# INFORMATION TECHNOLOGY JOURNAL

Information Technology Journal 10 (4): 877-882, 2011
ISSN 1812-5638 / DOI: 10.3923/itj.2011.877.882
© 2011 Asian Network for Scientific Information

# 3D Mesh Compression by Generalized Parallelogram Predictive Vector Quantization

[1]Jie Xu, [1]Hao Jiang and [2]Zhen Li
[1]Zhejiang University City College, Hangzhou, Zhejiang Province 310015, China
[2]Department of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

**Abstract:** The transmission and storage of large amounts of vertex geometry data are required for rendering geometrically detailed 3D graphic models. To mitigate bandwidth requirements, Vector Quantization (VQ) is an effective lossy compression technique for vertex data in triangular meshes. In this study, we present a new vertex encoding scheme based on VQ. Particularly, we propose a novel prediction method that generalizes the conventional parallelogram prediction method and further reduces the prediction error. During the encoding process, the vertex to be encoded is predicted by all the encoded vertices neighboring to it within a large edge-distance, instead of the encoded vertices directly connecting to it with 1-edge-distance as in the conventional parallelogram prediction. Experimental results show that, compared with the vertex encoding scheme based on the conventional parallelogram prediction, the proposed algorithm consistently achieves a higher encoding quality at the same bit rate.

**Key words:** Computer graphics, 3D mesh, vector quantization, vertex data compression, high order prediction

## INTRODUCTION

Interactive computer graphics plays an important role in various fields such as entertainment, manufacturing and virtual reality When combining the graphics technology with the Internet, the transmission delay for 3D graphics data is becoming a major performance bottleneck, especially for meshes consisting of millions of triangles (Chou and Meng, 2002). As regard to the limited network bandwidth, as well as the storage problem within host systems, reducing the amount of data is, go without saying, an effective solution. Consequently, the interests in compression techniques for the 3D geometry data have surged in recent years.

At present, connectivity compression has reached its limit of less than two bits per triangle for the connectivity portion of a mesh; On the other hand, relatively more works need to be done in the geometry portion of a triangle mesh. Therefore, it is necessary to develop effective compression techniques for the vertex data in order to further reduce overall mesh representation and to meet geometry bandwidth requirements.

In this study, we are engaged in the application of Vector Quantization (VQ) (Gray, 1984) as an effective compression technique for vertex geometry data. So far, VQ is a popular technique for data compression and has already been extensively applied in audio, still image and video coding (Ilyas *et al*., 2010; Al-Husainy, 2007; Naushahi *et al*., 2006). It has many advantages over entropy coding, for example, high rate-distortion performance, real-time decompression for its simple decoding and hardware tractability. Furthermore, we assume that the new triangle can be reconstructed by the neighboring previously reconstructed triangles, so the compatibility is maintained for most existing connectivity compression schemes.

**Prior work:** The early works usually quantize, uniformly and separately, the vertex positions for each coordinate in the Cartesian space. Deering (1995) first proposed a vertex data compression scheme where positions are first normalized within an axis-aligned bounding box. Since then, many variations of Deering's scheme were proposed (Deering, 1995; Touma and Gotsman, 1998; Taubin *et al*., 1998) and more sophisticated vector quantization schemes have been proposed by Chou and Meng (2002) and Lee and Ko (2000) as well. Karni and Gotsman (2000) demonstrated the relevance of applying quantization in the space denoted by spectral coefficients. Sorkine *et al*. (2003) addressed the issue on how to reduce the visual effect due to quantization errors.

As for the prediction scheme, the early work (Deering, 1995) employed simple delta coding or linear prediction

**Corresponding Author:** Jie Xu, Zhejiang University City College, Hangzhou, Zhejiang Province 310015, China

along the vertex order dictated by the coding of the connectivity data (Touma and Gotsman, 1998; Taubin *et al*., 1998). The approach proposed by Lee *et al*. (2002) achieves better visual appearance by applying quantization in the angle space after prediction. Inspired by the TG parallelogram prediction scheme (Touma and Gotsman, 1998). Isenburg and Alliez (2002) complete the techniques described by Khodakovsky *et al*. (2002) by generalizing it to polygon mesh geometry compression. Kronrod and Gotsman (2002), prediction trees are employed to improve the prediction where the geometry drives the traversal order instead of the connectivity. Sorkine *et al*. (2003) suggest a multi-way prediction technique, where each vertex position is predicted from all its neighboring vertices, as opposed to the one-way parallelogram prediction. In the approach proposed by Shikhare *et al*. (2001), the redundancy is removed by detecting similar geometric patterns.

**Proposed method:** The basic idea for vertex encoding in the existing 3D mesh geometry compression schemes is the parallelogram prediction, which predicts the vertex based on the assumption of planer mesh surface. In this work, we propose a VQ scheme which optimally employs the correlation between the vertex to be encoded and its adjacent encoded vertices in a larger edge-distance. The proposed prediction method is a generalized parallelogram prediction algorithm, which achieves better quality at the same bit rate as verified in the experiment section.

Conceptually, VQ is a generalization of non-uniform scalar quantization to operate on vectors rather than scalars and offers superior performance over scalar quantization in terms of rate-distortion (Gray and Neuhoff, 1998). VQ can be defined as a mapping procedure from a q-dimension Euclidian space to a finite subset, i.e., Q: $R^q \rightarrow C$, where the subset $C = \{c_i \mid i = 1, 2,...,N\}$ is called codebook, in which $c_i$ is a codevector and N is the codebook size. The mapping procedure is as follows: Select an appropriate codevector $c_p = (c_{p0}, c_{p1}, c_{p\,(q-1)})$ as the decoded vector for the training vector $x = (x_0, x_1,...,x_{(p-1)})$, to guarantee that the codevector ---- is the closest vector to x amongst all the codevectors in C. The distance metric is usually the square Euclidian measure as follows:

$$d(x,c_i)=\|x-c_i\|^2=\sum_{j=1}^{q}(x_j-c_{ij})^2 \qquad (1)$$

**Vector quantizer design:** The first issue in designing a VQ scheme for compressing any kind of source is how to map the source data into a vector sequence as the input of the vector quantizer.

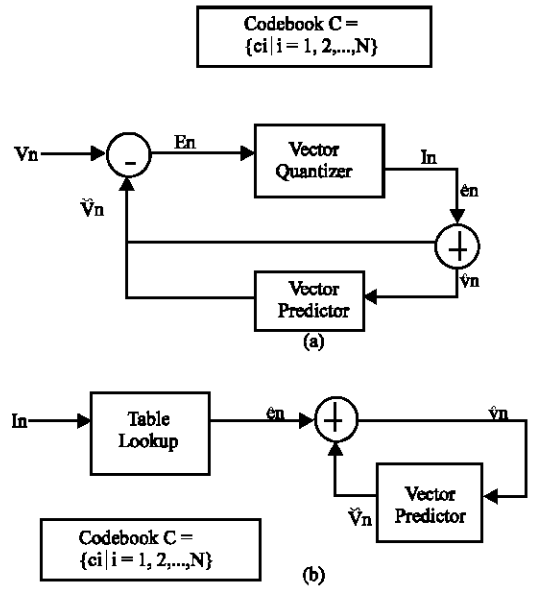

Fig. 1: Predictive vector quantizer (a) encoder and (b) decoder (a) Encoder (b) Decoder

We exploit Predictive Vector Quantization (PVQ) (Cuperman and Gersho, 1985) that is proper for 3D mesh compression (Chou and Meng, 2002). A block diagram of the PVQ encoder and decoder is shown in Fig. 1. In order to exploit the correlation between vertices, it is necessary to use a vector quantizer with memory. Let $\{v_i\}^N_{i=0}$ represent the sequence of vertices encountered as a mesh is being traversed dictated by its topology information and let $v_n$ be the vertex to be encoded. The encoder forms a prediction $v_n$ of $v_n$ based on observations of previously encoded vertices. The residual vector $e_n$, i.e. the error vector, is then computed as $e_n = v_n - v_n$. This residual vector is then quantized by the vector quantizer, which generates a codevector $e'_n$ that approximates $e_n$. The index identifying this codevector is then stored or transmitted to the decoder. Each vertex is thus encoded with $\log_2 N$ bits in ordinary VQ, where N is the codebook size.

To permit reconstruction of the vertices by the decoder, the prediction must only be based on previous reconstructed vertices. Therefore, the encoder also needs to reconstruct the vertex to be encoded for computing the prediction vectors for subsequent vertices. The decoder receives the sequence of indices. Given an index, the decoder first performs a table lookup operation to obtain the residual vector $e'_n$. The decoder then adds $e_n$ and $v_n$ as shown in Eq. 2 to reconstruct the quantized vertex $v_n$. As in the case of the encoder, $v_n$ is fed back to the predictor for computing subsequent vertex predictions. The residual vectors are then used as training vectors to generate the codebook based on the minimax partial distortion competitive learning technique (Zhu and Po, 1998). Compared with some well-known codebook design algorithms, the MMPDCL algorithm consistently produces the best codebooks with the smallest average distortions.

**Prediction design:** The proposed prediction scheme is inspired by the famous parallelogram prediction rule (Isenburg and Alliez, 2002), which is intuitively illustrated in Fig. 2. In parallelogram prediction, a vertex is predicted by its neighboring triangle, exploiting the tendency for neighboring triangles to be roughly coplanar and similar in size. This is particularly true for high-resolution, scanned meshes, which have little variation in the triangle size. Suppose that $\hat{v}^{(n-1)}$, $\hat{v}^{(n-2)}$ and $\hat{v}^{(n-3)}$ in Fig. 2 are the three vertices of a neighboring triangle buffered by the encoder. Then, an effective and computationally inexpensive way to compute the prediction $v_n$of the next vertex $v_n$ is:

$$\tilde{\mathbf{v}}_n = \hat{\mathbf{v}}^{(n-1)} + \hat{\mathbf{v}}^{(n-2)} - \hat{\mathbf{v}}^{(n-3)} \tag{2}$$

However, in case of more curving 3D meshes, Eq. 2 is not proper any more. To tackle this problem, we propose a generalized prediction scheme here, in which $e_n$ is minimized by observing the encoded vertices within k-ring edge-distance to achieve a better prediction.

If we consider $v_i \in M$, a vertex whose 3D coordinates are given by the vector $v_i$, from the 3D mesh M, we define its r-ring neighborhood as the vertices connected to it with a r-edge distance:

$$N_r(\mathbf{v}_i)=\left\{\forall \mathbf{v}_j, j=1,\cdots,L \left|\mathbf{v}_j\mathbf{v}_i\right| = r, r=0,1,\cdots\right\} \tag{3}$$

where, $|v_j\, v_i|$ denotes set cardinality and counts the set of points on the line segment joining the two vertices, but excluding the end-points and L denotes the total number of vertices of the neighborhoods $N_r$ ($v_i$). The point $v_i$ is considered as being its own 0-ring neighborhood.

Then we can represent quantized $v_n$ and its neighborhood in the i-th ring as follows:

$$\hat{v}^{(m)}_{n-i}=\tilde{v}^{(m)}_{n-i}+e^{(m)}_{n-i}, i=0,1,...,k \tag{4}$$

where $v_{n-1}$ is a i-ring neighborhood to $v_n$, m = 0, 1,...,$L_i$ and $L_i$ is the cardinality of vertices in $v_n$'s i-ring neighborhood, while $v^{(m)}_{n-i}$ and $e^{(m)}_n$-i are $v_n$'s m-th prediction vector from Eq. 2 and error vector, respectively.

We assume that all the triangles in the mesh are equilateral, which is an ordinary assumption in 3D mesh analyses. We define $\in^{(m)}_{n-i}$ as the average error vector $e^{(m)}_{n-i}$ from the i-th ring of $v_n$. It is certainly that when $v_n$ is the vertex to be encoded, $\in$n = $e_n$ (0) =$e_n$.

If the mesh surface is rather planar, we can estimate that $\forall \in_{n-i} = 0$, I = 0,...,k; when the shape of the surface gradually becomes more curving, it can be approximated that $\in_{n-i}$ is a non-zeros constant, i.e., $d^2 \in_n/dn^2 = 0$ and then

$$\boldsymbol{\varepsilon}_n = \boldsymbol{\varepsilon}_{n-1} \tag{5}$$

When the shape of the mesh surface becomes even more curving, namely, $\boldsymbol{\varepsilon}_{n-i}$ varies more dramatically, $d^2\boldsymbol{\varepsilon}_n/dn^2k = 0$ may occur and then $d\boldsymbol{\varepsilon}_n = d\boldsymbol{\varepsilon}_{n-1}$, i.e.

$$\boldsymbol{\varepsilon}_n = 2\boldsymbol{\varepsilon}_{n-1} - \boldsymbol{\varepsilon}_{n-2} \tag{6}$$

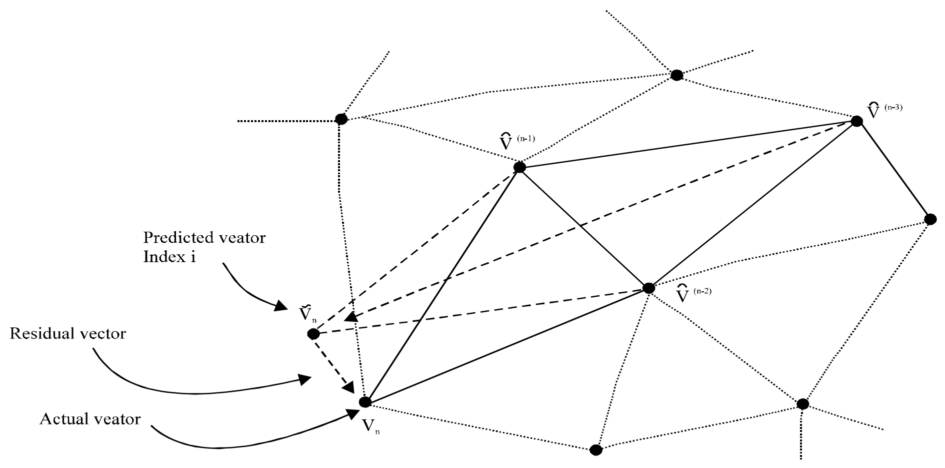

Fig. 2: Prediction $\tilde{v}_n$ of $v_n$ to be encoded using previously reconstructed vertices $\hat{v}^{(n-1)}$, $\hat{v}^{(n-2)}$, $\hat{v}^{(n-3)}$

When the shape of the mesh satisfy the following condition:

$$d^k \varepsilon_n / d\, n^k = 0 \tag{7}$$

Then we can generalize $\varepsilon_n$ using induction theory:

$$\mathbf{e}_n = \sum_{i=1}^{k} C_k^i \mathbf{e}_{n-i} \cdot (-1)^{i+1} \tag{8}$$

where $C^i_k$ is the combination of i out of k. It is obvious that Eq. 5 and 6 are specific instances of Eq. 8.

Then when the surface satisfies the following condition:

$$d^k \epsilon_n / d\, n^k > 0 \text{ and } d^{k+1} \epsilon_n / d\, n^{k+1} = 0 \tag{9}$$

it can be iteratively deduced as follows:

$$d^k e_n = d^k e_{n-1}$$
$$d^{k-1} e_n = 2 d^{k-1} e_{n-1} - d^{k-1} e_{n-2}$$
$$d^{k-2} e_n = 3 d^{k-2} e_{n-1} - 3 d^{k-2} e_{n-2} + d^{k-2} e_{n-3}$$

$$de_n = \sum_{i=1}^{k} C_k^i de_{n-i} \times (-1)^{i+1} \tag{10}$$

according to the assumption in Eq. 8.

From Eq. 10, we can easily deduce that:

$$\begin{aligned} e_n &= e_{n-1} + \left( \sum_{i=1}^{k} C_k^i e_{n-i} \times (-1)^{i+1} - \sum_{i=1}^{k} C_k^i e_{n-1-i} \times (-1)^{i+1} \right) \\ &= (k+1) e_{n-1} - (C_k^1 + C_k^2) e_{n-2} + (C_k^2 + C_k^3) e_{n-3} + L + e_{n-1-k} \times (-1)^{k+1} \end{aligned} \tag{11}$$

Combined with the basic theorem as follows:

$$C_k^m + C_k^{m+1} = C_{k+1}^m \tag{12}$$

we can deduce that:

$$e_n = \sum_{i=1}^{k+1} C_{k+1}^i e_{n-i} \times (-1)^{i+1} \tag{13}$$

It can be safely concluded now that assumption (8) is true according to the induction theory based on Eq. 5, 6, 8 and 13.

Now we can represent $v_n$ as follows when Eq. 7 is satisfied:

$$\bar{v}_n = \tilde{v}_n + e_n$$

$$= \tilde{\mathbf{v}}_n + \sum_{i=1}^{k} C_k^i \mathbf{e}_{n-i} \cdot (-1)^{i+1} \tag{14}$$

where $v_n$ is the newly predicted vector of $v_n$, while $\varepsilon_{n-1}$ is the average of the error vector $e^{(m)}_{n-1}$ for each predicted vertex.

Now Eq. 4 is updated to the following equation:

$$v_n = \bar{v}_n + \eta_n \tag{15}$$

where, $\eta_n$ is the error vector between $v_n$ and $v_n$ and is expected to be of less magnitude than $e_n$ that is generated by the conventional parallelogram prediction. Parallelogram prediction is obviously equivalent to the situation of $k = 0$ in the proposed algorithm. $\eta_n$ is then quantized as $\hat{\eta}_n$ and represented by its corresponding codevector index. The smaller error vectors are, the higher quantization quality is expected to be achieved.

**Vertex encoding:** At first, the 3 vertices of the initial triangle of the region growing process are uniformly scalar quantized at 8 bit in each coordinate and then Huffman encoded. Then we set the error vectors of these 3 vertices to be 0, i.e. if we encode the first vertex $v_j$ that is adjacent to the initial triangle, $\varepsilon_{j-1} = \varepsilon_{j-2} = 0$. $V_n$, a subsequent vertex in the traversal order, is at first predicted by Eq. 2, obtaining its initial predicted vector $v_n$ and each $\varepsilon_{n-1}$ is computed by averaging all the error vectors in the i-ring of $v_n$. Then the new predicted vector $v_n$ is computed according to Eq. 14. When we have encoded $v_n$, the prediction error $\eta_n$ is quantized by the codebook and identified by its corresponding codevector index. The encoded vector $v'_n$ is computed by adding $v_n$ to $\hat{\eta}_n$. $v_n$ is replaced with $\hat{\eta}_n$ in order not to accumulate errors for the encoding of subsequent vertices. The error vector $v'_n$ is calculated by Eq. 4 and then buffered for computing the subsequent average error vectors.

**Vertex decoding:** At first, the 3 vertices of the initial triangle are Huffman decoded. When we encode a subsequent vertex $v_n$, we get its initial predicted vertex $v'_n$ from Eq. 2 and compute each $\varepsilon_{n-1}$ of the i-th ring of $v_n$. Then $v'_n$ is computed according to Eq. 14. Now $v_n$ is easily acquired by adding $v_n$ to $\hat{\eta}_n$ that is identified by its corresponding codevector index. The error vector $e_n$ of $v_n$ is calculated by Eq. 4 and buffered. Note that when we encode the first vertex $v_j$ that is adjacent to the initial triangle, $\varepsilon_{j-1} = \varepsilon_{j-2} = 0$. Thus according to Eq. 14, it is apparent that $v_n = v'_n$ and then $\hat{\eta}_j = \varepsilon_j = e_j$, namely, the first VQ encoded vertex is actually predicted by the traditional parallelogram prediction rule.

**Residual vector quantization:** In our scheme, 42507 residual vectors were randomly selected from a famous 3D mesh library (Princeton University 2001) for training the approximate universal codebook off-line, and the size of it ranges from 64 to 8192. In this way, we expect the codebook to be suitable for nearly all triangle meshes for VQ compression and can be pre-stored in each terminal in the network. Thus the compressed bit stream can be transmitted alone without any codebook in convenience.

## EXPERIMENTAL RESULTS

In this study, we limit k to be 2 in order to reduce the computational complexity while maintaining prediction precision. Distortion results based on the conventional parallelogram prediction VQ is also implemented for comparison. Performance comparisons of the proposed method with the conventional parallelogram on models with various characteristics are shown in Table 1. The codebook size is 1024 for all the experimental meshes, i.e. the compression bit rate is 10.0 bpv. P1 and P2 in Table 1 denote PSNR by the conventional parallelogram method and the proposed method, respectively.

The proposed method performs as high as about 1.7 dB better than the conventional VQ for Stanford Bunny, because of the relatively smooth and curving surface which is proper for the proposed method. For Caltech Feline, some parts of the surface are jugged while other parts are smooth, so the gain of the proposed method is only 0.9 dB. The worst case appear in Fandisk, because it is rather smooth in most area (in this case the proposed method is the same as the parallelogram prediction) and in some parts, the surface is extremely jugged (in this case neither proposed method nor the parallelogram prediction works). We also experiment on a simplified version of Bunny, in order to verify the scalability of the proposed method. Although the surface of the simplified model becomes more jugged than that of the original model, the proposed method also achieves a gain of 0.7 dB.

PSNR in this study is defined as PSNR $20\log_{10}$ Peak/d, where peak is the mesh bounding box diagonal and d is the root mean square error defined by symmetric face to face Hausdorff distance (Aspert *et al*., 2002):

$$d = \max\{d(X, Y), d(X, Y)\} \tag{16}$$

where, d (X, Y) is the asymmetric distance from the mesh X to mesh Y and is defined as:

$$d(X,Y)=\sqrt{1/A(X)\times\int_{x\in X} d(x,Y)^2\,dx} \tag{17}$$

where, d (X, Y) is the Euclidean distance from a point x on X to its closest point on Y and A (X) is the area of X.

Table 1: Comparisons of two methods on different meshes

| Mesh | P1 | P2 | Vertices | Faces |
|---|---|---|---|---|
| Bunny | 61.25 | 62.93 | 35947 | 69451 |
| Feline | 57.12 | 58.02 | 49919 | 99732 |
| Fandisk | 58.03 | 58.11 | 6475 | 12946 |
| Bunny small | 51.97 | 52.65 | 1889 | 3851 |

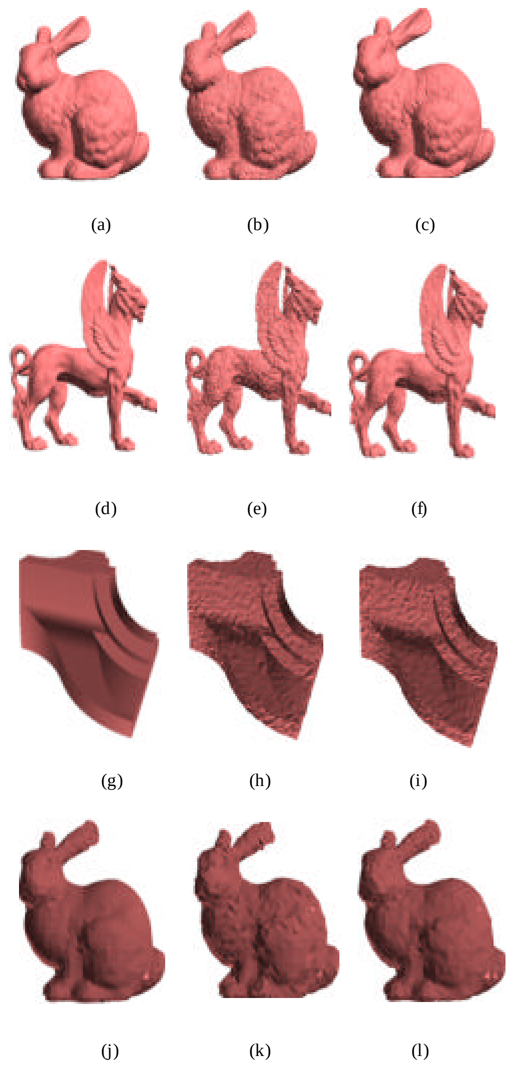

Fig. 3: Rendered meshes of the uncompressed models, compressed models with parallelogram prediction and proposed methods. (a, b, c) Bunny models (d, e, f) Feline models (g, h, i) Fandisk models (j, k, l) Simplified Bunny models. Models in the three columns are the uncompressed, compressed with parallelogram prediction and compressed with the proposed multi-ring prediction, respectively

To illustrate, Fig. 3 depicts the Bunny, Feline, Fandisk and the simplified Bunny meshes uncompressed,

compressed with the parallelogram VQ method and compressed with the proposed method for comparison.

## CONCLUSIONS

This study has demonstrated that the proposed generalized parallelogram prediction scheme based vector quantization is a promising VQ technique for compressing the geometry of triangle meshes. Experimental results indicate that the proposed prediction scheme shows superior performance over the conventional parallelogram prediction scheme which has been used extensively in 3D triangular mesh compression. The prominent benefits of the proposed technique include superior rate-distortion performance, efficient decompression that is amenable to hardware implementation, compatibility with existing connectivity compression schemes and maintaining low computation complexity. The future work will concentrate on the estimation of k based on the characteristics of local surface patches in order to achieve a better rate-distortion performance while maintaining low computation complexity.

## ACKNOWLEDGMENT

This study was supported by Science Foundation of Zhejiang Province in China “electronic walking Assistive Technology Research based on Multi-dimensional continuous data Sonification” (grant No. Y1080338).

## REFERENCES

Al-Husainy, M.A.F., 2007. A tool for compressing images based on genetic algorithm. Inform. Technol. J., 6: 457-462.

Aspert, N., D. Santa-Cruz and T. Ebrahimi, 2002. MESH: Measuring error between surfaces using the hausdorff distance. Proc. IEEE Int. Conf. Multimedia Expo, 1: 705-708.

Chou, P.H. and T.H. Meng, 2002. Vertex data compression through vector quantization. IEEE Trans. Visualization Comput. Graphics, 8: 373-382.

Cuperman, V. and A. Gersho, 1985. Vector predictive coding of speech at 16 kbits/s. IEEE Trans. Commun., 33: 685-696.

Deering, M., 1995. Geometry compression. Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, (ACCGIT’95), ACM., New York, USA., pp: 13-20.

Gray, R. and D. Neuhoff, 1998. Quantization. IEEE Trans. Inform. Theory, 44: 2325-2384.

Gray, R.M., 1984. Vector quantization. IEEE ASSP Magazine, 1: 4-29.

Ilyas, M.Z., S.A. Samad, A. Hussain and K.A. Ishak, 2010. Improving speaker verification in noisy environments using adaptive filtering and hybrid classification technique. Inform. Technol. J., 9: 107-115.

Isenburg, M. and P. Alliez, 2002. Compressing polygon mesh geometry with parallelogram prediction. Proceedings of the IEEE Visualization Conference, Oct. 27-Nov. 01, IEEE Computer Society, Washington, DC., USA., pp: 141-146.

Karni, Z. and C. Gotsman, 2000. Spectral compression of mesh geometry. Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, (CGIT’00), ACM Press/Addison-Wesley Publishing Co., New York, USA., pp: 279-286.

Khodakovsky, A. P. Alliez, M. Desbrun and P. Schroder, 2002. Near-optimal connectivity encoding of 2-manifold polygon meshes. Graphical Models, 64: 147-168.

Kronrod, B. and C. Gotsman, 2002. Optimized compression of triangle mesh geometry using prediction trees. Proceedings of the 1st International Symposium on 3D Data Processing, Visualization and Transmission, Nov. 7, Technion-Israel Institute of Technology, Haifa, pp: 602-608.

Naushahi, K.R., M.A.U. Khan and M.S.H. Khiyal, 2006. Lossless image coding using conditional entropy constrained vector quantization. Inform. Technol. J., 5: 1136-1139.

Lee, E. and H. Ko, 2000. Vertex data compression for triangular meshes. Proceedings of the Pacific Graphics, Oct. 3-5, Hong Kong, China, pp: 225-234.

Lee, H., P. Alliez and M. Desbrun, 2002. Angle-analyzer: a triangle-quad mesh codec. Eurographics, 21: 1-10.

Princeton University, 2001. 3D model search engine. http://shape.cs.princeton.edu/search.html.

Shikhare, S. Bhakar and S.P. Mudur, 2001. Compression of large 3D engineering models using automatic discovery of repeating geometric features. Proceedings of the 6th International Fall Workshop on Vision, Modeling and Visualization, Nov. 21-23, Aka GmbH, pp: 233-240.

Sorkine, O., D. Cohen-Or and S. Toldeo, 2003. High-pass quantization for mesh encoding. ACM Int. Conf. Proc. Ser., 43: 42-51.

Taubin, G., W. Horn, J. Rossignac and F. Lazarus, 1998. Geometry coding and VRML. Proc. IEEE, 86: 1228-1243.

Touma, C. and C. Gotsman, 1998. Triangle mesh compression. Proceedings of the Graphics Interface, June 1998, Vancouver, Canada, pp: 26-34.

Zhu, C. and L.M. Po, 1998. Minimax partial distortion competitive learning for optimal codebook. Design. IEEE Trans. Image Process., 7: 1400-1409.