Blind QIM-LDPC Watermarking of 3D-Meshes

(Invited Paper)

Bata Vasic
Department of Electronic
University of Nis, Faculty of Electronic Engineering
Nis, Serbia
bata.vasic@elfak.ni.ac.rs

Bane Vasic, *Fellow, IEEE*
Electrical and Computer Engineering Department
University of Arizona
Tucson, AZ, USA
vasic@ece.arizona.edu

***Abstract*—We introduce a method for blind watermarking of three dimensional (3D) meshes based low-density parity check (LDPC) coded quantization index modulation (QIM). The 3D host vertex selection algorithm identifies vertices in rough concave regions of the surface that are invariant to mesh optimization/simplification on which the QIM operates. The LDPC code recovers the runlength coded data hidden in the vertices deleted by mesh optimization and/or simplification. For a given bit-error rate performance of a recovered watermark, the proposed scheme permits storing much longer watermarks compared to the state of the art approaches.**

Keywords-3-D mesh; data hiding; deletion channels; error correction; iterative decoding; quantized index modulation; low-density parity check codes; watermarking

I. INTRODUCTION

Watermarking combined with encryption is an effective tool for copyright protection in digital media. Digital watermarking involves hiding a digital signal (watermark) into the cover (or host) signal. While the goal of cryptography is to make a message transmitted through a communication channel incomprehensible by the adversary, the role of watermarking is to hide the existence of communication. Past decade has seen a development of myriad of methods for data hiding and increasing amounts of hidden data while preserving the host fidelity and maintain security and robustness to transformation of host data. A watermarking algorithm is blind if no original host is required for extraction of the watermarked host signal.

A problem of blind watermarking of three-dimensional (3D) mesh objects has gained importance in recent years due to mass use of 3D meshes in computer games, architectural and medical simulations, as well as in computer-aided design (CAD) and digital cinematography. One of principal difficulties in 3D watermarking compared to lower dimensional counterparts (audio and image watermarking) is that even common signal processing operations for manipulation and editing 3D objects, such as mesh simplification and optimization, involve complex geometric and topological operations, which is a serious threat to watermark data. Mesh optimization and simplification is an unavoidable step used in order to accelerate rendering.

As mesh optimization and simplification involve removal of vertices and merging of polygonal faces in the original 3D mesh, deletion of a vertex containing even a single bit of hidden data, makes not only this bit unrecoverable, but also causes loss of alignment between the original watermark and recovered watermark data. For blind watermarking, where original cover signal is not required for watermark detection, such loss of synchronization is catastrophic for watermark robustness. Compared to additive white Gaussian noise attacks which is typically considered in the literature and has solid information-theoretic foundation [1], the difficulty of dealing with a deletion "attack" comes from both incompleteness of theoretical results on coding for the deletion channel as well high complexity of decoders.

In this paper we are concerned with ensuring robustness of blind watermarking under simplification and affine transformations of the host 3D mesh. We analyze a non-malicious transformation scenario, in which some vertices from a mesh are deleted because such transformations, and propose a secure watermarking scheme with guaranteed robustness to mesh optimization and simplification based on a combination of the quantized index modulation (QIM) [2] for data hiding with run-length modulated low-density parity check (LDPC) codes for recovering deleted watermark bits. The watermark recovery has two steps, namely detection of QIM bits and error correction decoding. The redundancy (overhead) introduced in the coded watermark sequence is used in the decoding process to recover the deleted bits. Up to our best knowledge, this is the first error control coded scheme for watermarking of 3D meshes. The only watermarking scheme in literature that employs error correction coding is the method recently proposed by Coumou and Sharma [3]. However it is for audio signal watermarking and is based on the insertion/deletion/substitution correction scheme of Davey and MacKay [4] which combines marker codes for providing synchronization and LDPC codes for error correction. To be effective in terms of correctable number of errors, the Davey and MacKay scheme must use long, non-binary LDPC codes, which make the decoding very complex. In addition, the algorithm involves using highly complex iterative maximum a posteriori (MAP) symbol detection operating on a huge graphical model of the insertion/deletion/substitution channel.

The rest of the paper is organized as follows. Section II describes the block diagram of our approach. It explains the proposed algorithm and its two main components: host vertex selection and coding. Numerical results of the selection of 3D mesh vertices designated for data hiding, as well as results of capacity estimation and error probability are given in Section III. Section VI gives discussion and conclusions.

This work is funded in part by NSF, Grant CCF-0963726. The long version of this paper with a study of different meshes is accepted for publication at *IEEE Transactions on Multimedia* [20].

II. THE PROPOSED ALGORITHM

The block diagram of the proposed watermarking system is shown in Figure 1. In this section we describe in detail its components. We start with QIM, then explain host vertex selection, and finally discuss the coding parts.

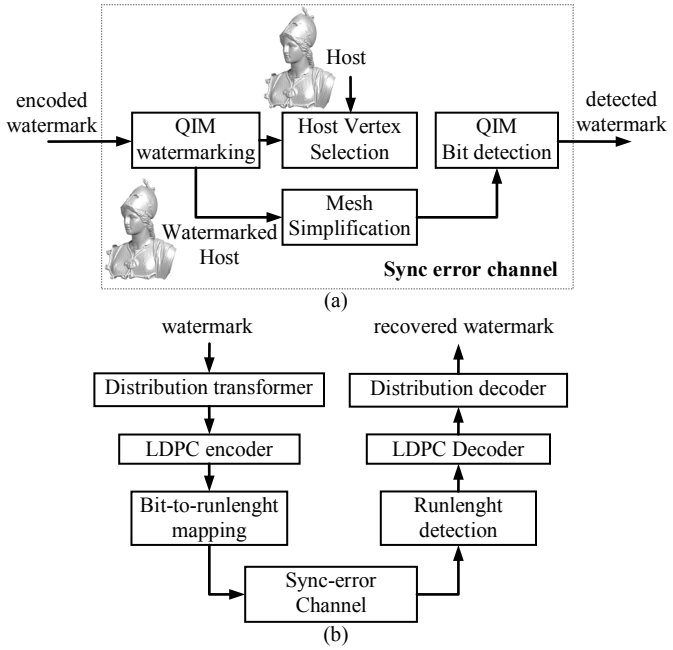


Figure 1. A block diagram of the proposed watermarking system.(a) Operations within a sync-error channel due to mesh simplfication, (b) Block diagram of an equivalent communication system with watermark coding.

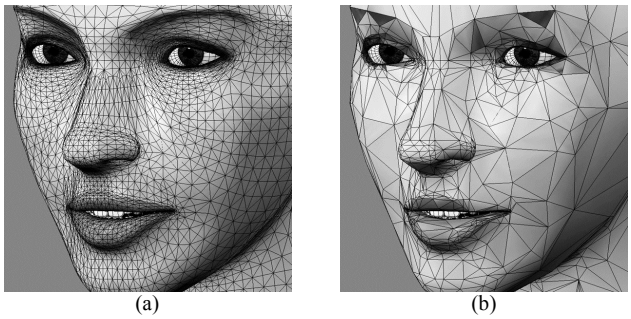
The encoded watermark sequence is embedded into selected vertices of a 3D object by using QIM. Let $\mathbf{u}\in\{0,1\}^n$ and $\mathbf{x}\in\mathbf{R}^n$ be the watermark sequence and the cover sequence, respectively. The *embedder* combines the *n*-dimensional vectors **u** and **x** and produces the watermarked sequence $\mathbf{y}\in\mathbf{R}^n$. The difference **w=y-x** is referred to as the *watermarking displacement* signal. The embedder must keep the distortion $d(\mathbf{x},\mathbf{y})$ within a prescribed limit *D*. The QIM operates on independently on the elements *u* and *x* of the vectors **u** and **x**. Embedding the bit $u\in\{0,1\}$, is done by a choice between two different quantizers $\boldsymbol{Q}_0$ and $\boldsymbol{Q}_1$ with the quantization step Δ so that $y = \boldsymbol{Q}_u(x)$. The quantization function is $\boldsymbol{Q}_u(x)=\Delta\left[\frac{1}{\Delta}\left(x-(-1)^u\frac{\Delta}{4}\right)\right]+(-1)^u\frac{\Delta}{4}$ where, $[x]$ is the integer closest to *x*. Equivalently, the quantization level of the "nominal" quantizer Δ $[x/\Delta]$ is "dithered" by $\Delta/4$ depending on the value of *u*. The quantizer step size affects watermarking robustness and fidelity: larger step leads to more robust detection, but reduces the cover signal fidelity.

The above definition of QIM not put any restrictions on the type of a cover signal, their application to 3D-meshes requires addressing the nontrivial issues connected to robustness to affine transformations. Recently Darazi, *et al.* [5] adapted the QIM for 3D-mesh watermarking.

To ensure robustness to affine transformation, quantization is performed in the spherical coordinate system and the distance from the center of gravity (mass) is the only coordinate that is quantized. As explained in [6], the coordinate origin to the mass center and the principal component vector is aligned with the *z* axis. Our method uses the same approach.

A. Host vertex selection

To ensure adequate fidelity as well as robustness to mesh simplification, watermark data is hidden only in vertices with high curvature and roughness because such vertices are unlikely to be removed by the simplification algorithms. This is illustrated in Figure 2.



(a) (b)

Figure 2. An illustration of the host vertex deletion due to mesh simplification. (a) original object, (b) simplified object. Note that vertices in regions of very high curvature are not deleted in the process of mesh simplification.

The proposed content-aware embedding places the hidden data into the mesh vertices that are invariant to mesh simplification, thus reducing the probability of bit deletion. *Host vertex selection* is based on our recent work [7] which utilizes a number of mesh topology with human visual system (HVS) metrics to calculate vertex stabilities and trace vertices most susceptible for extraction during simplification. The vertex *importance* is proportional to the curvature of its surrounding surface. The removal of these *important* vertices destroys information about the shape, so simplification processes avoid those vertices. The input to the selection algorithm is $M(V,F)$, a mesh of a given 3D surface.

Our vertex selection algorithm [7] has three steps: (i) defining assessment criteria and their ranking, (ii) accurate curvature evaluation with its characteristic features computation, and (iii) tracing importance of extracted vertices in relation to mesh topology. In first step the algorithm extracts the matrix of topological error vertices and the matrix of boundary vertices. Using the values and signs of Gaussian $\kappa_G(\mathbf{v}_i)$ and mean curvature $\kappa_H(\mathbf{v}_i)$ at vertex $\mathbf{v}_i$, the algorithm marks *risky* vertices. Then, using other features, the algorithm calculates the vertex stability vector. Two basic curvature estimation criteria have defined from differential geometry [8] and fitting quadric [9], [10].

$$\mathbf{K}\left(\mathbf{v}_i\right)=\frac{1}{2\mathcal{A}}\sum_{j\in N_1(i)}\left(cot\alpha_{ij}+cot\beta_{ij}\right)\left(\mathbf{v}_j-\mathbf{v}_i\right)$$

$$\kappa_G\left(\mathbf{v}_i\right)=\frac{1}{\mathcal{A}}\left(2\pi-\sum_{j=1}^{N_f}\theta_{ij}\right)$$

where N_f is a number of adjacent triangular faces at the point $\mathbf{v}_i$, and θ_{ij} is the angle of *j*-th adjacent triangle at the point $\mathbf{v}_i$. $\mathcal{A}$ is the area of the first ring of triangles around the point $\mathbf{v}_i$.

For a given point $\mathbf{v}_i$ of discrete surface M the mean curvature normal given in (2) (Laplace-Beltrami operator) gives both the mean curvature $2\kappa_H(\mathbf{v}_i)$ and unit normal $\mathbf{n}(\mathbf{v}_i)$ at the vertex $\mathbf{v}_i$. $\mathbf{K}(\mathbf{v}_i) = 2\kappa_H(\mathbf{v}_i)\mathbf{n}(\mathbf{v}_i)$. The two *principal curvatures:* κ_1 and κ_2 of the surface M, with their associated orthogonal directions $\mathbf{e}_1$ and $\mathbf{e}_2$ are extreme values of all normal curvatures. Voronoi region

$$\mathcal{A}_{Voronoi} = \frac{1}{8}\sum_{j\in N_1(i)}\left(cot\alpha_{ij} + cot\beta_{ij}\right)\left\|v_i - v_j\right\|^2$$

can be defined as a mixed region, which is suggested by Mayer *et al.* [8] for minimization a discretization error:

$$\mathcal{A}_{Mixed} = \begin{cases} \dfrac{1}{4}\displaystyle\sum_{j\in N_1(i)}\left(\cot\theta_{ij} + \cot\alpha_{ij}\right)\left\|v_i - v_j\right\|^2 \sin^2\theta_{ij} \\ \dfrac{1}{8}\displaystyle\sum_{j\in N_1(i)}\left(\cot\theta_{ij} + \cot\alpha_{ij}\right)\left\|v_i - v_j\right\|^2 \sin^2\theta_{ij} \end{cases}$$

The second group of vertex features that we have used in consideration is a set of features that characterize salience of regions and resistance to vertex decimation [11].

B. Ensuring synchronization

The main difficulty in addressing the problem of code construction is the fact that channels with synchronization errors have infinite memory, i.e., a synchronization error affects all subsequent symbols. Thus we use *runlength coding* to translate a channel with synchronization errors into a memoryless channel. Prior to runlength coding the watermark is first encoded by a binary LDPC code. The runlenghts representing zeros and ones are selected according to channel synchronization statistics, and we choose the length two to represent a zero, and length three to represent a 1. We explain the function of runlength coding after we introduce the channel corresponding to extracting the watermark.

The distribution transformer is readily achieved by employing arithmetic decoder as suggested in [12]. The data is then encoded by using an error-correcting code. Subsequently, the information symbols are encoded in runs of channel bits as described previously. At the output, all the operations are performed in reverse in order to obtain an estimate of the input data. In this paper we consider the scenario in which the distortion of the signal occurs only due to common, non-malicious mesh simplifications. As we see, in this case, the data hiding channel introduces deletions only. If the channel introduces deletions, then the received sequence has the form $(y_1, y_2, ..., y_{i_1-1}, y_{i_1+1}, ..., y_{i_2-1}, y_{i_2+1}, ..., y_{i_l-1}, y_{i_l+1}, ..., y_{n-l})$ where i_1, i_2, ..., i_l are the positions of l deletions. Mitigating the effects of deletions is achieved by protecting the watermarked bits by an LDPC code and modulating coded bits by a run-length code. As showed by Krishnan and Vasic [13] who use this idea in the context of data storage, the run-length code has a crucial role in transforming the notoriously difficult insertion/deletion/substitution channel which has infinite memory into a memoryless channel. Runlength coding introduces an inherent rate loss, but from the decoder side the transformed channel looks like the memoryless channel, and allows us to use the reach knowledge in LDPC codes to design powerful codes and low-complexity iterative decoders. LDPC codes and iterative decoders can be designed to guarantee upper bounds of watermark decoding probability.

Consider a channel with binary input and output alphabets. The most general formalism that describes synchronization and bit-flip error is a finite-state machine [4]. At time t_i a bit may be inserted with a probability p_i, deleted with a probability p_d, or successfully transmitted with a probability $1-p_i-p_d$. We refer to insertions and deletions as to *synchronization errors*. The possibility of a change in the value of the transmitted bit can also be encompassed by this model. In the context of watermarking, the channel introduces only deletions ($p_i = 0$). Furthermore we assume that no two consecutive vertices are deleted in the process of simplification. This assumption is justified by the fact that the selection algorithm identifies stable vertices, and two consecutive vertices are deleted extremely rarely. In other words, the space between two consecutive deleted vertices is sufficiently large, i.e., there are no *bursts* of synchronization errors. Separation of synchronization errors within a codeword may be also achieved by interleaving, i.e., by proper indexing of vertices. Consequently, we consider a variant of the channel described above in which the number of consecutive synchronization errors is restricted and operates on individual runs of binary sequences. Here, we define the term *run* in a binary sequence as an occurrence of k consecutive, identical symbols.

We now define a channel with the parameter s_d, which is the maximum number of consecutive deletions. We denote this channel C as (p_d, s_d). It is a special case of the channel we introduced in [13]. The probability of j consecutive deletions is $(p_d)^j$ for $j = 1,\ldots,s_d$. In each run of zeros or ones, only one of the $s_d + 1$ error events occurs, namely, (i) error-free transmission, or (ii) deletion of s_d bits, $j = 1,\ldots,s_d$. As we have shown in [13] if all the runs in $\mathbf{x}$ have lengths greater than s, then the number of runs in any output $\mathbf{y}$ produced by C is equal to the number of runs in $\mathbf{x}$. Summarizing, when transmitted through C, each run of s_t bits, $s_t > s$, results in a run of j bits, $j = s_t - s, ..., s_t$. This observation leads naturally to an encoding scheme where symbols are encoded in runs of bits. Now transmission through the synchronization-error channel can now be represented as that of transmission through a memoryless channel. That is, any synchronization-error manifests as an error in the corresponding output symbol, and does not affect other symbols. Thus, classical error-correcting codes can be used to compensate for such errors.

Let $c(x)$ denote the length of symbol x in the input alphabet X. More generally, $c(x)$ can be viewed as the cost of transmission of the symbol x**.** Let define **c**, the transmission cost vector, $\mathbf{c}=(c(x))_{x\in X}$. In such channels, we use the notion of *unit-cost capacity* [14] $C_{unit} = \max_{\mathbf{p}} I(\mathbf{p})/\mathbf{c}\mathbf{p}^T$, where:

$$I(\mathbf{p}) = \sum_{x\in X}\sum_{y\in Y} p(x)p(y\,|\,x)\log\frac{p(x)p(y\,|\,x)}{p(x)\sum_{x\in X}p(x)P(y\,|\,x)}$$

is the mutual information between the input alphabet **X** and the output alphabet **Y**, over all input distributions of **X**, and maximization is done over all input probability distribution vectors $\mathbf{p}=(p(x))_{x\in X}$. The channel transition probabilities are denoted as $p(y\,|\,x)$. For a $(p_d, 1)$ channel, $p(x\text{-}1|x)=p_d$,

$p(x|x)=1-p_d$, and other elements in the transition probability matrix $P=[p(y\mid x)]_{x\in X,y\in Y}$ are equal to zero.

It can be seen that in our transmission model, the unit-cost capacity translates to capacity per channel-use for the transmission scheme under consideration. The reason for optimizing probability of symbols in the *distribution transformer* comes from the fact that symbols have different lengths, thus the costs of transmission of different symbols are not equal. Note that the mapping from coded bits to runlengths can be generalized by parsing the input bit sequence into *m*-tuples. Then the alphabet **X** has 2^m symbols. Let us denote these symbols as follows $\mathbf{X}=\{1,2,\ldots,2^m\}$, and suppose that the lengths assigned to a symbol x is $c(x)=x+1$. Then $\mathbf{Y}=\{1,2,\ldots,2^m, 2^{m+1}\}$, and the transition probabilities can be readily found by using the above facts about the number of possible deletions in one run.

For a channel *C*, with information alphabet **X**, we can calculate an associated unit-cost capacity $C(C; |\mathbf{X}|)$, and determine optimal capacity-achieving input probabilities. The capacity achieving distribution $p(\mathbf{x})$ can be obtained by Jimbo-Kunisawa algorithm [15]. The capacity is a function of the synchronization error probability as well as the size of the input alphabet **X**, as shown in next Figure 3.

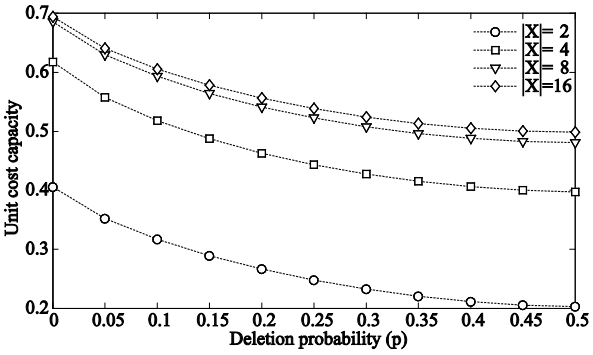


Figure 3. Capacity estimates for the channel (*p,* 1) The alphabet sizes are in the set {2, 4, 8, 16}. For alphabet size |X|, the costs of transmission of a bit is in the set {2,…|X| + 1}. The capacity increases with the size of the alphabet.

III. Numerical Results

For computing vertex stability in relation with optimization process we used 4 mesh models [16] which we refer for brevity as to A, B, C, and D: (A) Part of Athena statue, (B) Statue of girl, (C) Statue of Lucy, and (D) Statue of Neptune. These four models differ with respect to the percentage of curved and flat areas, but also in total number of vertices, (i.e. faces). Their geometric structures are unique. Athena statue is for example anisotropic 3D mesh that is obtained from 3D scanning process and contains more than 800.000 vertices, but many of them are isolated or boundary vertices. Model of Lucy, girl (B) and Neptune are also scanned models, but optimized and thus isotropic and closed. The vertex selection algorithm [17] works equally well with both homogeneous and complex meshes, as well as with open meshes with a boundary. For comparison and evaluation we used 'Optimize' modifier from 3D Studio Max 2012 application [18]. We have performed 6 different levels of optimization with the following face thresholds (FT) for the objects A, B, C and D: $FT_A = FT_B = FT_C = FT_D$ = (0,2,4,6,8,10,13). The optimization with the value FT > 20 completely destroys the geometric structure of the mesh, leaving only 5-10% of the total number of vertices. Higher maximal values are used for meshes with the larger total number of faces. Determination of face threshold limits preserving the usability of a 3D mesh is illustrated in Figure 4. on the mesh statue example.

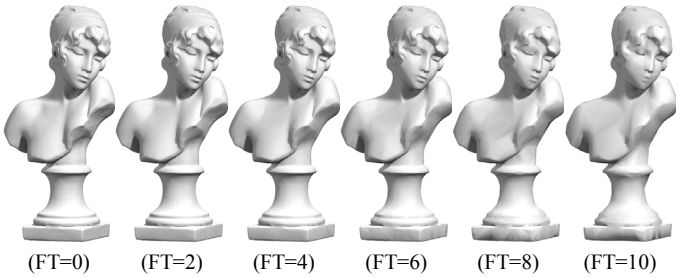


Figure 4. Perceptual degradation of the 3D mesh as a function of the face threshold level: 0,2,4,6,8,10.

We observed that the optimization face threshold higher then 8, leads to significant perceptual and geometric degradation of mesh. Experimental tests of stability 1000 vertices, selected by our algorithm compared to the randomly allocated group of 1000 vertices showed the superiority of this approach compared to random selection of vertices.

Our algorithm provides two orders of magnitude better probably of deletion compared to random selection of vertices. Another important observation is that our algorithm provides an extremely low probability of deleting two consecutive selected vertices. Actually, in 1000 selected vertices of all meshes and also all of optimization levels optimization we did not find any deleted pair of consecutive vertices. This justifying to the (*p,*1) deletion channel assumption.

A. Reconstructed bit probability of error performance

In order to demonstrate the feasibility of implementation of our encoding methodology, we conducted experiments to simulate transmission of coded information through the channel (*p,* 1). For our simulations, we chose synchronization-error probability (*p*) ranging from 0*.*01 to 0*.*05. Two codes were chosen as candidates for simulation. The codes were constructed by methods described in [12], and provide guarantees on error-correction capability under iterative error-correction decoding for the binary symmetric channel. The LDPC codes are based on arrays of permutation matrices obtained from Latin squares [19], which perform well on variety of memoryless channels. A permutation matrix is a square binary matrix that has exactly one entry 1 in each row and each column and 0's elsewhere. Our codes make use of permutation matrices that have disjoint support, and . can be obtained from Latin squares. A Latin square of size *q* (or order *q*) is a $q\times q$ array in which each cell contains a single symbol from a *q*-set *S*, such that each symbol occurs exactly once in each row and exactly once in each column. The code parameters (code length, *n*, and the number of message bits, k) are chosen as in [20]: $n = 2212$, $k = 1899$, $Reff = 0.34$ for Code 1, and $n = 848$, $k = 661$, $Reff = 0.32$ for Code 2. R_{eff} denotes the effective rate, i.e. the number of information bits

transmitted per channel use. The cost of transmitting the information-bit $c(0)=2$, and $c(1)=3$. For simplicity, in the simulations the equiprobable inputs are used. Since the average cost of a bit transmission is 2.5 (symbols of duration 2 and 3 are used half of the time in average), there runlength coding introduces a rate loss of 2/5 The Effective rate is thus $R_{eff}=(2/5)R$ where R is the rate of the LDPC code.

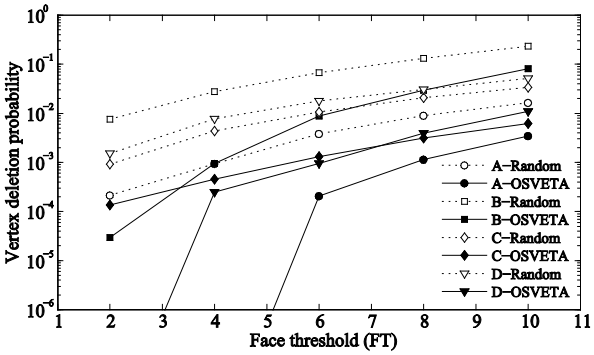


Figure 5. Probability of vertex deletion p_d as a function of the face threshold (FT) for the random (R) selection of vertices (white markers), and vertices selected by our algorithm (O - black markers) for the four objects

From Figure 5. we see that this is readily achieved for most of the objects when FT is less than 6. For the same BER, random choice of vertices is unusable because the required raw deletion criterion is met only for one object and for FT lower than 4. The proposed scheme handles even higher FT if the maximum BER requirement is relaxed.

We note that no the input distribution is not optimized, i.e., the inputs were uniformly distributed. We also note that the use of distribution transformers introduce additional loss in the effective rate. However, a detailed discussion of this is beyond the scope of this analysis. We also note that although effective rates of codes for binary alphabet are low, by using codes over higher alphabets superior guarantees on maximum achievable rates may be obtained. This problem is left for future research.

IV. DISCUSSION

Runlength coding provides conceptual simplicity as it makes the watermarking process equivalent to a memoryless channel, while iterative decoding used to recover deleted vertices yields low computational complexity. The complexity of the proposed is algorithm is the determined by the vector selection algorithm, and iterative decoding algorithm. The vector selection algorithm calculates the geometric properties of the 3D mesh (such as curvature), and involves only algebraic operations and sorting, and iterative decoding involves message passing over edges on the graph corresponding to the parity check matrix and has complexity linear in number of edges.

A number of open problem remain to be addressed such as possible modifications of the algorithm to make it robust to a malicious AWGN-deletion attack A combined deletion-AWGN channel is not well understood, and coding for such channel is nontrivial and the decoding algorithm is complex. A comparison with more complex methods such as those originating from work in [4] would help to understand various tradeoffs between rate, complexity and robustness.

REFERENCES

[1] T. Mittelholzer, "An Information-Theoretic Approach to Steganography and Watermarking," *in Information Hiding*, Lecture Notes in Computer Science vol. 1768, pp 1-16, 2000.

[2] B. Chen and G. Wornell, "Quantization index modulation: A class of provably good methods for digital watermarking and information embedding," *IEEE Transactions on Information Theory*, vol. 47, no. 4, pp. 1423–1443, May 2001.

[3] D. Coumou and G. Sharma, "Insertion, deletion codes with feature-based embedding: a new paradigm for watermark synchronization with applications to speech watermarking," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 2, pp. 153-165, June 2008.

[4] M. C. Davey and D. J. C. Mackay, "Reliable communication over channels with insertions, deletions, and substitutions," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 687–698, February 2001.

[5] R. Darazi, R. Hu, and B. Macq, "Applying spread transform dither modulation for 3D-mesh watermarking by using perceptual models," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2010)*, March 2010, pp. 1742-1745.

[6] A. Kalivas, A. Tefas, and I. Pitas, "Watermarking of 3D Models Using Principal Component Analysis," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2003)*, April 2003, vol. 5, pp. 676-679.

[7] B. Vasic, "Ordered Statistics Vertex Extraction and Tracing Algorithm (OSVETA)," *Advances in Electrical and Computer Engineering*, vol. 12, no 4, pp. 25 -32, November 2012.

[8] M. Meyer, M. Desbrun, P. Schröder, and A.H. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds," *Visualization and Mathematics III*, ED-12, pp. 35 - 57, August 2003.

[9] D. Cohen-Steiner and J. M. Morvan. "Restricted Delaunay triangulations and normal cycle.", *Proceedings of the 19th annual ACM symposium on Computational geometry (SCG '03)*, June 2003, pp. 312 - 321.

[10] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Levy, and M. Desbrun, "Anisotropic polygonal remeshing," *ACM Transactions on Graphics*, vol. 22, no 3, pp. 485 - 493, July 2003.

[11] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen, "Decimation of triangle meshes," *Proceedings of the 19st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 1992)*, July 1992, pp. 65 – 70.

[12] B. Vasic, O. Milenkovic, and S. McLaughlin, "Scrambling for nonequiprobable signalling," *IEEE Electronics Letters*, vol. 32, no. 17, pp. 1551–1552, Augist 1996.

[13] A. R. Krishnan and B. Vasic, "Coding for correcting insertions and deletions in bit-patterned media recording," *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '11)*, December 2011, pp 1-5.

[14] S. Verdu, "On Channel Capacity per Unit Cost," *IEEE. Transactions on Information Theory*, vol. 36, no. 5, pp. 1019–1030, September 1990.

[15] M. Jimbo and K. Kunisawa, "An iteration method for calculating the relative capacity," *Information and Control*, vol. 43, no. 2, pp. 216–223, November 1979.

[16] Online free 3D model "3DModelFree.com" [Online] Available: http://down.3dmodelfree.com

[17] B. Vasic, OSVETA - MatLab Software, [Online] Available: http://www.batavasic.com/research/OSVETA.zip

[18] M. Kauffman, "Optimizing Your Autodesk® 3ds Max® Design Models for Project Newport", [Online] Available: http://au.autodesk.com/?nd=material&session_material_id=6296, pp:6.

[19] D. V. Nguyen, S. K. Chilappagari, B. Vasic, and M. W. Marcellin, "On the construction of structured LDPC codes free of small trapping sets," *IEEE. Transactions on Information Theory* (to appear).

[20] B. Vasic, and B. Vasic, "Simplification Resilient LDPC-Coded Sparse-QIM Watermarking for 3D-Meshes," *Accepted for publication at IEEE Transactions on Multimedia*, (in press) [Online] Available: http://www.batavasic.com/research/ieee_2013.pdf