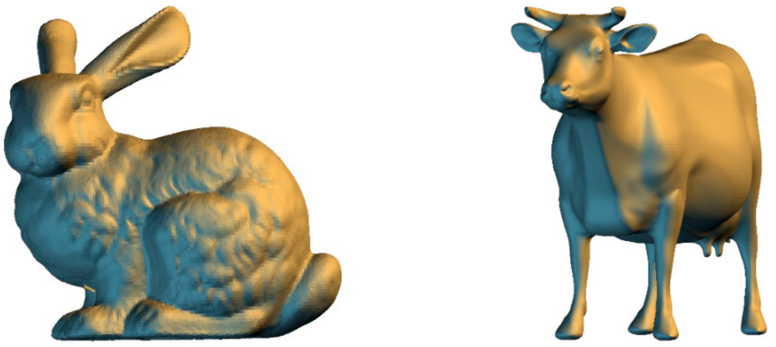
This section presents the results of experiments using the proposed method to demonstrate its feasibility. The information related to the 3D models used in this experiment and the visual effects of the models are shown in Figure 4. Experiments were performed using the C programming language, on a personal computer equipped with an Intel Core i7 3.40 GHz processor and 32 GB of memory. The normalized root-mean-square error was used to calculate the level of distortion between the original model and the model with authentication codes, and it was calculated as follows:

$$NRMSE = \left(\sqrt{\sum_{i=1}^{N_V} \|C_i - S_i\|^2 / N_V} / DL_{BV} \right) \times 100\% \tag{4}$$

where C_i and S_i represent the i-th vertex in the original model and embedded model, respectively.

First, the experimental results were presented, consisting of the embedding capacity and the quality of the models with the authentication code embedded. The visual effects of the data-embedded models under various embedding thresholds were then detailed. We also presented the experimental results for tamper detection, to demonstrate the feasibility of our proposed algorithm. Finally, the feasibility of the proposed technique was confirmed through comparison between currently available methods and the proposed method.

Tables 1 and 2 list the encoding lengths for each model under various embedding thresholds, and the model distortion after authentication code embedding, respectively. Figures 5–8 shows the visual effects for each model with the authentication code embedded under various thresholds. When a lower embedding threshold was employed, the subspaces were smaller, the tree structure was larger, the encoding length of each vertex was higher, and the small subspace size limited the modification of vertex coordinates, resulting in a lower level of distortion. Given the space subdivision operation was related to the diagonal length ratio of the bounding volume, the encoding length of the vertices for each model was similar. The code length of each vertex was 24 bits when the threshold was 0.005, and 36 bits when the threshold was 0.0003. The model distortion decreased from 0.166% to 0.009% as the threshold value decreased.


(**a**) Bunny; $V_N = 34834,\ DL_{BV} = 25.02$ (**b**) Cow; $V_N = 46433,\ DL_{BV} = 30.49$

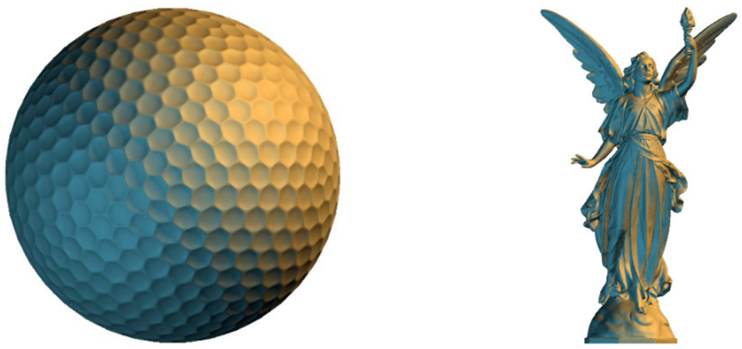

(**c**) Golf ball; $V_N = 122882,\ DL_{BV} = 17.31$ (**d**) Lucy; $V_N = 262909,\ DL_{BV} = 1918.29$

Figure 4. Visual effects of our test models: (**a**) Bunny Model; (**b**) Cow Model; (**c**) Golf ball Model; (**d**) Lucy Model.

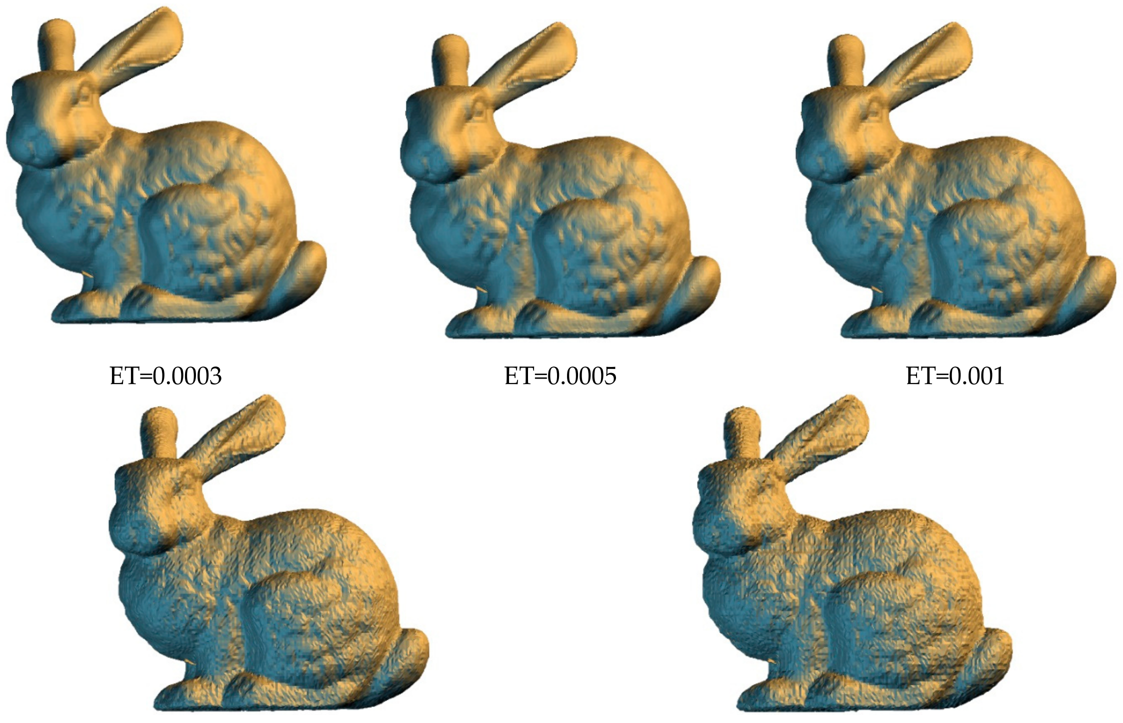


Figure 5. Visual effects of Bunny model with message embedded under various thresholds.

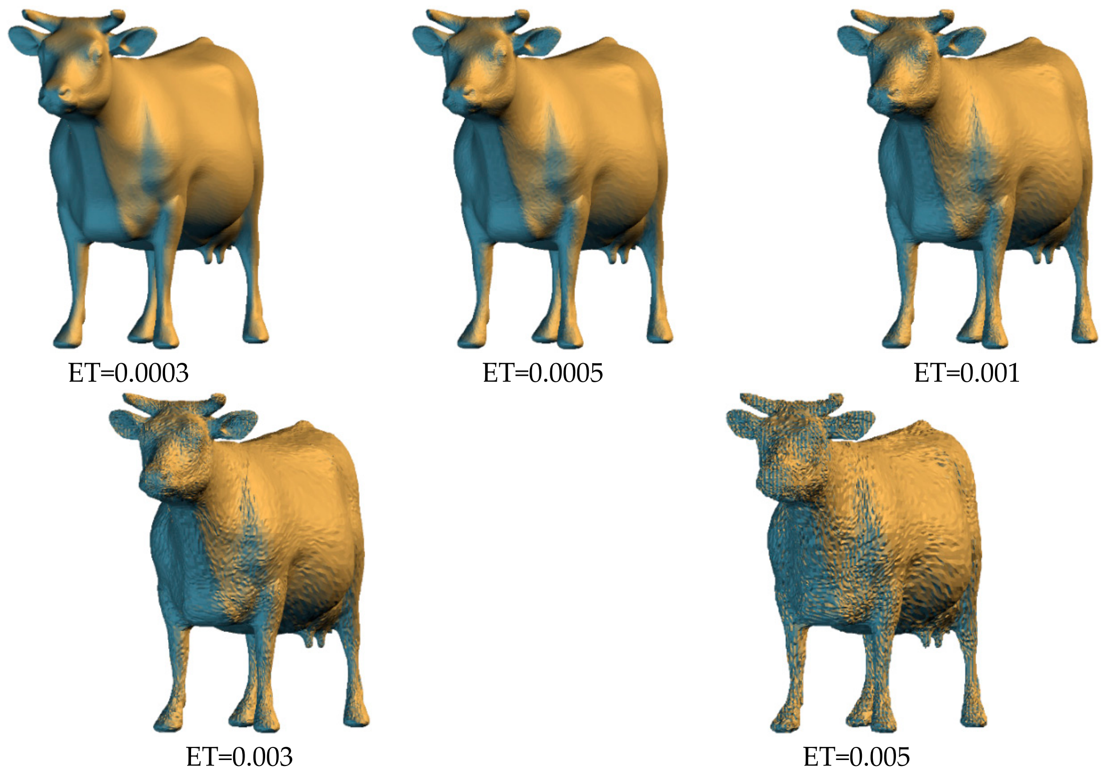


Figure 6. Visual effects of Cow model with message embedded under various thresholds.

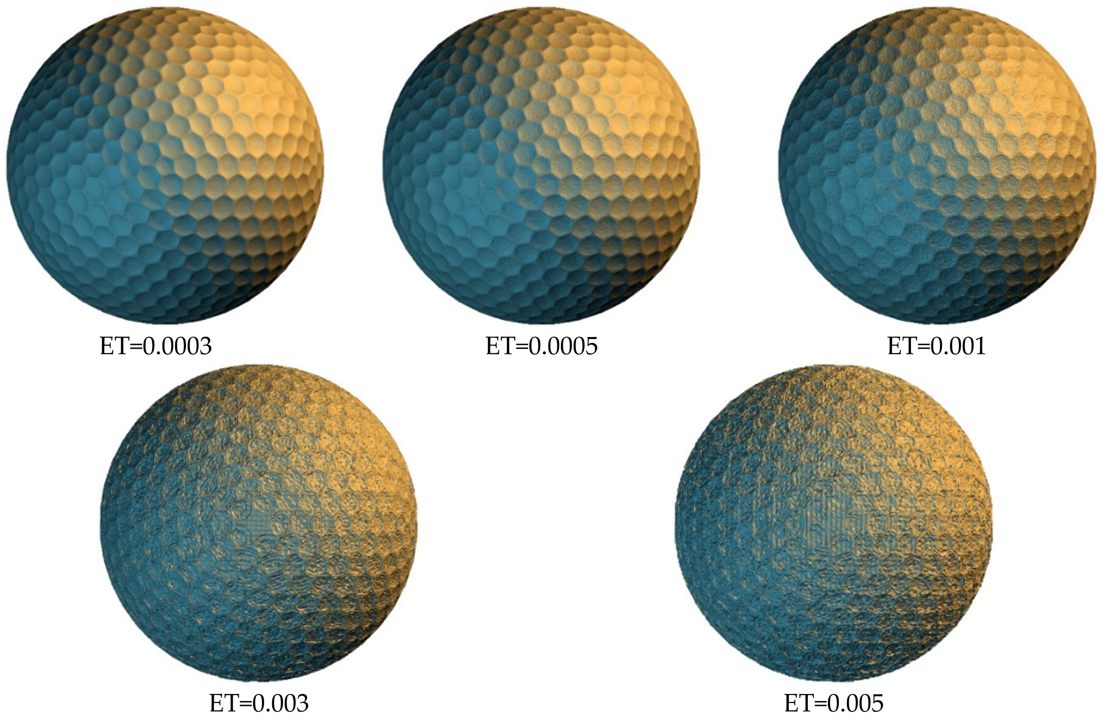


Figure 7. Visual effects of Golf ball model with message embedded under various thresholds.

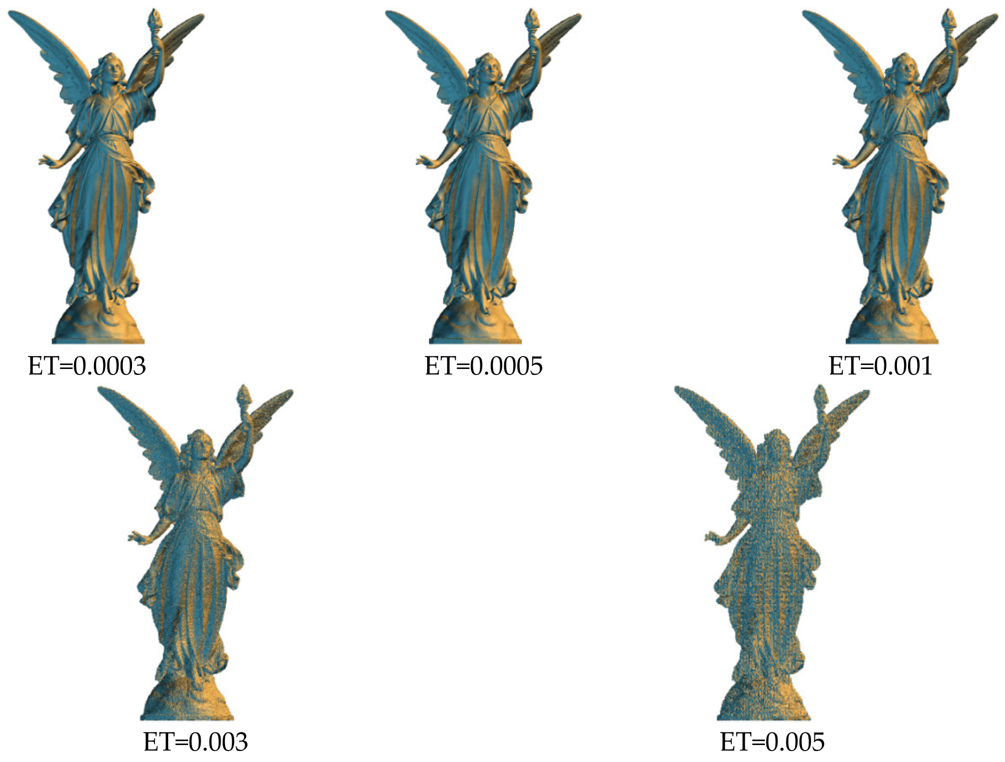


Figure 8. Visual effects of Lucy model with message embedded under various thresholds.

Table 1. Encoding length of each vertex, for each model, under various embedding thresholds.

–		Length				
Embedding Threshold	–	0.005	0.003	0.001	0.0005	0.0003
Model	Bunny	24	26	30	33	36
	Cow	24	27	30	33	36
	Golf ball	24	26	30	33	36
	Lucy	23	26	30	33	35

Table 2. Model distortion under various embedding thresholds.

–		Model Distortion				
Embedding Threshold	–	0.005	0.003	0.001	0.0005	0.0003
Model	Bunny	0.148%	0.097%	0.037%	0.018%	0.009%
	Cow	0.148%	0.075%	0.037%	0.019%	0.009%
	Golf ball	0.150%	0.106%	0.037%	0.019%	0.009%
	Lucy	0.166%	0.083%	0.037%	0.019%	0.010%

Table 3 shows the average number of detected suspicious vertices, for each model, under different embedding thresholds, for 100 iterations. In each iteration, we randomly added or subtracted 0.01% of the diagonal length of the bounding volume of the test model from x, y, and z coordinate value of 50 vertices separately. The proposed algorithm was then used to perform tamper detection and self-recovery operation. The experimental results showed that all modified vertices could be detected. However, the modified vertex may affect the construction results of the BSP tree, resulting in unaltered vertices with different encoding results. Fortunately, the entire tree structure can be entirely recovered after moving each tampered vertex to its original located subspace. With the increasing value of the embedding threshold, the varied range of the modified vertex may be limited within the same subspace, and the tree structure is not affected. Thus, the number of suspicious vertices is decreased.

The only problem is that the boundary vertices are the key to successfully construct the same tree structure and cannot be modified. Otherwise, the algorithm fails to perform the following processes.

Table 3. Average number of suspicious vertices detected under different embedding thresholds.

–		**Number**				
Embedding Threshold	–	0.005	0.003	0.001	0.0005	0.0003
Model	Bunny	57.96	63.81	68.87	78.81	105.64
	Cow	57.97	65.39	70.18	89.15	105.89
	Golf ball	68.66	73.64	79.87	90.97	115.35
	Lucy	60.63	65.31	73.56	89.68	105.30

Finally, Table 4 is a comparison of the proposed method in this study, with currently available methods. The embedding methods for current algorithms can be categorized into four groups, including quantization index modification (QIM), coefficient modification (CM), hamming code (HC), and least-significant-bit substitution (LSBs). Our proposed algorithm was based on message-digit conversion (MC). LSBs-based algorithms directly replace the coordinate value by authentication code. Therefore, except Reference [23] proposed in the spherical coordinate system, common similarity transformation attacks can terminate the algorithm execution. From the aspect of embedding capacity, although the technique proposed by Wang et al. [18] had the highest embedding capacity, this technique stored vertex coordinates values using the IEEE single-precision floating-point format, whereas our technique presented the numerical values in a 3D model to six decimal places and had an embedding capacity of 36 bits. Wang et al.'s algorithm [24] was also performed on the 3D model with binary representation; their embedding capacity may be similar to ours. In addition, the proposed approach controlled the encoding length of vertices by setting the threshold, resulting in an adaptive embedding capacity, and the model distortion could also be controlled by the threshold. QIM-based algorithms [17,18,20] can also have the capability of controllable distortion with settable quantization steps. Finally, the bounding volume used by the BSP tree was produced by the boundary vertices. To maintain the same bounding volume, six or fewer vertices cannot be embedded with messages.

Table 4. Comparisons of the proposed method with previous ones.

Algorithm	**[17]**	**[18]**	**[19]**	**[20]**	**[21]**	**[22]**	**[23]**	**[24]**	**Proposed**
Blind extraction	Semi-blind	Blind	Semi-blind	Semi-blind	Blind	Blind	Blind	Blind	Blind
Embedded ratio	N_V	N_V	N_V	N_V	N_V	N_V	N_V	N_V	$N_V - 6$
Embedding capacity	1	48	1	1	3	1	1	32	36
Distortion control	Yes	Yes	No	Yes	No	No	No	No	Yes
Robustness *	VR	R, S, T	R, S, T	VR	VR	None	T	None	T, S, VR
Embedding method	QIM	QIM	CM	QIM	HC	LSBs	LSBs	LSBs	MC

* R: Rotation, S: Scaling, T: Translation, VR: Vertex Reorder.

5. Conclusions and Future Studies

This study extended Tsai et al.'s tree-based 3D information hiding algorithm to a 3D model authentication algorithm. In the framework of the BSP tree, the tree traversal result of each vertex can be the encoding result of the vertex. With the absence of vertex coordinate values, the encoding result can be used to locate the vertex in a certain subspace in the 3D space; where the vertex can be accurately located when the encoding result is sufficiently long. Since this technique does not require the connection attributes between vertices, it can be applied to polygonal models and point-cloud models. In addition to using the BSP tree technique, we employed the message-digit conversion mechanism to embed the encoding result into vertices and use a secret key to determine the reference vertex corresponding to each embedded vertex, for effectively performing self-recovery. The experimental results indicated that the proposed method had high embedding capacity and a relatively low false alarm rate. The 3D models with authentication codes embedded presented a decent visual effect, proving that the proposed algorithm was feasible.

In the future, we will continue to explore and enhance the robustness of authentication coding, so that it can resist both non-malicious attacks (e.g., rotation, translation, and scaling) and malicious attacks, such as vertex deletion. In addition, we will discuss variations of the vertex encoding and authentication code embedding procedure to embed more authentication code. Finally, it is worth investigating how to determine the threshold value and the embedding order of multiple vertices in the same subspace.

Author Contributions: Y.-Y.T. supervised the whole work, designed the experiments, analyzed the performance of the proposed idea, and wrote the paper. C.-S.C. instructed Y.-S.T. and I.-T.C. to perform experiments, simulations, and collect the results.

Funding: This research was funded by Ministry of Science and Technology of Taiwan under the grant numbers MOST 106-2813-C-468-092-E, MOST 106-2632-E-468-001, and MOST 106-2218-E-468-001.

Acknowledgments: The authors would like to thank the editor and three anonymous reviewers for their constructive suggestions in improving the paper significantly.

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

References

1. Cox, I.J.; Miller, M.L.; Bloom, J.A.; Fridrich, J.; Kalker, T. *Digital Watermarking and Steganography*, 2nd ed.; Morgan Kaufmann: Burlington, HR, USA, 2008.
2. Petitcolas, F.A.P.; Anderson, R.J.; Kuhn, M.G. Information Hiding—A Survey. *Proc. IEEE* **1999**, *87*, 1062–1078. [CrossRef]
3. Watermark3D. Available online: https://www.watermark3d.com/ (accessed on 17 September 2018).
4. Ahmad, F.; Cheng, L.M. Authenticity and Copyright Verification of Printed Images. *Signal Process* **2018**, *148*, 322–335. [CrossRef]
5. Chen, C.H.; Tang, Y.L.; Hsieh, W.S. Color Image Authentication and Recovery via Adaptive Encoding. *Math. Probl. Eng.* **2014**, *2014*. [CrossRef]
6. Lee, C.W.; Tsai, W.H. A Data Hiding Method Based on Information Sharing via PNG Images for Applications of Color Image Authentication and Metadata Embedding. *Signal Process* **2013**, *93*, 2010–2025. [CrossRef]
7. Karsh, R.K.; Saikia, A.; Laskar, R.H. Image Authentication Based on Robust Image Hashing with Geometric Correction. *Multimed. Tools Appl.* **2018**, *77*, 25409–25429. [CrossRef]
8. Qin, C.; Ji, P.; Zhang, X.; Dong, J.; Wang, J. Fragile Image Watermarking with Pixel-wise Recovery Based on Overlapping Embedding Strategy. *Signal Process.* **2017**, *138*, 280–293. [CrossRef]
9. Qin, C.; Wang, H.; Zhang, X.; Sun, X. Self-embedding Fragile Watermarking Based on Reference-data Interleaving and Adaptive Selection of Embedding Mode. *Inf. Sci.* **2016**, *373*, 233–250. [CrossRef]
10. Rosales-Roldan, L.; Cedillo-Hernandez, M.; Nakano-Miyatake, M.; Perez-Meana, H.; Kurkoski, B. Watermarking-based Image Authentication with Recovery Capability Using Halftoning Technique. *Signal Process Image* **2013**, *28*, 69–83. [CrossRef]
11. Sowmya, K.N.; Chennamma, H.R.; Rangarajan, L. Video Authentication Using Spatio Temporal Relationship for Tampering Detection. *J. Inf. Secur. Appl.* **2018**, *41*, 159–169.
12. Tew, Y.; Wong, K.; Phan, R.C.-W.; Ngan, K.N. Multi-layer Authentication Scheme for HEVC Video Based on Embedded Statistics. *J. Vis. Commun. Image Represent.* **2016**, *40*, 502–515. [CrossRef]
13. Ulutas, G.; Ustubioglu, A.; Ustubioglu, B.V.; Nabiyev, V.; Ulutas, M. Medical Image Tamper Detection Based on Passive Image Authentication. *J. Digit. Imaging* **2017**, *30*, 695–709. [CrossRef] [PubMed]
14. Wang, Q.; Alfalou, A.; Brosseau, C. Security Enhanced Multiple-image Authentication Based on Cascaded Optical Interference and Sparse Phase Mixed Encoding. *Opt. Commun.* **2016**, *372*, 144–154. [CrossRef]
15. Wu, W.C. Quantization-based Image Authentication Scheme Using QR Error Correction. *EURASIP J. Image Video Process.* **2017**, *2017*. [CrossRef]
16. Borah, S.; Borah, B. Watermarking Techniques for Three Dimensional (3D) Mesh Authentication in Spatial Domain. *3D Res.* **2018**, *9*, 1–22. [CrossRef]
17. Chen, T.Y.; Hwang, M.S.; Jan, J.K. Adaptive Authentication Schemes for 3D Mesh Models. *Int. J. Innov. Comput. Inf. Control* **2009**, *5*, 4561–4572.

18. Wang, J.T.; Yang, Y.W.; Chang, Y.T.; Yu, S.S. A High Verification Capacity Reversible Fragile Watermarking Scheme for 3D Models. *Int. J. Innov. Comput. Inf. Control* **2011**, *7*, 365–378.
19. Xu, T.; Cai, Z.G. A Novel Semi-fragile Watermarking Algorithm for 3D Mesh Models. In Proceedings of the International Conference on Control Engineering and Communication Technology, Liaoning, China, 7–9 December 2012; pp. 782–785.
20. Huang, C.C.; Yang, Y.W.; Fan, C.M.; Wang, J.T. A Spherical Coordinate Based Fragile Watermarking Scheme for 3D Models. In Proceedings of the International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems, Amsterdam, The Netherlands, 17–21 June 2013; pp. 566–571.
21. Wang, J.T.; Chang, Y.C.; Yu, C.Y.; Yu, S.S. Hamming Code Based Watermarking Scheme for 3D Model Verification. *Math. Probl. Eng.* **2014**, *2014*. [CrossRef]
22. Wang, J.T.; Yang, W.H.; Wang, P.C.; Chang, Y.T. A Novel Chaos Sequence Based 3D Fragile Watermarking Scheme. In Proceedings of the International Symposium on Computer, Consumer and Control, Taichung, Taiwan, 10–12 June 2014; pp. 745–748.
23. Chang, Y.C.; Wang, J.T.; Chang, Y.T.; Yu, S.S. An Error-detecting Code Based Fragile Watermarking Scheme in Spherical Coordinate System. In Proceedings of the International Symposium on Computer, Consumer and Control, Xi'an, China, 4–6 July 2016; pp. 287–290.
24. Wang, J.T.; Chang, Y.C.; Lu, C.W.; Yu, S.S. An OFB-based Fragile Watermarking Scheme for 3D Polygonal Meshes. In Proceedings of the International Symposium on Computer, Consumer and Control, Xi'an, China, 4–6 July 2016; pp. 291–294.
25. Tsai, Y.Y.; Wang, C.M.; Cheng, Y.M.; Chang, C.H.; Wang, P.C. Steganography on 3D Models Using a Spatial Subdivision Technique. In Proceedings of the 24th Computer Graphics International Conference, Xi'an, China, 4–6 July 2016; pp. 469–476.
26. Wang, P.C.; Wang, C.M. Reversible Data Hiding for Point-Sampled Geometry. *J. Inf. Sci. Eng.* **2007**, *23*, 1865–1887.