An High-capacity Steganographic Scheme for 3D Point Cloud Models Using Self-similarity Partition

QI Ke, Xie Dong-qing

Abstract—**This paper presents a new steganographic scheme for 3D point cloud models using self-similarity partition. The new scheme partitions the 3D point cloud model to patches using self-similarity measures, and generates the codebook. The representative patches and similar patches are then taken from the codebook for every similar patch chain as the reference patches and the message patches. Finally, every message point in the similar message patches which has the point-to-point correspondence with a certain reference point in the reference patch can embed at least three bits by shifting the message point from current point to the corresponding embedding position using space subdivision. Experimental results show that the proposed technique is secure, has high capacity and low distortion, and is robust against uniform affine transformations such as transformation, rotation, scaling. In addition, a concise shape description and a similarity measures are devoted to improving performance for forming codebook and constructing point traversal. The technique can be considered as a side-match steganography and has proven to be a feasible alternative to other steganographic schemes for 3D point cloud model.**

Index Terms—**steganography, 3D point cloud models, self-similarity patch chain, spatial domain**

I. Introduction

Steganography is an art of communicating in a way that hides the existence of the communication. Compared with classical watermarking, which is a process for protecting copyright ownership, steganography is a technique about concealing the existence of the secret messages. Therefore, steganography tends to require higher data capacity, low distortion and security, but it can lead to relatively poor robustness.

With the development of 3D applications and animation, many steganographic algorithms have been presented for 3D models. While 3D models are usually classified into mesh model and point cloud model, in which 3D mesh model normally consists of vertexes, edges and polygonal surfaces while point sample model consists of 3D geometry i.e. point information only, there have been comparatively many steganographic algorithms for 3D mesh model [1, 2] which always use topology, edge length or angle among meshes to hide information. However, since 3D point cloud model consists only point information without any edges and surfaces which are basic elements of information hiding for 3D mesh models, almost all steganography for 3D mesh models can not be applied to 3D point cloud models, and only fewer data hiding approaches [3, 4, 5] for 3D point cloud model have been presented until now. Finding a way to fully exploit the features of 3D point cloud models is an important research issue.

Cheng et al. [1] uses the Virtual Multi-Level Embed Procedure to embed three bits per point based on shifting the message point by its virtual sliding, extending, and arching geometry property. As a result, they exploited high embedding capacity in the 3D space, which seems to be the source for the maximal capacity on 3D point cloud model.

Cotting et al. [3] proposed a watermarking algorithm of point sampled geometry based on pseudo-spectral analysis. The algorithm partitioned the model into a set of patches by applying a fast hierarchical clustering scheme. Next, each patch was mapped into the space of eigenfuncitons of a Laplacian operator to obtain discrete frequency bands. Finally, the messages were embedded into the low frequency components. The algorithm is suitable for watermarking since it has high robustness but with low capacity.

Wang et al. [4] presented a data hiding scheme for point models. The scheme used a principal component analysis (PCA) and symmetrical swap procedure to embed messages. The algorithm suffered a capacity drawback that the data capacity in bits generally achieved only about half of the number of points in the model.

Luo et al. [5] presented a reversible data hiding for 3D point cloud model. It started with creating a set of 8-neighbor vertices clustered set with randomly selected seed vertices. Next, an 8-neighbor integer DCT was performed to obtain coefficient. Finally, a highest frequency coefficient modification technique was employed to embed messages. The scheme has characteristic of reversibility but of very low capacity.

All of the above-mentioned steganography for 3D point cloud model can be categorized into transform domain and spatial domain. Approaches in transform domain [3-4] normally have high robustness but very low capacity, while approaches in spatial domain [1, 5] have comparatively high capacity but low robustness. Research presented by Cheng [1] seems to be the source for the maximal capacity on 3D point cloud model. However, his scheme normally embeds only about multiples of three bits per embedding point.

Since steganography tends to require high capacity and low robustness, we prefer to develop a blind scheme in

Manuscript received March 5, 2011. This work was supported in part by the National High Technology Research and Development Program of China (Grant no. 2009AA012420) and Natural Science Fund of Guangdong Province (Grant no. 9151009001000059), China.

Qi ke is with the School of Computer Science and Education Software, Guangzhou University, Guangzhou 510006, China (phone:86-020-39366375; fax: 86-020-39366375; e-mail: qikersa@163.com)

Xie Dong-qing is with the School of Computer Science and Education Software, Guangzhou University, Guangzhou 510006, China (e-mail: dqxie@hnu.cn)

spatial domain from the high capacity point of view. This paper presents a new blind high-capacity steganography for 3D point cloud model, inspired by the concepts proposed by [6]. The key idea is to construct codebook for self-similarity partitioned patch of the 3D point cloud model using self-similarity measures, and exploit space subdivision to embed information by shifting the message point to a certain spatial position which is computed from the point-to-point corresponding reference point. To the best of our knowledge, this is the first 3D point cloud model steganographic scheme that uses the reference point matching relationship to embed messages which can be considered as a side-match steganogrpahy. This procedure also efficiently achieves high capacity of around three bits per point with little visual distortion. Furthermore, patch size, codebook size, the traversal list of codebook and embedding list over each message patch are used as secret keys for more security. Similar to previous 3D steganographic methods, the scheme is robust against affine transformations, which include translation, rotation, uniform scaling, or their combined operations.

The rest of the paper is organized as follows. Section Ⅱ introduces construction of similarity patch chains and codebook. Section Ⅲ describes the details of the proposed steganographic scheme including the kernel method of self-similarity position matching procedure for steganography. The experimental results are given in Section Ⅳ and a brief conclusion of the paper is made in Section Ⅴ.

II. Construction Of Similarity Patch Chains And Codebook

As an important technique of 3D model compression, self-similarity based compression of point set surfaces [6] is a promising compression technology for massive point sets and ray tracing. For many 3D point models consist commonly of massive point sets with repetitive patterns and similar structures, such as creases, ridges, bumps, compression can be completed by replacing similar surface patches with an instance of a representative patch with efficient encoding and decoding.

Self-similarity patch chain is generally defined as the set of segments scattered around 3D point model with similar shapes or structures that can be identified by pre-defined shape descriptors and similarity measures (see Fig.1). As every patch is similar to the other patches in the same self-similarity clustered chain, every pair of the matching points in the pairs of similar patches can embed information using extended QIM (see Section Ⅲ).

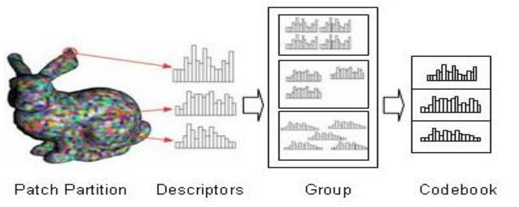


Fig.1 Patch partition and construction of self-similarity patch chains and codebook

In our proposed stegonagraphic scheme, unlike the conventional compression purpose of Hubo [6], similar patch chains are efficiently constructed and the spatial position of the message point is shifted to embed information. Owing to their favorable similarity characteristics, these pairs of matching points are better for a 3D point cloud model to design the stegonagraphic scheme.

The construction of self-similarity patch chain can be decomposed in three steps: (1) patch partition with an octree; (2) measuring similarity among patches; and (3) codebook generation for self-similarity patch chains.

A. Patch partition

Clustering-based partition uses iterative clustering as a tool to separate the input model into multiple regions according to local properties of points. particle simulation [7], and octree [8] are partition algorithms.

For the purpose of meaningful self-similarity comparison and improving the likelihood of finding a similar match, the patch partition favors patches that are equally sized and lie on salient features [6]. To achieve this goal, we use the local surface reconstruction technique [9] which choose the octree among possible candidates for its hierarchical multi-resolution structure with equally sized cells of regular shapes, and for the efficiently geometrical approximation of the local point cloud in a cell.

As proposed by Tamy [9], let P be the original 3D point cloud model. We begin with C, the bounding cube of P. The initial point set P is uniformly splitted into local point set P_i in a sub-cube C_i according to the cell boundaries, and we run again the uniform partition along the axes until the following conditions are met.

$$\begin{aligned} &\forall j \in [0, k_i - 1],\ n_{ij} \cdot n_i > \delta_a, \quad \delta_a \in [0,1] \\ &\frac{\left|(p_{ij} - o_i) \cdot n_i\right|}{d/2} < \delta_b, \quad \delta_b \in [0,1] \qquad (1) \\ &k_i < K \end{aligned}$$

where p_{ij} be the j th point of the partition P_i, n_{ij} be the local normal vector of the point p_{ij}, o_i be the local centroid of the sub-cube C_i, n_i be the average local normal vector of the partition P_i, k_i be the number of points in the sub-cube C_i, d be the edge length of the sub-cube C_i, the threshold value K be the patch size representing the max number of points in the sub-cube C_i, the parameter δ_a represents the maximal deviation angle that the normal of a point can make according to the average normal of its current sub-cube, the parameter δ_b represents the distance between a point and its projection onto the local average plane defined by o_i and n_i. We have set δ_a to 0.2 and δ_b to 0.3 in our experiments.

The unique hierarchical octree structure for the original 3D point model is then constructed using above partition algorithm in which every nonempty node of bottom layer represents a partitioned patch with no more than K points. The number of octree subdivision is saved for extraction.

B. Measuring similarity among patches

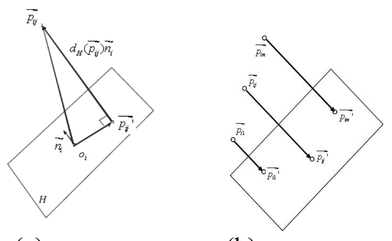
Before we can group similar patches, we need a way to measure how similar they are. Since the patches are partitioned into sub-cells with same size and lie on salient features, the measuring is rather straightforward in two steps:

1) Alignment.

Since patches are usually close to flat, they all can be oriented such that their supporting planes coincide. The key idea is to find supporting planes and align these planes. For efficiency reasons and inspired by Cheng [10] which uses principle plane analysis for segmentation, we simply compute the supporting plane defined by the local centroid of the sub-cube o_i and the local average normal vector n_i, and rotate the patch such that the plane's normal is aligned with the Z-axis.

2) Computing and comparing patch descriptors.

After patches alignment, we design a simple statistical descriptor to measure the self-similarity. Inspired by Johnson [11] which uses the spin map to compare the similarity, we compute height and distance histograms to compare the similarity of two patches.



(a) (b)

Fig.2 Projecting point of a patch onto its supporting plane H. (a) the relationship between the original point and its corresponding point on H; (b) H is the supporting plane for the projected points

For a patch P_i, let $d_H(\overrightarrow{p_{ij}})$ be the height value from point P_{ij} projecting to the supporting plane H (see Fig.2(a)) which can be calculated by $d_H(\overrightarrow{p_{ij}}) = (p_{ij} - o_i) \cdot n_i$. Let $P_{ij}'(j = 1,...,m)$ be a set of points collected from projecting the point P_{ij} onto its supporting plane H, shown in Fig.2(b). Let $r_H(\overrightarrow{p_{ij}})$ be the distance between P_{ij}' and local centroid of the sub-cube o_i which can be calculated by $r_H(\overrightarrow{p_{ij}}) = \left\| p_{ij}' - o_i \right\|$.

We compute two histograms of $d_H(\overrightarrow{p_{ij}})$ and $r_H(\overrightarrow{p_{ij}})$. The similarity between two patches can then be evaluated using any distance measure suitable for histograms. We use the cross-bin match distance technique proposed by Hubo [6] to measure the similarity:

The two histograms of $d_H(\overrightarrow{p_{ij}})$ and $r_H(\overrightarrow{p_{ij}})$ are converted to their cumulative version, and concatenated into a single descriptor vector. Finally, the similarity of two patches can be simple performed by comparison on this vector using cross-bin match distance:

$$d(E,G) = \sum_i |E_i - G_i| \tag{2}$$

where $E_i = \sum_{j \le i} E_j$ is the cumulative histogram of E, and similarly for histogram G.

C. Codebook generation for self-similarity patch chains

Codebook generation for self-similarity patch chains is achieved by grouping similar items (patch descriptors) together. To achieve the goal, we use codebook generation technique [6] with following steps:

1) Searching and clustering. We select a representative patch, and look for a set of all similar patches using a nearest neighbor query in the space of descriptor vectors. The cross-bin match distance and the radius of the query determine codebook size which represents how many code words will be created. This process continues until all patches have been clustered into one of the self-similarity patch chains. We always select the patch with the most number of points in the pool of the remaining patches as the representative patch of a certain patch chain. This ensures that every point in the similar patches can find its matching point in the reference patch.
2) Refined matching. In each cluster, we compute the affine alignment of each patch, i.e., translation and rotation, with respect to its representative patch via the Iterated Closest Point (ICP) algorithm [12]. After ICP processing, the similar patch orientation and translation matrix can be obtained and we can optimally match every point in the similar patches with a certain reference point in the reference patch. The refined matching is needed to construct point-to-point correspondence among points in the similar patch and the reference patch.
3) Codebook generation. After similar patches clustering and refined matching, the codebook is then generated for all self-similarity patch chains in which every entry simply consists of the representative patch, the similar patch index, the similar patch orientation and translation.

III. The Proposed Steganographic Scheme

The overview of the embedding procedure and extraction procedure is shown in Fig.3.

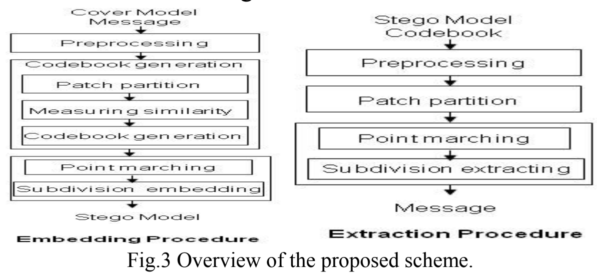


Fig.3 Overview of the proposed scheme.

A. Information embedding

The steps of embedding procedure are as follows.

1) Preprocessing. Principal component analysis (PCA) determines three principle axes centered on the centroid of the model. The original coordinate system is translated to a new one. The new coordinate system has a new origin, which is the center of the model. It also has three basis vectors, which are the three principal axes.
2) Patch partition and codebook generation. This step partitions the model to patches using octree structure, constructs similarity patch chains with self-similarity measures and generates codebook. The details are

described in section Ⅱ. The representative patch of every chain in the codebook is then selected as reference patch; we use this reference-patch model to embed information.

3) space subdivision embedding (SSE). We consider every patch except representative patch in the codebook as message patches. We also consider every point in the message patch as message point. To embed information in every message point, we use a SSE. In SSE, we embed the information by translating the message point from current point to the corresponding numbered subspace. The SSE procedure is as following steps:

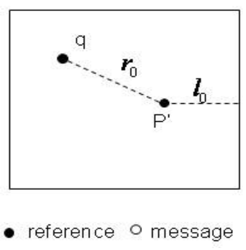
● We use a pseudo sequence to traverse the codebook. The traversal list is maintained as a secret key for both the embedding and extraction procedures.

● For each entry in the codebook, we take the representative patch P as the reference patch and select the similar patch Q as the message patch according to the similar patch index. The similar patch orientation and translation matrix is then used to accurate aligning the message patch with the reference patch P.

● The reference patch P is translated so that its centre is the center of the message patch Q. Let P' be the translated reference patch and $p' \in P'$ be the translated reference point. In the following we always operate on the reference point p' and the reference patch P'.

● In order to embed information, we need to establish a point-to-point correspondence among points in the message patch Q and the reference patch P'. For this purpose we use point matching process. The method finds a point $q \in Q$ such that the 3D Euclidian distance from a point $p' \in P'$ is the smallest among all the points (see Fig.4), and repeats the process for all the point p' in P'. Since we select the reference patch with the most number of points, the points in the message patch can always find its corresponding point in the reference patch.

Let r_0 be the distance between point q and the reference point p'. Let l_0 be the minimum distance from the point p' to the cell boundaries (see Fig.4). The threshold value R is set to $R = \alpha \min(r_0, l_0)$ $(0 < \alpha < 1)$ where α be the modulation amplitude ratio, which ensures that the shifting of the message point will not cross the cell boundaries.



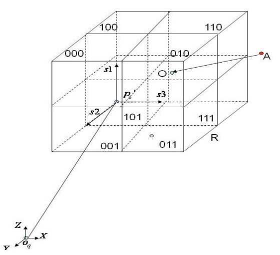


Fig.4 Find corresponding points Fig.5 Message point A shifts to the embedding point O when embedding information

●As the point-to-point correspondence has been established, we embed information using space subdivision embedding (SSE) process. The key idea is to translate the message point to the corresponding numbered subspace.

Let o_q be the local centroid of the patch Q. Assume the reference point $p' \in P'$ is the vertex of the cube and the edge length of it is the threshold value R. Define embedding points as some specific points inside the virtual cube.

We construct the local coordinate system with the point p' as origin for the virtual cube (see Fig.5). The three principal axes of $s1, s2, s3$ coincides with the z, y, x axis, respectively.

Extending the QIM concept to the $s1, s2, s3$ directions of virtual cube, we subdivide the virtual cube into smaller subspaces according to the magnitude of c, which represents the maximum bits that can be embedded into each point of the message patch. Let $m = 2^c$ be the number of subspaces that are subdivided. Figure 5 demonstrates a situation where eight subspaces can be generated when c is set to three. Each subspace is numbered using a three-bit string, namely, 000, 001, 010, 011, 100, 101, 110, 111.

We acquire every three-bit string from the payload message and embed them into the message point of the message patch. For example, in Figure 5, the message point A currently is to embed three bits payload message "010", then we translate the position of the point from its current position to the centre of the new 010-subspace, the vertex O. This translation represents a status change for this message point, and thus a message is embedded.

Finally, after all points in the message patch have been processed by the SSE, we use inverse transformation of patch orientation and translation matrix to recover the embedded message patch to its original position and orientation.

The shifting of the message point keeps the feature of the smallest 3D Euclidian distance between the reference point and the embedding point, which should not change the point-to-point correspondence between them. It is impossible to distort the embedding list in the extraction. Since the codebook is used in both the embedding and extraction, it is a side-match steganography.

B. Information extracting

In this paper, one of our main goals is capacity. We assume no robustness requirements, except basic operations of 3D point cloud models such as affine transformations, which include translation, rotation, uniform scaling, or their combined operations. For simplicity, we only hide three bits per message point in the spatial domain using the SSE. Nevertheless, it is possible for our method to hide a message with a capacity of more than three bits per point except the points of the reference patches. No errors were found in the recovered messages, even when we applied some arbitrary affine transformations.

As suggested by Cheng [2], we evaluate the distortion using Eq. 3.

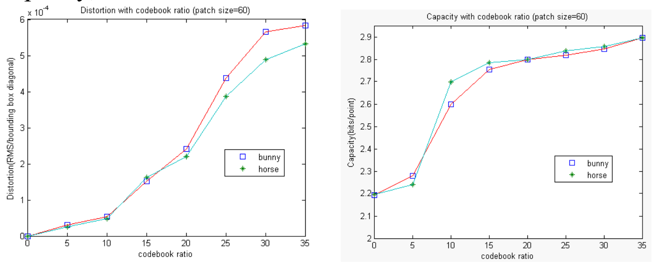
$$\frac{\max(RMS(U,C), RMS(C,U))}{d_B} \qquad (3)$$

where U is the original point cloud and C the steganographic point cloud, d_B is the bounding box diagonal of U and RMS is the symmetric root mean square distance which measures Hausdorff distance between discrete point-pairs.

Since the distortion and capacity are influenced by patch size and codebook, we show the correlation among them. We define codebook ratio as the ratio of the original number of patches vs. the number of patches in the codebook to measure codebook size [6]. Codebook ratio can be influenced by changing the maximal search distance of the nearest neighbor query in the patch space. Since a larger codebook which represents shorter patch chains and more reference patches decreases the capacity, patches should be as big as possible to reduce codebook. However, using big patches results in low capacity because big patches cannot find similar patches as easy as small patches. Let g be the codebook ratio and v be the number of points in the model. The theoretical maximal capacity (M) of our scheme can be simply stated by Eq.4.

$$M = 3 \times \frac{g-1}{g} \times v \qquad (4)$$

We have tested our steganographic scheme on several models with patch size being equal to 60 points: bunny and horse. Fig.6a shows the relation between the code book ratio and distortion. Fig.6b shows the relation between the code book ratio and the capacity. The curves indicate a larger code book ratio has a smaller codebook size in return for a higher capacity, but leads to more distortion.



(a) (b)

Fig.6 Relation between code book ratio and distortion, and code book ratio and capacityfor different models with the same patch size=60

Table 1 presents the model details, the embedded message, model distortion and processing time under the specified conditions. The distortion values are rather small, indicating insignificant loss of quality for the stego models.

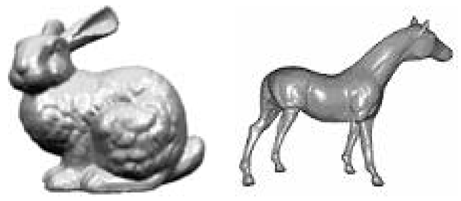
Visual results of the cover and stego models of the bunny and horse models are shown in Fig.7 and Fig.8 respectively. The visual appearance of images shows insignificant distortion for the stego models.

From the security point of view, the guarantee of security of our scheme comes from four keys: patch size which influences the partition, codebook size which is controlled by the maximal search distance of the nearest neighbor query in the patch space, the traversal list of codebook and embedding list over each patch which is controlled by pseudo-random sequence. It is difficult for attacks to obtain four keys even they use exhaustive search. Retrieving the message without the keys is virtually impossible. So our scheme has high security in the sense of cryptography.

Finally, Table 2 offers a detailed comparison of the five related steganographic methods for 3D point cloud, which include [1, 3, 4, 5]. All of them are robust against affine transformations and can extract messages without the assistance of the cover model. As mentioned previously, our approach goes one step further to achieve side-match steganography for 3D point cloud. Moreover, our approach offers a large improvement in capacity with little distortion, compared to the previous 3D point cloud steganogrpahic methods.

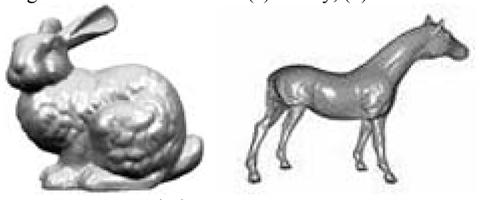
TABLE 1. RESULTS OF VARIOUS MODELS. FOR TESTING, WE EMBEDDED 3BITS PER POINT EXCEPT POINTS OF REFERENCE PATCHES AND CHOSE PATCH SIZE=60, CODEBOOK RATIO=15

Model	Points	Embedded messages(bits)	Distortion
bunny	35947	98973	1.52×10^{-4}
horse	48485	134955	1.63×10^{-4}
dinosaur	56194	155562	2.87×10^{-4}
teeth	116604	320604	7.94×10^{-5}



(a) (b)

Fig.7 The cover models for (a) Bunny, (b) Horse



(a) (b)

Fig.8 The stego models for (a) Bunny, (b) Horse

TABLE 2. THE COMPARISON OF THE FIVE RELATED METHODS. THE CAPACITY IS DENOTED AS AVERAGE BITS PER POINT.

	Cheng [1]	Cotting [3]	Wang [4]	Luo [5]	Our method
capacity	≈3	≈1	≈0.5	<1	≈3
extraction	blind	blind	blind	blind	blind
robustness	affine	affine	affine	affine	affine
domain	spatial	transform	transform	transform	spatial

IV. CONCLU0SIONS

This paper presents a high-capacity spatial blind steganogrpahic approach for 3D point cloud models. A new methodology has been developed to construct self-similarity patch chains and embed messages to every matching point using proposed SSE procedure. Our technique provides steganography with high capacity, security, low distortion, and robustness against affine transformations.

The main remarkable features of the proposed scheme include: (1) PCA-based preprocess and octree-based partition uniquely segment 3D point cloud models to patches, which provides the robustness of affine attacks. (2) Reference patches which are selected from constructed similarity patch chains and codebook are side-match information for steganography. (3) The scheme efficiently gains high capacity that every point except points of reference patches can embed at least four bits using the proposed SSPM. In addition, the secret keys of patch size, codebook size, the traversal list of codebook and embedding list over each patch provide more security, recovering messages without assistance of the secret keys is really impossible.

To the best of our knowledge, this is the first side-match steganographic scheme for 3D point cloud that uses the spatial point marching to embed messages. Our approach is intuitive and can achieve high capacity with little distortion.

REFERENCES

[1] Y. M. Cheng, C. M. Wang, Y. Y. Tsai, et al, "Steganography for three-dimensional models," CGI 2006, LNCS 4035, 2006, pp. 469-476.

[2] Y. M. Cheng, C. M. Wang, "A high-capacity steganographic approach for 3D polygonal meshes," IEEE Trans. Visual Computer, 2006, 8(22), pp. 845-855.

[3] D. Cotting, T. Weyrich, M. Pauly, and M. Gross, "Robust watermarking of point-sampled geometry," In: Proceedings of International Conference on Shape Modeling and Applications, June 7-9, Palazzo Ducale, Genova, Italy, 2005, pp. 233-242.

[4] C. M. Wang, P.C. Wang, "Data hiding approach of point-sampled geometry," IEICE Trans. Comm, 2005, E88-B(1), pp. 190-194

[5] H. Luo, J. S. Pan, Z. M. Lu, and H.C. Huang, "Reversible data hiding for 3D point cloud model," In: Proc. of the 2006 International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Dec 18-20, Pasadena, California, USA., 2006, pp. 863-867.

[6] E. Hubo, T. Mertens, T. Haber, P. Bekaert, "Self-similarity based compression of point set surfaces with application to ray tracing, " Computer & Graphics, 2008, (32), pp. 221-234.

[7] M. Pauly, M. Gross, L.P. Kobbelt, "Efficient simplification of point sampled surfaces," In: Proceeding of IEEE Visualization 2002. Washington, DC, USA: IEEE Computer Society, 2002, pp. 163-170.

[8] Q.S. Ai, Q. Liu, Z.D. Zhou, et al, "A new digital watermarking scheme for 3D triangular mesh models," Signal Processing, 2009, (89), pp. 2159-2170.

[9] B. Tamy, R. Patrick, S. Christophe, "Visualization of point-based surfaces with locally reconstructed subdivision surfaces," In: IEEE Proceedings of the International Conference on Shape Modeling and Applications, June 15-17, Cambridge, MA, USA, 2005, pp.23-32.

[10] S. C. Cheng, C. T. Kuo, D. C. Wu, "A novel 3D mesh compression using mesh segmentation with multiple principal plane analysis," Pattern Recognition, 2010,(43), pp. 267-279.

[11] A. E. Johnson, M. Hebert, "Using spin-images for efficient multiple model recognition in cluttered scenes," IEEE Transactions on Pattern Analysis and Machine Intelligence, 1999, 21(5), pp. 433–449.

[12] S. Rusinkiewicz, M. Levoy, "Efficient variants of the ICP algorithm," In: Proceedings of the third international conference on 3D digital imaging and modeling, May 28-June 1, Quebec City, Canada, 2001, pp.145-152.

[13] A. C. Rencher, Methods of multivariate analysis, 2nd edn., Wiley, New York (2002).

Qi Ke was born in HengYang, Hunan Province, China, in 1972. He is currently a Ph.D. candidate at the school of Computer and Communications, Hunan University, Changsha, China. He received his Master's degree in Computer Application Technology in 1999 from Hunan University, China.
He is an associate professor of School of Computer Science and Education Software, Guangzhou University, Guangzhou, China. His research interests include three-dimensional steganography and watermarking, secret sharing, and trusted computing.

Xie Dong-Qing was born in YiYang, Hunan Province, China, in 1965. He received his Ph.D. degree in Computer Science from Hunan University, China, in 1999.
He is currently a professor at the school of Computer Science and Educational Software, Guangzhou University, Guangzhou, China. His research interests include network security, trusted computing and network, three-dimensional steganography and watermarking, and cryptographic algorithm.