DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-396

SECURE HIGH CAPACITY DATA HIDING for 3D MESHES

V. Itier [#*1]***, A. G. Bors*** [†2]***, W. Puech*** [*3] ***, J.-P. Pedeboy*** [#4]

[*] ***LIRMM, UMR 5506 CNRS, University of Montpellier***
860 rue de St Priest, Bat. 5, 34095 Montpellier, FRANCE
[1] vincent.itier@lirmm.fr
[3] william.puech@lirmm.fr

[†] ***Dept. of Computer Science, University of York***
YORK YO10 5GH, UK
[2] adrian.bors@york.ac.uk

[#] ***STRATEGIES S.A, Parc d'Affaires Icade, Immeuble Amsterdam***
56 Rue dArcueil, 94578 Rungis, FRANCE
[4] jp.pedeboy@cadwin.com

Abstract

In this paper we present a high capacity data hiding method for 3D meshes. The proposed method is blind and orders selected vertices from the 3D object's mesh based on a pseudo random path. When a vertex is added to the path we embed a part of the message by displacing its location according to the location of another vertex called reference vertex. We solve the causality issue by removing certain vertices which would otherwise interfere with the path of selected vertices, during the message retrieval stage. Then we fill the resulting holes by remeshing. During the experiments, high bit capacity messages are embedded in several 3D objects. These messages are hidden under high levels of security while causing negligible surface distortions.

Introduction

3D objects are increasingly used in various areas of activity and their processing, analysis, compression, printing are continuously in the attention of research development. Adding securely meta-data to the 3D content is required in many applications. For example, it could be useful to add patient informations to its 3D medical data, or product information in the case of CAD (computer-aided design) objects. Such operations have to be done with a significant security level because of the data sensitivity. Moreover, the method should be blind and in the detection stage we should not have any information about the cover media when retrieving the hidden data. Generally, data hiding aims to ensure a trade-off between capacity, imperceptibility, robustness and security. Imperceptibility has to be enforced in order to preserve the user experience. In the literature of data hiding in 3D meshes, most authors focus on watermarking for copyright purpose, where robustness is enforced in order to preserve the ownership of 3D models. However, such information embedding methods have low capacity embedding [1, 2, 3]. Security is defined in [4] as the challenge in finding the secret parameters of the embedding function based on the observation of watermarked data. High capacity data hiding methods aim to increase the payload capacity. While watermarking needs to be robust and secure, high capacity data hiding has to focus on its security. The security of data hiding methods must be assessed under the Kerckhoffs principle [5], which assumes that the algorithm is known by the attacker. In order to retrieve the embedded data, we should have good synchronization between vertex paths at the embedding and retrieval stages. A desynchronization can occur if the mesh is attacked or if the embedding modifies the order of the embedded vertices in the path. This paper proposes a blind and secure high capacity data hiding method. The security is ensured by a new vertex synchronization approach using random jumps on the surface of the 3D object mesh. Each vertex is either a carrier, or a reference or is not used for data hiding. In this work, we use an embedding method based on [6], which consists of quantifying vertex coordinates.

The rest of the paper is organized as follows. In Section , we present an overview of the existing data hiding methods in 3D objects. The proposed method is introduced in the first section. In the second section, we provide the experimental results when hiding high capacity data into 3D objects. Finally, the last section concludes the paper and lists future objectives on how to continue this research.

High capacity data hiding in 3D graphics

When attempting to embed high capacity data in 3D models, we focus on the imperceptibility of the embedding while increasing the payload. Chao *et al.* [7] proposed an approach where 21 to 31 bits per vertex can be embedded in the 3D object. They order the vertices and embed the payload bits by displacing each vertex along x, y and z axes, after dividing the surface of the 3D object into regions. Their method produces nearly invisible distortion. Li *et al.* [8], have proposed a high capacity steganography method which preserves the surface of the 3D object. The algorithm proposed by Bogomjakov *et al.* [9] embeds a high capacity payload by changing the order of vertices in the object's file. This method does not modify the 3D mesh, and the message is included into the object's data file. However, this method is not robust to mesh file reshuffling. Gao *et al.* [10] proposed a semi-blind wa-

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-396

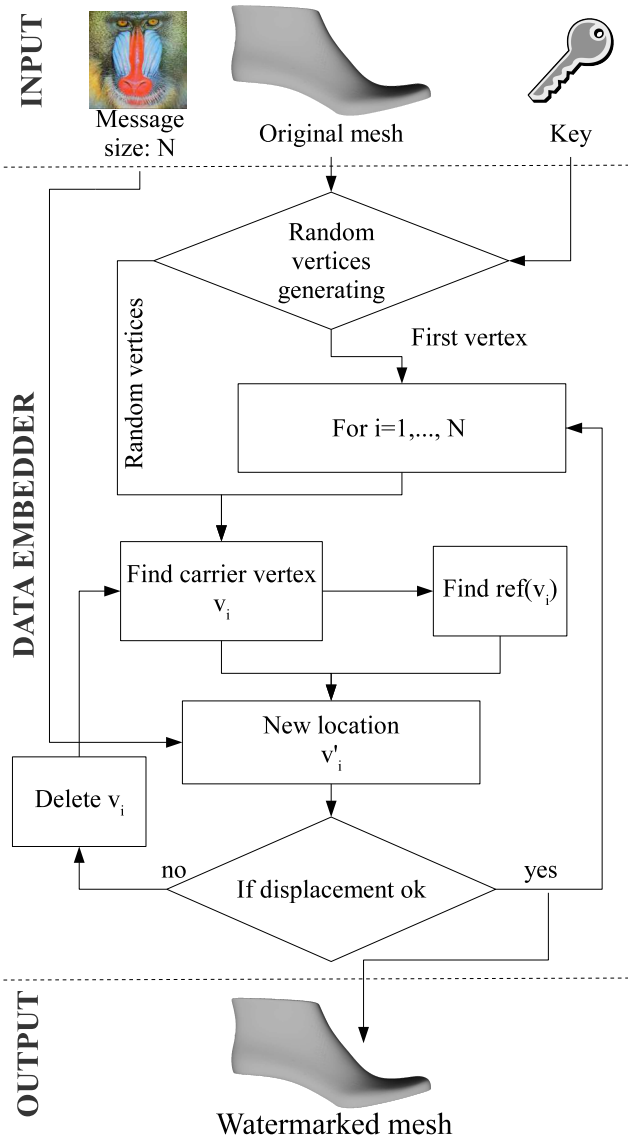


Figure 1: Diagram of the proposed data hiding method.

termarking which has a capacity of 1-2 bits per vertex. Watermark codes are inserted by modifying the length ratios extracted from a co-planar convex quadrilateral. As these length ratios are affine invariant, this method can withstand affine transformations. Multiple copies of the watermark are embedded, in order to be robust against cropping, which reduces the payload. In [11], the capacity is of 3 bits per vertex. The value of each coordinate is divided into n intervals of size Δ, which correspond alternatively to 0 and 1. The new value is the center of the closest interval which codes the bit to be inserted. Moreover this method generates errors, thus authors use error correction code. In conclusion there are data hiding methods that embed high capacity payloads into 3D objects, without significantly distorting their meshes, but such methods lack in security.

Selecting support vertices for data hiding

In this Section, we outline the main stages of the proposed blind 3D data hiding method. The synchronization consists of building a path to order the chosen vertices. Random jumps are considered, according to a secret key, when selecting vertices on the surface in order to ensure the security and to spread the message on the entire mesh. The embedding is done by moving a vertex v_i depending of the location of its reference vertex, which is defined as its closest vertex from the mesh. The payload message is embedded bit by bit, while synchronizing with mesh path selection in order to avoid the causality problem. The proposed data hiding method does not depend on the vertex connectivity, but only onto their actual location. Fig. 1, gives an overview of the proposed method for embedding a message, which can be an image for example. The loop corresponds to the iterative process which includes both the synchronization and the the payload embedding step. For each pixel of a colour image considered for embedding, the proposed method selects a carrier vertex according to a pseudo randomly generated vertex and its corresponding reference vertex. Then, if the displacement is allowed, the method proceeds with embedding the next bit from the payload. Otherwise, if the displacement is not permitted due to the causality problem, we would delete the current carrier vertex. Finally, when the entire message is embedded, holes in the surface of the 3D object caused by removing vertices, are covered by using remeshing.

Steps of the proposed method are described in following sections. First, we define the synchronization approach. Then, we present the embedding process. After that, we explain how to deal with the causality problem in the proposed data hiding approach, while in the next section we explain how to extract the data. In the last section, we discuss the security aspects of the proposed 3D data hiding method.

Synchronization

Firstly, a random sequence of vertices is chosen for embedding information, based on a sequence of secret keys. The location of the first vertex is chosen using a user specific key. Then, for the following vertices, a series of spheres, each with their center located in a reference vertex and of random radius, are generated and then again a random location is chosen on each of these spheres. The location on the sphere is selected using the Marsaglia method [12]. This method generates uniformly distributed points on a sphere using random values obtained using a randomly generated secret key. Each sphere is centered on the chosen reference vertex. We use a random variable as the radius of each sphere. The radius depends on the longest distance on the mesh D_{max}, and is calculated as:

$$r_i = k \cdot D_{max}, k \in [0,1]. \quad (1)$$

Using a small k, we would obtain a predictable path by choosing the closest vertex at each step, which is not that secure. The proposed synchronization allows us to distribute the message on the entire mesh.

Embedding

In this Section, we propose to embed a part of the message using a pair of vertices each time, aiming to increase the capacity while ensuring a high security. The coordinates of each vertex v_i are converted into spherical coordinates. A distance interval

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-396

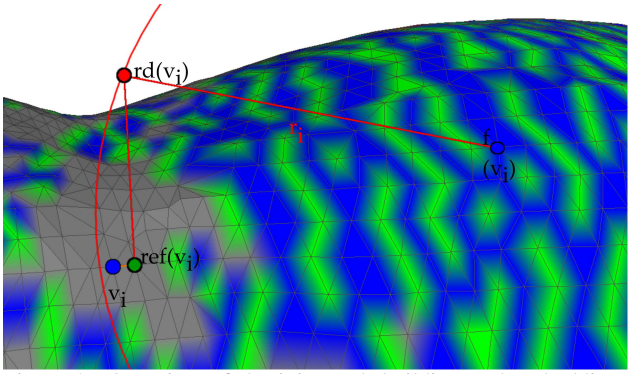


Figure 2: Overview of the joint path building and embedding; The formerly chosen vertex $f(v_i)$ is in blue. The generated random value $rd(v_i)$, chosen onto the surface of the sphere, centered at $f(v_i)$ and of random radius r_i is in red. The newly chosen payload carried vertex v_i is in blue and its reference vertex $ref(v_i)$ in green. With blue are carrier vertices, in green reference vertices and in grey vertices which are not used.

Δ is defined to bound the new location of each payload carrying vertex. In order to embed one byte per coordinate, the interval Δ is split into 256 subintervals. Thus three bytes per pixel, into each of the three spherical coordinates. The embedding is done at each stage of the path building, *i.e.* for each vertex, added to the path, we embed a part of the message. This process is iterative in order not to disturb a path that would have been built in previously. Due to the use of jumps, the distances between a vertex v_i and its predecessor $f(v_i)$ can be relatively large.

We propose to use a vertex as a reference for embedding in the current vertex v_i and we denote it $ref(v_i)$. The reference vertex is defined as the closest vertex to v_i which has not been previously selected in the path. This vertex is used for finding the new location v_i' of v_i, as reference for the conversion into spherical coordinates. To insert the value $n \in [0,255]$, into one of the coordinates of vertex $v_i(c_1,c_2,c_3)$, we use the following mapping function:

$$c_1' = \left\lfloor \frac{c_1}{\Delta} \right\rfloor \Delta + n\frac{\Delta}{256}. \quad (2)$$

We define three different labels for the vertices from the 3D object: carriers which are used to embed data, references, and vertices which are not used. A reference vertex can be used by several carrier vertices. The iterative embedding is illustrated in Fig. 2. The random vertex $rd(v_i)$ is used to find the vertex v_i to be added to the path. Its predecessor in the path is denoted $f(v_i)$, which is the reference of the last displaced vertex. The current vertex v_i is displaced with a ratio from its distance to the reference vertex, $ref(v_i)$. By using reference vertices allows us to have low distortion and higher security. The data hiding capacity is not that much affected because the same vertex reference can be used for several information embedding vertices. Actually, according to our experimental results, we use the same reference for three carrier vertices in average.

The message length is limited by the capacity of the vertex which is three bytes, *i.e.* $|m| \in [0,2^{24}]$ allowing to embed a color image pixel. Moreover, the proposed methodology allows embedding data into objects with high density of vertices, thus significantly increasing the payload embedded into such objects.

Solving the vertex causality problem in the embedding path

When embedding data and slightly displacing vertices, we may affect the order of the previously chosen vertices from the selected path, *i.e.* the causality of the embedded strings of information. This is an important issue because in such situations we would not be able to retrieve the embedded information. One of the following situations may arise:

1. Current vertex v_i: should not be moved too far neither from its corresponding generated random value rd_{v_i}, nor from its reference $ref(v_i)$.
2. Path: we have to ensure that the current vertex v_i should not have been already chosen at a previous step. This is the reason why all previously marked vertices as well as their reference vertices are labeled.

The path of chosen marked vertices must be the same for both embedding and retrieval stages of the data hiding procedure. We can choose to move all vertices, but that will cause a desynchronization and at the retrieval stage we would not be able to retrieve the embedded information. On the other hand, we can choose not to move certain vertices, which would interfere with the path of previously chosen payload vertices, and in this case the corresponding data to be embedded would be accounted as bit-loss at the retrieval stage. Such situations depend on the 3D model and on its vertex density distribution. Nevertheless, a small number of errors can significantly change the existing vertex path and thus the synchronization. Our strategy to overcome this, is to remove the vertex and fill the resulting hole. Generally, we can consider, in dense and large graphical objects, that the vertex neighborhood is a flat surface and thus removing a vertex does not introduce significant distortions. Then, we fill the hole left in the mesh, by splitting it into simple triangular faces, while reducing the surface error. In order to minimize the object surface distortion, the edges filling the hole are placed as close as possible to the location of removed vertices.

Message extraction

We should know the secret key in order to extract the correct message. The order of the vertices is retrieved by performing the same processing steps as in the embedding stage. Then, for each pair of carrier vertices and their reference vertices, we compute the spherical coordinates. Then, we extract the information from the vertices where it was embedded, as:

$$x = 256(c_j - \left\lfloor \frac{c_j}{\Delta} \right\rfloor \Delta). \quad (3)$$

After firstly retrieving the message length, we extract the embedded information bit by bit from the succession of chosen vertices.

Security analysis of the approach

In this section we analyse the security of the proposed 3D model data hiding approach. Firstly, assuming the Kerckhoffs principle, the security relies on the secrecy of the key. The secret key is used as a seed for the random generation of the vertex path list. The security keys ensure that an attacker would not be able to retrieve the sequence of vertices in their embedding order. In fact, there exists $n!$ different paths. The spreading paramater, *i.e.*

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-396

k in equation (1), reduces the number of different path as function of his range. Nevertheless, the number of combination has a very high computational cost. Moreover, an attacker could not properly classify the vertices as either carrier, reference or not-used vertex. We assume different attack scenarios such as those proposed by Perez-Freire and Perez-Gonzalez [4]: known message attack (KMA), constant message attack (CMA) and watermark only attack (WOA). The attacker, when having the original object, can attempt to do a quantization attack. However, in order to find the hidden message, it would have to know the order of the displaced vertices. If the attacker would know the vertices which are displaced he can guess their reference vertices by finding their nearest neighbor. Then he can do a quantization attack to find the value embedded into each vertex. The difficulty consists of synchronizing the values extracted. Nevertheless, in the most relevant security breach scenario, defined by WOA, an attacker cannot actually find where the message is embedded.

Experimental results

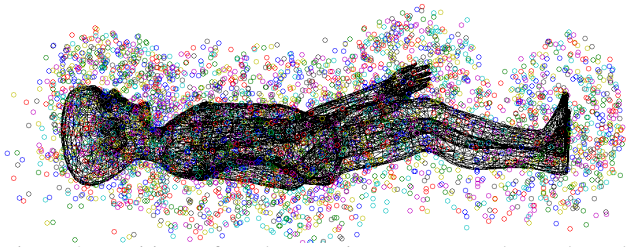


Figure 3: Positions of random vertices, represented as coloured circles, generated for the Alien mesh with 7401 vertices.

Results are produced using a database of 3D meshes which have various shapes and sizes. For our experimentations, we use float representation *i.e* 32 bits per coordinate in order to have a huge capacity. The meshes of 3D objects are firstly normalized, in order to make the mean edge size equal to 1. Thus, the parameter Δ is set to 10^{-4} in equation (2), which is, experimentally, a good trade-off between the imperceptibility and the data capacity to be embedded into each vertex. We use random seeding for the Marsaglia algorithm [12]. Fig. 3 illustrates the position of the random vertices, represented as coloured circles, around the mesh of the 3D object named "Alien". In the following we embed colour images into the 3D object because we can embed 3 bytes into each selected vertex from the 3D mesh. Images are two dimensional, so we have to embed the height and the width of the image in the first two vertices of the path.

In order to select securely vertices where we hide data, we generate a sequence of spheres. The radius of such spheres is chosen as a randomly generated number from a uniform distribution. In the plot from Fig. 4 we analyze the impact of the radius length on the number of vertices to be deleted because otherwise they would interfere with the causality of the embedded bytes. The radius length is chosen as a percentage $k \in [0.1, 0.9]$ of the maximal length in the mesh D_{max}, according to equation (1). In the following, we evaluate the influence of the radius of the generated spheres on the number of vertices which are removed, for ten meshes of various shapes and sizes. We standardize the results by calculating the average of the vertices removed, calculating the difference to that and dividing by the standard deviation, calculated for this set of shapes. Fig. 4, illustrates the average of standardized values, obtained for all meshes. We can see that using a small radius leads to few deleted vertices. According to this study, we found that a range of $k \in [0.15, 0.35]$ is optimal for choosing the radius of the sphere.

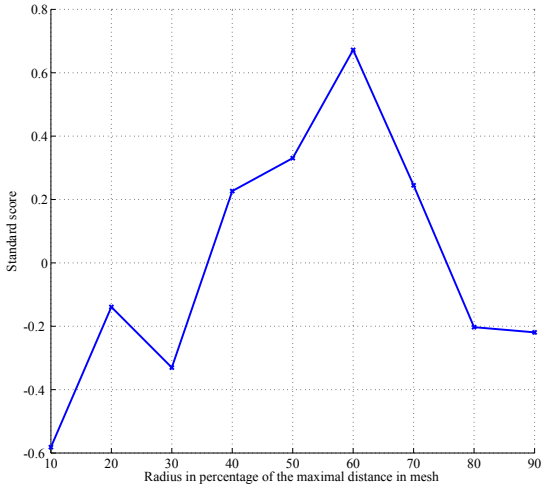
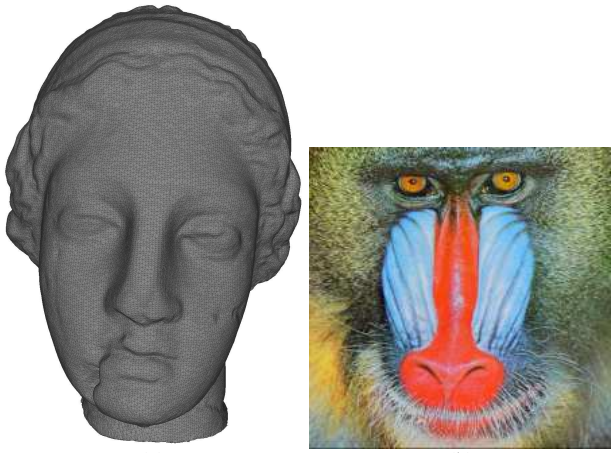


Figure 4: The average of the standardized percentage of deleted vertices depending on the radius of generated spheres, calculated as the percentage of the maximal distance of the mesh, D_{max}.



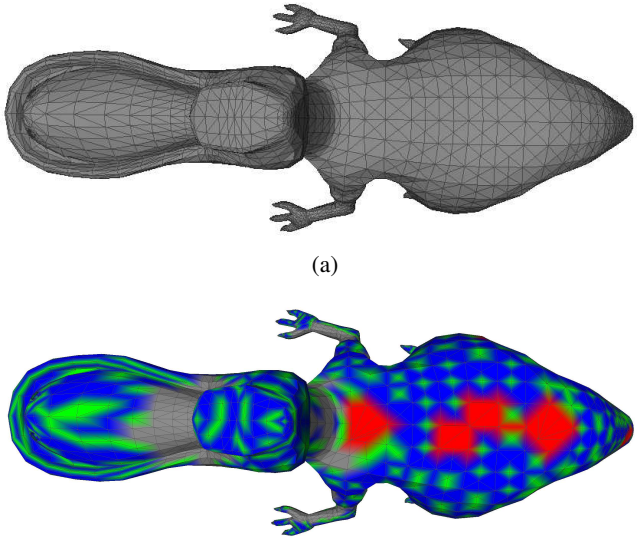
(a) (b)

Figure 5: a) Original mesh with 100759 vertices, b) Embedded image 224×224 pixels: 1204224 bits.

We provide an example when embedding the colour image "Baboon" shown in Fig. 5.b into the mesh of the graphical object "Venus", shown in Fig. 5.a. For our experiments, we set the size of the message to be half of all vertices from the graphical object. For this example the capacity is 11.95 bits per vertex, but 15637 vertices are not used and there are 0.69 references per vertex in average. To reach the optimal capacity of the mesh, we can embed a message as long as we have unused carrier vertices.

Fig. 6, presents the result when embedding a message in the mesh of Dinopet object. In Fig. 6.a, we show the original Dinopet mesh while the watermarked object is provided in Fig. 6.b. We can see that the deleted vertices lie mostly in the regular meshed

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-396



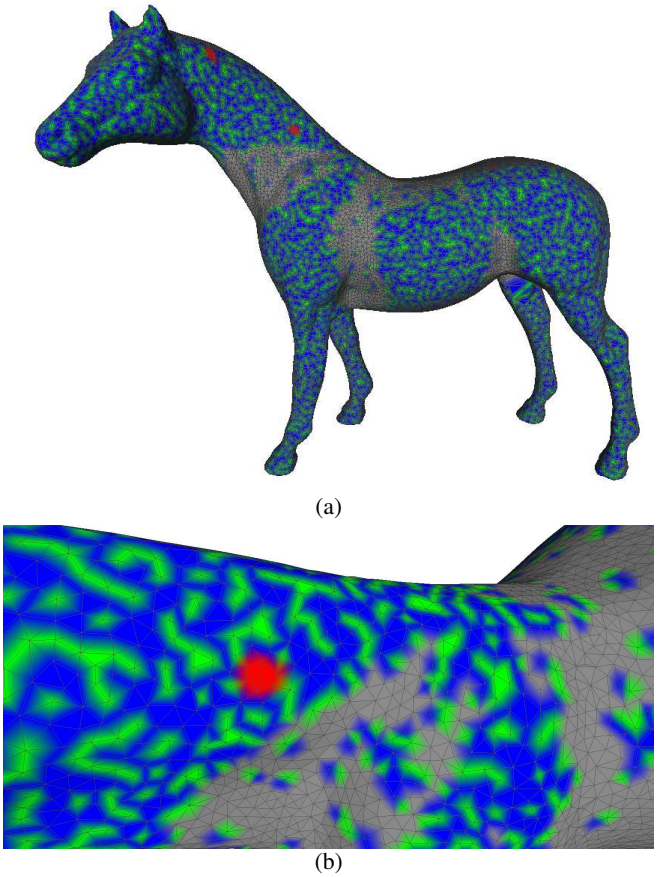
(a) (b)

Figure 6: a) Original mesh with 4500 vertices, b) Watermarked mesh with 4493 vertices, in blue: the carrier vertices, in green: reference vertices, in red: holes corresponding to removed vertices which are then remeshed and in grey: unchanged vertices.

areas of the object. The mesh is relatively small, so the filling of the holes resulted from the removal of vertices is sometimes visible and does not always look natural. However, on larger and not so regular meshes, the filling is almost imperceptible. It can be seen from the watermarked 3D model of the horse from Fig. 7.a, that following the filling of holes, the resulting distortions are imperceptible. Moreover, in the details of the mesh surface from Fig. 7.b, we notice that the filling is smooth and does not distort the surface.

In order to evaluate the proposed data hidding algorithm we use two quality assessment metrics representing the Hausdorff distance, and the MSDM2 distance [13]. Contrary to the Hausdorff distance, the MSDM2 distance is more correlated with human perception and its value is between 0 and 1, where 0 is for no distortion while a value close to 1 represents a high distortion. Table 1 presents the distortion results on several meshes. We do not consider meshes with large flat regular areas, because the proposed method could introduce distortions in such objects. However, the meshes of such graphical objects can be used for data hiding after undergoing mesh simplification operations.

In Table 1, we provide the results when hiding data into a set of ten 3D objects, each characterized by a variety of shape properties. The Hausdorff distance seems to correlate with the object shape and its size. For example in small meshes with many curvatures, like the mesh of "Dinopet" object, deleting a vertex implies the flattening of the surface. When embedding data into the mesh of "Casting" object we have many deleted vertices because of the regularity usually found in CAD objects. It has low distortions, because of having rather flat areas, and removing a vertex has almost no effect. Contrary, deleting a vertex on a edge of this kind of object, could produce visual distortions, which is shown by the MSDM2 score. Generally, these results show that the visual human perception of the distortions resulting from remeshing holes



(a) (b)

Figure 7: a) Watermarked mesh with 19995 vertices, b) Close-up look to the mesh surface.

of removed vertices, following data hiding into 3D objects, is not that significant, according to MSMD2 measure.

While other high capacity methods do not provide security, the proposed scheme has, in addition, a huge capacity and low distortions. We compare the proposed method with high capacity methods [7, 10, 11] in terms of capacity, Hausdorff distance and $PSNR_1$ [7]. The $PSNR_1$ is defined as: $20log_{10}\left(\frac{dmax}{RMSE}\right)$, where $dmax$ is the diagonal length of the object bounding box and $RMSE$ is the root mean square error. Results obtained on the mesh of "Bunny" object are presented in Table 2. We choose $\Delta = 1.10^{-4}$ because it is a good trade-off and its impact has been studied in [11]. Empty cells correspond to results which are not given by authors.

We can see that our method produces more distortions than other methods because of the deleting of vertices. Nevertheless, distortions are still low and perceptual metric assesses that they are not visible when information is embedded into large meshes. The proposed method provides better security than others while embedding large data.

Conclusion

In this paper, we presented a new high capacity data hiding for 3D objects, using only their vertex location and not their connectivity. Vertices which produce path causality problems are removed from the mesh, assuming that this would not cause sig-

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-396

Table 1: Distortion results.

3D Object	Number of 3D object vertices	Number of deleted vertices	Hausdorff distance $\times 10^{-3}$	MSDM2 $\times 10^{-2}$
Bitorus	3000	28 (0.930%)	1.581	9.69
Dinopet	4500	7 (0.160%)	4.659	8.10
Casting	5096	51 (1.0%)	0.272	14.51
Horse	20000	9 (0.045%)	1.347	2.59
Blade	24738	4 (0.016%)	1.550	2.23
Bunny	34834	44 (0.126%)	1.796	6.844
Shoe	45002	12 (0.027%)	0.152	1.28
Rabbit	70658	47 (0.067%)	0.816	1.07
Shoe2	83698	149 (0.178%)	0.429	12.77
Venus	100759	64 (0.064%)	0.800	4.21

Table 2: Comparison to previous methods on Bunny model.

Method	Capacity	Hausdorff Distance $\times 10^{-6}$	$PSNR_1$
[7]	940464		100.57
[10]	51408	548	
[11]	54289	1	127.3
Proposed method $\Delta = 1.10^{-4}$	411864	1796	82.55

nificant shape distortions. This method offers data embedding capacity of around 12 bits per vertex, while producing very low distortion. The proposed data hiding method provides an increased security due to the use of a secret key, used for choosing the initial vertex, the hopping sphere radius and the location on this sphere. Some vertices from the 3D model are used as references. Few vertices, which interfere with the causality of the selected vertices path, are removed from the surface of the 3D object. The resulting holes in the 3D shape are filled in by remeshing. In the future we will study better methods for developing new secure ways of selecting vertex sequences for data hiding while minimizing the resulting surface errors.

References

[1] K. Wang, G. Lavoué, F. Denis, A. Baskurt, Robust and blind mesh watermarking based on volume moments, Computer Graphics, 35, pg. 1-19 (2011).

[2] A. G. Bors, M. Luo, Optimized 3D Watermarking for Minimal Surface Distortion, IEEE Trans. on Image Proc., 22, pg. 1822-1835 (2013).

[3] X. Rolland-Neviere, G. Doerr, P. Alliez, Triangle Surface Mesh Watermarking Based on a Constrained Optimization Framework, IEEE Trans. on Information Forensics and Security, 9, pg. 1491-1501. (2014).

[4] L. Perez-Freire, F. Perez-Gonzalez, Spread Spectrum Watermark Security, IEEE Trans. on Information Forensics and Security, 4, pg. 2-24. (2009).

[5] A. Kerckhoffs, La cryptographie militaire, Journal des sciences militaires, IX, pg. 5-38. (1883).

[6] B. Chen, G. W. Wornell, Quantization Index Modulation Methods for Digital Watermarking and Information Embedding of Multimedia, J. of VLSI signal processing systems for signal, image and video technology, 27, pg. 7-33. (2001).

[7] M.-W. Chao, C.-H. Lin, C.-W. Yu, T.-Y. Lee, A High Capacity 3D Steganography Algorithm, IEEE Trans. on Visualization and Computer Graphics, 15, pg. 274-284. (2009).

[8] M. T. Li, N. C. Huang, C. M. Wang, A novel high capacity 3D steganographic algorithm, J. of Innovative Computing, Information and Control, 7, pg. 1055-1074. (2011).

[9] "A. Bogomjakov, C. Gotsman, M. Isenburg, Distortion-Free Steganography for Polygonal Meshes, Computer Graphics Forum, 27, pg. 637-642. (2008).

[10] X. Gao, C. Zhang, Y. Huang, Z. Deng, A Robust High-capacity Affine-transformation-invariant Scheme for Watermarking 3D Geometric Models, ACM Trans. on Multimedia Computing, Communications and Applications, 8, pg. 34:1-34:21. (2012).

[11] V. Itier, W. Puech, G. Gesquière, J.-P. Pedeboy, Joint synchronization and high capacity data hiding for 3D meshes, Proc. SPIE, 9393, pg. 05-15. (2015).

[12] G. Marsaglia, Choosing a Point from the Surface of a Sphere, The Annals of Mathematical Statistics, pg. 645-646. (1972).

[13] G. Lavoué, A Multiscale Metric for 3D Mesh Visual Quality Assessment, Computer Graphics Forum, 30, pg. 1427-1437. (2011).

Author Biography

Vincent ITIER received the M.S degree in Computer Science from the University of Montpellier, France, in 2012 and the Ph.D. degree in Computer Science from the University of Montpellier, France, in 2015. Currently, is is teaching assistant at the University of Montpellier, France. His work has focused on the development of 3D mesh data-hiding applications.

Adrian G. BORS received the M.S. degree in Electronics Engineering from the Polytechnic University of Bucharest, Romania, in 1992, and the Ph.D. degree in Informatics from the University of Thessaloniki, Greece in 1999. During 1992-1993 he was a research scientist at the Signal Processing Laboratory, Tampere University of Technology, Finland. In 1999 he joined the Department of Computer Science, University of York, U.K., where he is currently a lecturer. During 2006, Dr. Bors was a Visiting Scholar at the University of California at San Diego (UCSD), USA, and an Invited Professor at the University of Montpellier II, France. Dr. Bors has authored and coauthored more than 100 research papers including 21 in journals. Dr. Bors was an associate editor of IEEE Trans. on Image Processing during 2010-2015 and for IEEE Trans. on Neural Networks from 2001 to 2009. He is a Senior Member of the IEEE since 2004.

William PUECH received the diploma of Electrical Engineering from the University of Montpellier, France, in 1991 and the Ph.D. De-

DOI: 10.2352/ISSN.2470-1173.2016.21.3DIPM-396

gree in Signal-Image-Speech from the Polytechnic National Institute of Grenoble, France in 1997. He started his research activities in image processing and computer vision. He served as a Visiting Research Associate to the University of Thessaloniki, Greece. From 1997 to 2000, he had been an Assistant Professor in the University of Toulon, France, with research interests including methods of active contours applied to medical images sequences. Between 2000 and 2008, he had been Associate Professor and since 2009, he is full Professor in image processing at the University of Montpellier, France. He works in the LIRMM Laboratory (Laboratory of Computer Science, Robotic and Microelectronic of Montpellier). His current interests are in the areas of protection of visual data (image, video and 3D object) for safe transfer by combining watermarking, data hiding, compression and cryptography. He has applications on medical images, cultural heritage and video surveillance. He is the head of the ICAR team (Image & Interaction) and he has published more than 15 journal papers, 8 book chapters and more than 80 conference papers. W. Puech is associate editor of J. of Advances in Signal Processing, Springer, Signal Processing: Image Communications, Elsevier and Signal Processing, Elsevier and he is reviewer for more than 15 journals (IEEE Trans. on Image Processing, IEEE Trans. on Multimedia, IEEE TCSVT, IEEE TIFS, Signal Processing: Image Communication, Multimedia Tools and Applications ...) and for more than 10 conferences (IEEE ICIP, EUSIPCO, ...).

Jean-Pierre PEDEBOY is the head of STRATEGIES Company since 30 years. He is an engineer and he is very well known in 3D modeling of manufactured objects.