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 polygon 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 $O(log(n!)) = O(nlogn)$ bits, which is much better than the results of the previous work for 3D polygonal meshes but at the expense of computation time $\Omega(n^2 log^2 nloglogn)$. 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.

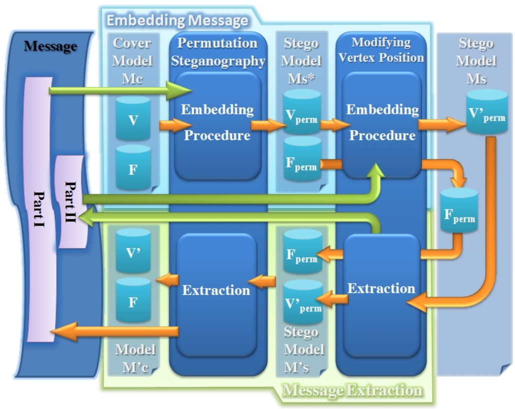


Figure 1: The framework of hybrid data hiding approach.

3. PROPOSED APPROACH

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, $M_c = (V, F)$, 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 indexed in a non-decreasing order from left to right. Figure 2 shows an example of binary trie with $n = 11$ embedding primitives at the leaf node (red) where each is indexed and associated with the corresponding embedded bitstream shown below the index.

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, $k = \lfloor log_2(n - i) \rfloor$, 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 $\lceil log_2(n - i)) \rceil - 1$ 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.

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

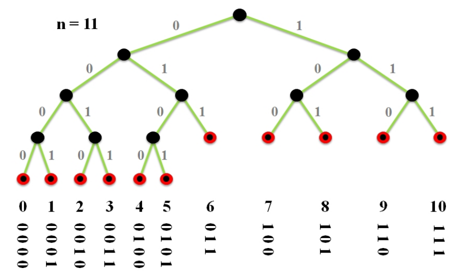


Figure 2: An example of the binary trie search structure with *11* embedding primitives at the leaf node (red). The index and the embedded bitstream are shown below the leaf node.

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 $n^i_{voxel} = l^i_{BV}/(2^{k_i} \times l^i_v)$, l^i_{BV} is the side length of the bounding volume, and l^i_v 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.

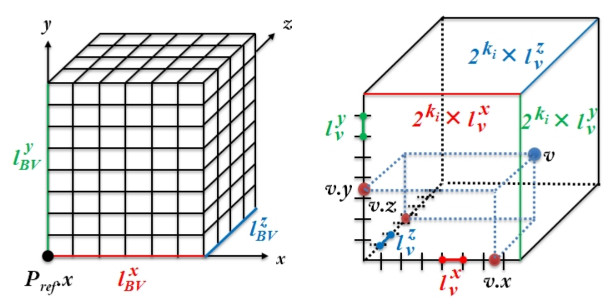


Figure 3: The axis aligned bounding volume of the cover model is conceptually subdivided into voxels, and the voxel is again subdivided into 2^{k_i} units for each component x, y, and z. The vertex v is embedded in k_x, k_y, and k_z bits in x, y, and z components respectively.

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.

Input: $perm[]$
Output: *output*[]
// $perm[]$ is the output primitive permutation from permutation
// steganography.
// p_{ref} is the minimal vertex of the bounding volume.
// n is the number of the vertices.
// $peek(k_i)$ peeks next k_i bits from the embedding message.
for $i = 0$ **to** n - 1
$v = perm[i]$

$$output[i].x \leftarrow p_{ref}.x + \lfloor \frac{\|v.x - p_{ref}.x\|}{2^{k_x} \times l_v^x} \rfloor \times (2^{k_x} \times l_v^x) + INT(peek(k_x)) \times l_v^x;$$

$$output[i].y \leftarrow p_{ref}.y + \lfloor \frac{\|v.y - p_{ref}.y\|}{2^{k_y} \times l_v^y} \rfloor \times (2^{k_y} \times l_v^y) + INT(peek(k_y)) \times l_v^y;$$

$$output[i].z \leftarrow p_{ref}.z + \lfloor \frac{\|v.z - p_{ref}.z\|}{2^{k_z} \times l_v^z} \rfloor \times (2^{k_z} \times l_v^z) + INT(peek(k_z)) \times l_v^z;$$

end

Figure 4: Pseudo code of the embedding procedure of *MVP*.

3.2.1 Extraction by Permutation Steganography

The message extraction procedure of [1], is applied to extract the message in the stego model. Given a permutation of n primitives in the stego model, again the same reference ordering as one from the cover model is computed. And, we build the binary trie search structure for the primitives in the reference ordering. At each step i, we choose the next primitive from the primitive permutation to extract the message bitstream by finding the position (index) p where the primitive is located in the remaining $n-i$ primitives of the reference ordering. If the number of primitive (leaf) nodes in the binary trie at the highest level, $e = ((n-i) - 2^k) \times 2$, at step i is larger than p, then p represents the extracted $k+1 = \lfloor log_2(n-i) \rfloor + 1$ bits. Otherwise, one of the primitives at level $\lceil log_2(n-i) \rceil - 1$ is the candidate for extracting next k bits. In this case, p is updated as $p - e/2$ to represent the extracted k bits.

3.2.2 Extraction by Modifying Vertex Position (MVP)

In this stage, we extract the embedded message from the vertices in the stego model. Actually, for each component x, y, and z, the index of the unit in the voxel where the vertex is located at is the integer value of the embedded message. Given the maximal embedding bits, k_i, the minimal vertex of the bounding volume of the cover model, p_{ref}, and the unit length, l_v^i, in the voxel, the embedding message m_i can be extracted by

$$m_i = \frac{v.i - p_{ref}.i - \lfloor \frac{\|v.i - p_{ref}.i\|}{2^{k_i} \times l_v^i} \rfloor \times (2^{k_i} \times l_v^i)}{l_v^i}, i = x, y, z.$$

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_v^i is set to 1×10^{-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, $\approx 2log_2 n!$, 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 *NDH* each for the testing model as shown in Table 1. As experiments show, the distortion is visually acceptable when the value of *NHD* is around 1×10^{-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.

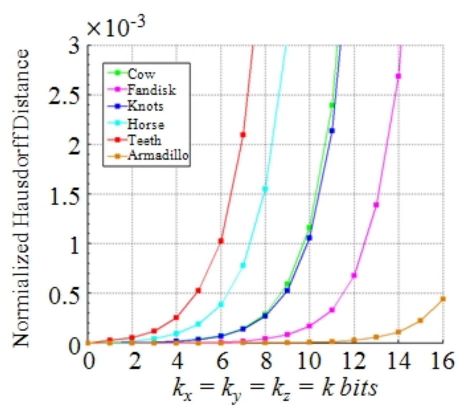


Figure 5: The average normalized Hausdorff distance as a function of embedding k bits in each component of the vertex by *MVP*.

Table 2 shows timing statistics in milliseconds of the embedding and extraction procedures of permutation steganography [1], and *MVP* for testing models respectively. The time complexity of *MVP*

Model			Capacity (bpv) [bits]				normalized Hausdorff distance
name	#verts	#faces	Tu et al. [1]	*MVP*	k	bpv	
Cow	2,904	5,804	92,918 (32.00)	87,120 (27.00)	9	57.00	5.92×10^{-4}
Fandisk	6,475	12,946	229,686 (35.47)	233,100 (36.00)	12	71.47	6.79×10^{-4}
Knots	23,232	46,464	952,798 (41.01)	627,264 (27.00)	9	68.01	5.26×10^{-4}
Horse	48,485	96,966	2,142,566 (44.19)	872,712 (18.00)	6	62.19	3.87×10^{-4}
Teeth	116,604	233,204	5,595,303 (47.99)	1,749,060 (15.00)	5	62.99	5.25×10^{-4}
Armadillo	172,974	345,944	8,597,908 (49.71)	8,302,752 (48.00)	16	97.71	4.42×10^{-4}

Table 1: The statistics of measured average capacity and normalized Hausdorff distance for testing models. The number in column k means that the model distortion is still visually unperceivable after k bits are embedded in each component of the vertex, meanwhile the normalized Hausdorff distance measured is shown in the last column for each model.

Model			Timings (msecs)				
			Embbeding		Extraction		Total
name	#verts	#faces	Tu et al. [1]	*MVP*	Tu et al. [1]	*MVP*	
cow	2,904	5,804	6	1	3	$\ll$1	10
fandisk	6,475	12,946	13	1	9	1	24
knots	23,232	46,464	56	4	39	2	101
horse	48,485	96,966	124	9	87	5	225
teeth	116,604	233,204	311	27	225	17	580
armadillo	172,974	345,944	481	40	341	24	886

Table 2: Timing statistics for proposed embedding and extraction procedures of *MVP* and permutation steganography [1], for testing models.

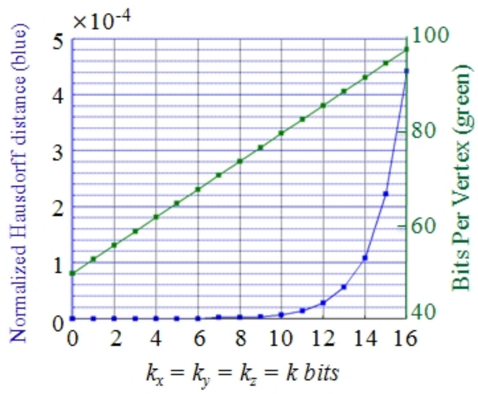


Figure 6: Adaptability illustration for Armadillo model. Given a preferred *NHD*, the maximal k bits to be embedded is adaptively determined.

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 $2log_2n!$. 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, $slog_2n!$, $s \geq 2$. *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

[1] Shih-Chun Tu, Wen-Kai Tai, Martin Isenburg, and Chin-Chen Chang, "An improved data hiding approach for polygon meshes," *The Visual Computer*, 2009, DOI: 10.1007/s00371-009-0398-1.

[2] R. Ohbuchi, H. Masuda, and M. Aono, "Watermaking three-dimensional polygonal models," in *MULTIMEDIA'97:Proceedings of the fifth ACM international conference on Multimedia*, 1997, pp. 261–272.

[3] Nicolas Aspert, E. Drelie, Y. Maret, and Touradj Ebrahimi, "Steganography for three-dimensional polygonal meshes," in *Proceedings of SPIE*, Nov. 2002, vol. 4790, pp. 211–219.

[4] Oliver Benedens, "Geometry-based watermarking of 3d models," *IEEE Computer Graphics and Applications*, vol. 19, no. 1, pp. 46–55, 1999.

[5] F. Cayre and B. Macq, "Data hiding on 3-d triangle meshes," *IEEE Transactions on Signal Processing*, vol. 51, no. 4, pp. 939–949, April 2003.

[6] Yu-Ming Cheng and Chung-Ming Wang, "A high-capacity steganographic approach for 3d polygonal meshes," *The Visual Computer*, vol. 22, no. 9, pp. 845–855, 2006.

[7] Yu-Ming Cheng and Chung-Ming Wang, "An adaptive steganographic algorithm for 3d polygonal meshes," *The Visual Computer*, vol. 23, no. 9-11, pp. 721–732, 2007.

[8] Jae-Won Cho, Rémy Prost, and Ho-Youl Jung, "An oblivious watermarking for 3-d polygonal meshes using distribution of vertex norms," *IEEE Transactions on Signal Processing*, vol. 55, no. 1, pp. 142–155, 2007.

[9] Y. Maret and T. Ebrahimi, "Data hiding on 3d polygonal meshes," in *MM&Sec'04: Proceedings of the ACM 2004 workshop on Multimedia and security*, 2004, pp. 68–74.

[10] Emil Praun, Hugues Hoppe, and Adam Finkelstein, "Robust mesh watermarking," in *Siggraph 1999, Computer Graphics Proceedings*, 1999, pp. 49–56.

[11] C.-M. Wang and Y.-M. Cheng, "An efficient information hiding algorithm for polygon models," *Computer Graphics Forum*, vol. 24, no. 3, pp. 591–600, 2005.

[12] C.-M. Wang and P.-C Wang, "Data hiding approach for point-sampled geometry," *IEICE Transactions on Communications*, vol. E88-B, no. 1, pp. 190–194, 2005.

[13] C.-M. Wang and P.-C Wang, "Steganography on point-sampled geometry," *Computers & Graphics*, vol. 30, no. 2, pp. 244–254, 2006.

[14] H.-T. Wu and Y.-M Cheung, "A reversible data hiding approach to mesh authentication," in *WI'05: the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, 2005, pp. 774–777.

[15] Z. Yu, H. S. IP Horace, and L. F. Kwok, "A robust watermarking scheme for 3d triangular mesh models," *Pattern Recognition*, vol. 36, no. 11, pp. 2603–2614, 2003.

[16] S. Zafeiriou, A. Tefas, and I. Pitas, "Blind robust watermarking schemes for copyright protection of 3d mesh objects," *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 5, pp. 596–607, 2005.

[17] D. Cotting, T. Weyrich, M. Pauly, and M. Gross, "Robust watermarking of polygonal polygonal meshes," in *SMI'04: International Conference on Shape Modeling and Applications*, 2004, pp. 233–242.

[18] S. Kanai, H. Date, and T. Kishinami, "Digital watermarking for 3d polygons using multiresolution wavelet decomposition," in *Proceedings of Sixth IFIP WG 5.2 GEO-6*, 1998, pp. 296–307.

[19] R. Ohbuchi, A. Mukaiyama, and S. Takahashi, "A frequency domain approach to watermarking 3d shapes," *Computer Graphics Forum*, vol. 21, no. 3, pp. 373–382, Sept. 2002.

[20] R. Ohbuchi, A. Mukaiyama, and S. Takahashi, "Watermarking a 3d shape model defined as a point set," in *CW'04:Third International Conference on Cyberworlds*, 2004, pp. 392–399.

[21] R. Ohbuchi, S. Takahashi, T. Miyazawa, and A. Mukaiyama, "Watermarking 3d polygonal meshes in the mesh spectral domain," in *GRIN'01:No description on Graphics interface 2001*, 2001, pp. 9–17.

[22] M. G. Wagner, "Robust watermarking of polygonal polygonal meshes," in *Proceedings of Geometric Modeling and Processing*, 2000, pp. 201–208.

[23] Min-Wen Chao, Chao-Hung Lin, Cheng-Wei Yu, and Tong-Yee Lee, "A high capacity 3d steganography algorithm," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 2, pp. 274–284, March 2009.

[24] Stephen Forrest, "Html steganography tool," http://wandership.ca/projects/deogol/intro.html.

[25] David Glaude, "Hiding data into the palette of a gif," http://users.skynet.be/glu/sgpo.htm.

[26] M. Kwan, "Gif colormap steganography," http://www.darkside.com.au/gifshuffle/.

[27] Donvan Artz, "Digital steganography: Hiding data within data," *IEEE Internet Computing*, vol. 5, no. 3, pp. 75–80, 2001.

[28] Alexander Bogomjakov, Craig Gotsman, and Martin Isenburg, "Distortion-free steganography for polygonal meshes," *Computer Graphics Forum*, vol. 27, no. 2, pp. 637–642, 2008.

[29] Jarek Rossignac, "Edgebreaker: Connectivity compression for triangle meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 5, no. 1, pp. 47–61, 1999.

[30] Ellis Horowitz, Sartaj Sahni, and Dinesh Mehta, *Fundamentals of Data Structures in C++*, Silicon Press, 2006.

[31] Normand Gregoire and Mikael Bouillot, "Hausdorff distance between convex polygons," http://cgm.cs.mcgill.ca/∼godfried/teaching/cg-projects/98/normand/main.html.

[32] Paolo Cignoni, Claudio Rocchini, and Roberto Scopigno, "Metro: measuring error on simplified surfaces," *Computer Graphics Forum*, vol. 17, no. 2, pp. 167–174, June 1998.

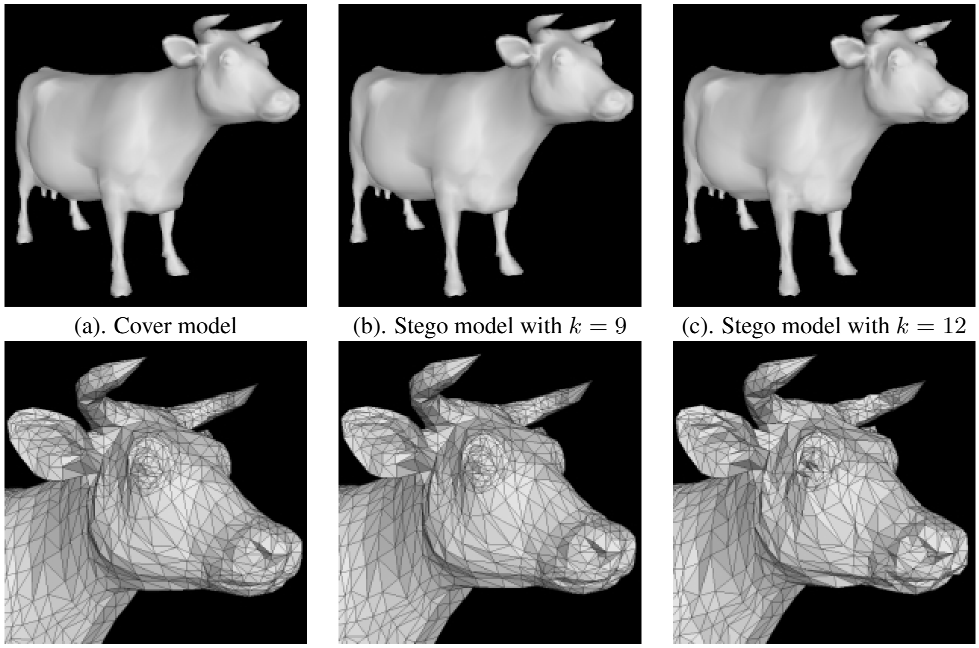
[33] Nicolas Aspert, Diego Santa-Cruz, and Touradj Ebrahimi, "Mesh: Measuring error between surfaces using the hausdorff distance," in *IEEE International Conference on Multimedia and Expo Proceedings*, 2002, pp. 705–708, http://mesh.berlios.de/.

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.

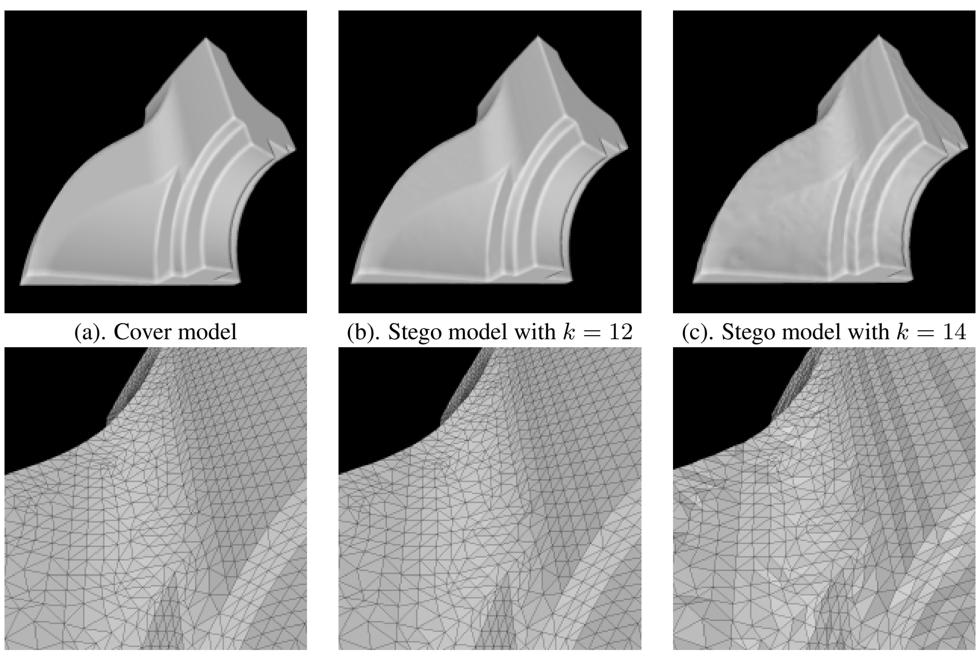
Wen-Kai Tai is currently an associate professor of Computer Science and Information Engineering department at National Dong Hwa University, Taiwan, R.O.C. His current research interests include data hiding, real-time rendering, visibility, and techniques for gaming.



(a). Cover model (b). Stego model with $k = 9$ (c). Stego model with $k = 12$

(d). Head close-up of (a) (e). Head close-up of (b) (f). Head close-up of (c)

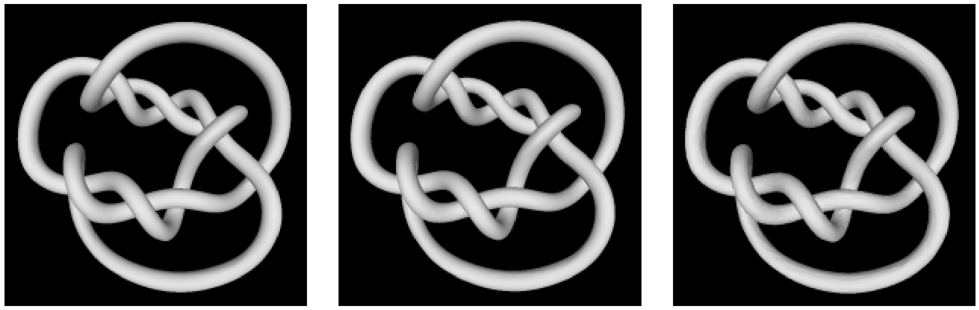
Figure 7: The visual perception for the cover model (a), Cow, its stego models for different value of k ((b) and (c)), and three close-up views (bottom) at head regarding to the figure on the top.



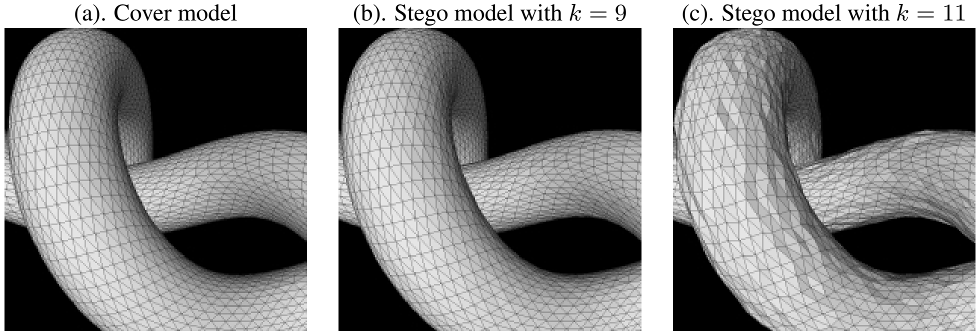
(a). Cover model (b). Stego model with $k = 12$ (c). Stego model with $k = 14$

(d). Middle close-up of (a) (e). Middle close-up of (b) (f). Middle close-up of (c)

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.

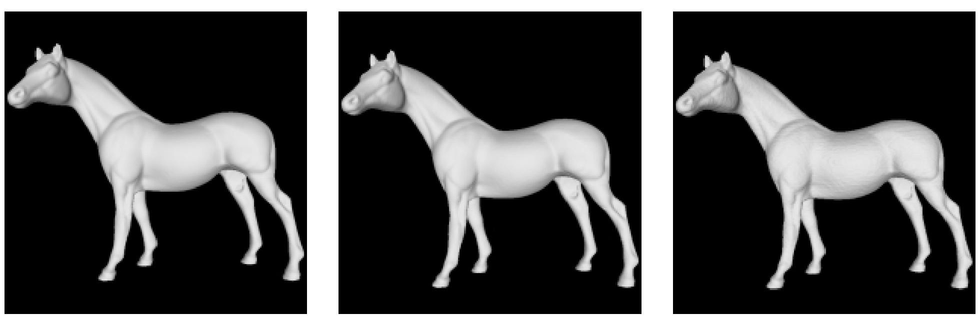


(a). Cover model (b). Stego model with $k = 9$ (c). Stego model with $k = 11$

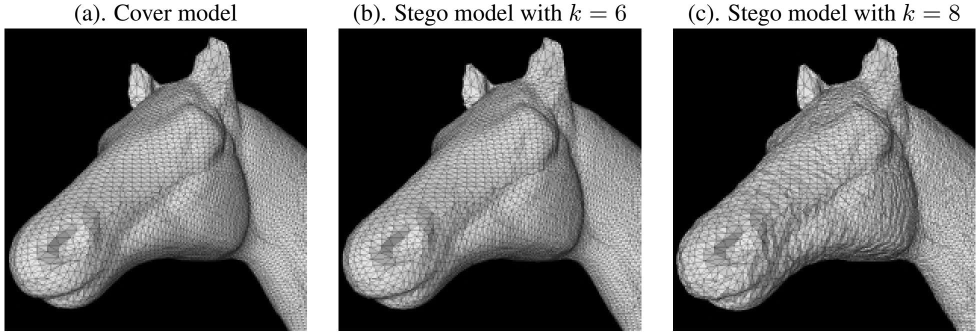


(d). Middle close-up of (a) (e). Middle close-up of (b) (f). Middle close-up of (c)

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.

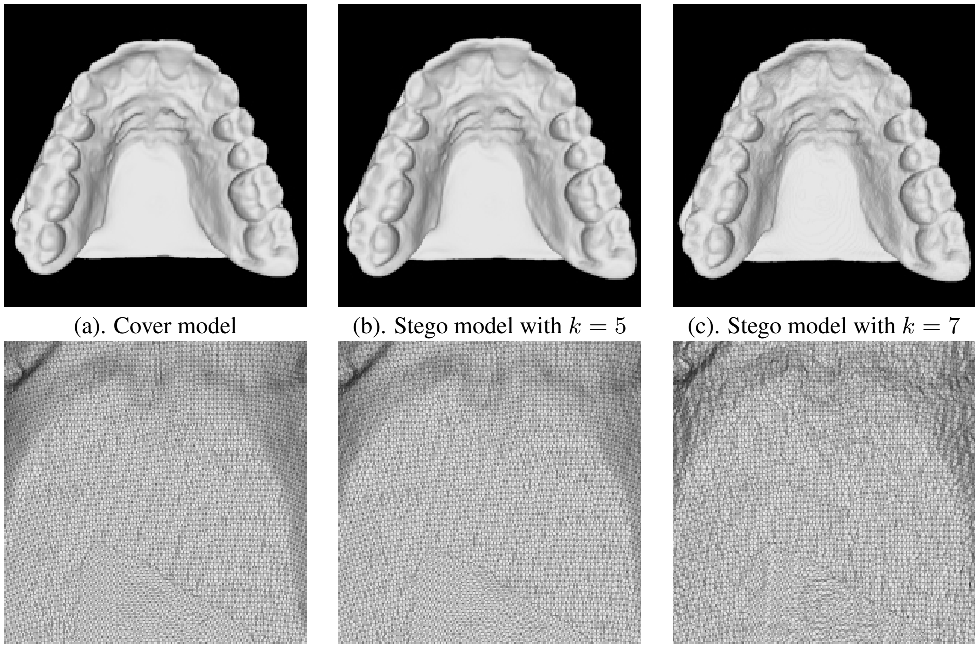


(a). Cover model (b). Stego model with $k = 6$ (c). Stego model with $k = 8$



(d). Head close-up of (a) (e). Head close-up of (b) (f). Head close-up of (c)

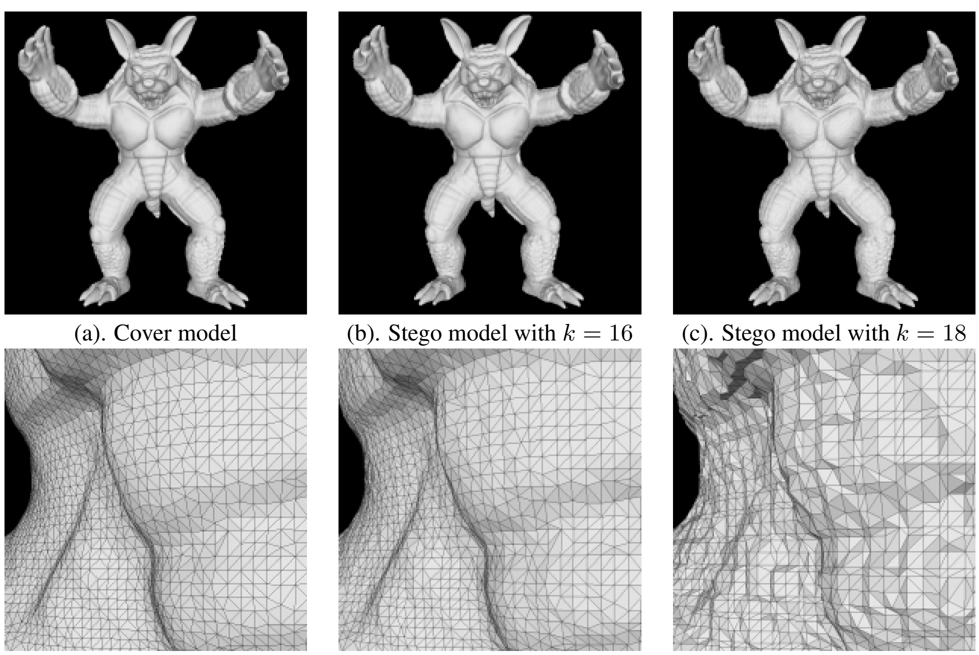
Figure 10: The visual perception for the cover model (a), Horse, its stego models for different value of k ((b) and (c)), and three close-up views (bottom) at head regarding to the figure on the top.



(a). Cover model (b). Stego model with $k = 5$ (c). Stego model with $k = 7$

(d). Middle close-up of (a) (e). Middle close-up of (b) (f). Middle close-up of (c)

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.



(a). Cover model (b). Stego model with $k = 16$ (c). Stego model with $k = 18$

(d). Chest close-up of (a) (e). Chest close-up of (b) (f). Chest close-up of (c)

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.