J. Vis. Commun. Image R. 21 (2010) 17–28

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

3D Model compression using Connectivity-Guided Adaptive Wavelet Transform built into 2D SPIHT

Kıvanç Köse [a,*], A. Enis Çetin [a], Uğur Güdükbay [b], Levent Onural [a]

[a] Department of Electrical and Electronics Engineering, Bilkent University, 06800 Bilkent, Ankara, Turkey
[b] Department of Computer Engineering, Bilkent University, 06800 Bilkent, Ankara, Turkey

ARTICLE INFO

Article history:
Received 20 October 2008
Accepted 21 September 2009
Available online 29 September 2009

Keywords:
SPIHT
Mesh compression
Adaptive wavelet transform
Projection
Image compression
Static mesh
Connectivity coding
Progressive mesh representation

ABSTRACT

Connectivity-Guided Adaptive Wavelet Transform based mesh compression framework is proposed. The transformation uses the connectivity information of the 3D model to exploit the inter-pixel correlations. Orthographic projection is used for converting the 3D mesh into a 2D image-like representation. The proposed conversion method does not change the connectivity among the vertices of the 3D model. There is a correlation between the pixels of the composed image due to the connectivity of the 3D mesh. The proposed wavelet transform uses an adaptive predictor that exploits the connectivity information of the 3D model. Known image compression tools cannot take advantage of the correlations between the samples. The wavelet transformed data is then encoded using a zero-tree wavelet based method. Since the encoder creates a hierarchical bitstream, the proposed technique is a progressive mesh compression technique. Experimental results show that the proposed method has a better rate distortion performance than MPEG-3DGC/MPEG-4 mesh coder.

1. Introduction

Three-dimensional (3D) models of real life objects have already been extensively used in computer graphics. 3D models are also used in many other areas of science, as well (e.g., computational electromagnetics [1]). There is an increasing demand for more realistic and complex scene models as a consequence of increasing capabilities of related hardware.

Storage and transmission of complex 3D models are difficult. We should exploit the correlation among the mesh vertices of a realistic 3D model representation to simplify the storage and transmission of these models. The main goal here in lossy compression is to find a way of exploiting these redundancies to compress the 3D models to smaller data sizes while creating the least distortion.

There are several methods for the compression of 3D models in the literature [2–4]. These methods can be classified into two groups: *single-rate compression* and *progressive mesh compression* [3]. In single-rate compression schemes, the entire 3D mesh is compressed to a single data stream. The model is decodable if all of the data stream is received. On the other hand, a multiresolution representation of 3D meshes [5] is used in progressive mesh compression. In most of the progressive mesh compression techniques, the original model is first partitioned into a base mesh and refinement coefficients; e.g., using wavelet-based approaches [5,6]. First, the base mesh is coded and sent to the receiving party and the refinement coefficients are sent afterwards. Thus, progressive mesh representations enable the user to obtain different resolutions of the model corresponding to different sizes of the code stream. At the decoder side, first the base mesh is decoded, and then, the base mesh is updated to higher resolutions using the new-coming refinement coefficients.

Gu et al. [7] propose an entirely different approach, called Geometry Images (GI). In this approach, the 3D model is converted to a 2D image-like representation. First, the 3D model is remeshed to obtain a semi-regular mesh. Then, the images are calculated using mesh parameterization. The 3D mesh should be cut and opened homeomorphic to a disc before parameterization. All these three steps are computationally-intensive tasks. In the last step, mesh is transformed to an image by solving several linear equations. This approach makes it possible to use any image processing algorithm on 3D models. This idea is also extended to dynamic meshes so that video processing algorithms are applied on 3D models [8].

In this paper, we present a framework for compressing geometry information of 3D models using image processing techniques like 2D wavelet transform as in Set Partitioning in Hierarchical Trees (SPIHT) [9,10] or JPEG2000 [11] *Edgebreaker* [12] is used for the compression of the connectivity data. Starting from a random face of the model, Edgebreaker traverses the faces of the model

* Corresponding author. Fax: +90 3122664192.
E-mail addresses: kkivanc@ee.bilkent.edu.tr (K. Köse), cetin@ee.bilkent.edu.tr (A.E. Çetin), gudukbay@cs.bilkent.edu.tr (U. Güdükbay), onural@ee.bilkent.edu.tr (L. Onural).

1047-3203/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jvcir.2009.09.007

and outputs the sequence of movements it performs. This sequence of movements is encoded at the last step of the Edgebreaker.

The proposed algorithm is composed of two main parts: (i) converting the 3D models to a 2D image and (ii) compressing that image using an image coder that uses Connectivity-Guided Adaptive Wavelet Transform [13]. The proposed static mesh compression framework is summarized in Fig. 1. $\mathbf{M_O}$ represents the original 3D model, $\mathbf{I}_1$ and $\mathbf{W}_{\mathbf{I}_1}$ are the 2D representation of the 3D model and its wavelet transformed version, respectively. $\mathbf{Str_{Trans}}$ and $\mathbf{Str_{Receive}}$ are the transmitted and the received SPIHT bitstreams, respectively. $\mathbf{W}_{\mathbf{I}_{Rec}}$ and $\mathbf{I_{Rec}}$ are the reconstructed versions of $\mathbf{W}_{\mathbf{I}_1}$ and $\mathbf{I}_1$, respectively. The reconstructed mesh is represented by $\mathbf{M}_{Rec}$.

This paper is organized as follows: the proposed static mesh compression framework including the projection operation and the Connectivity-Guided Adaptive Wavelet Transform are explained in Sections 2 and 3, respectively. The compression method and decoding are given in Sections 4 and 5, respectively. The simulation and compression results are given in Section 6.

2. Static mesh compression framework

The proposed mesh coding algorithm uses orthographic projection to convert the 3D model to an image whose pixel values are related to the respective coordinates of the vertex point to the projection plane. Thus 2D signal processing methods become applicable to a given mesh. In this section, the 3D mesh representation is given and the orthographic projection to obtain the 2D image-like representation of the 3D mesh is described.

2.1. 3D Mesh representation

A 3D mesh can be considered as an irregularly sampled discrete signal, this means that the signal is only defined at vertex positions. In the proposed approach, the mesh is converted into a regularly sampled 2D signal by orthographic projection onto a chosen plane whose sampling matrix is known.

The 3D mesh is formed by geometry and connectivity information. The geometry of the 3D mesh is constituted by vertices in $\mathbf{R}^3$. It is defined as, $\mathbf{V}' = (V'_x, V'_y, V'_z)^T \in \mathbf{R}^3$ [14]. The vertices of the original 3D model are represented by $\mathbf{v}'_i = (V_{x'_i}, V_{y'_i}, V_{z'_i})^T, i = 1, \ldots, v$ where v is the number of the vertices in the mesh.

First, the space in which the 3D model was defined is normalized in $\mathbf{R}^3[-0.5, 0.5]$ as,

$$\mathbf{V} = (V_x, V_y, V_z)^T = \alpha \mathbf{V}', \quad \alpha = (\alpha_x, \alpha_y, \alpha_z) \in \mathbf{R}^3, \tag{1}$$

and the normalized mesh vertices are represented as,

$$\mathbf{v}_i = (V_{x_i}, V_{y_i}, V_{z_i})^T = (\alpha_x V_{x'_i}, \alpha_y V_{y'_i}, \alpha_z V_{z'_i}), \quad i = 1, \ldots, v. \tag{2}$$

Normalization is a necessary step to make the algorithm independent from the size of the 3D model. Connectivity information is represented as triplets of vertex indices. Each triplet corresponds to a face of the mesh [14]. Faces of the mesh are,

$$\mathbf{F}_i = (\mathbf{v}_a, \mathbf{v}_b, \mathbf{v}_c)^T, \quad i = 1, \ldots, f,$$
$$a, b, c \in \{1, \ldots, v\}, \quad a \neq b \neq c, \tag{3}$$

where f is the number of the faces. So a mesh in $\mathbf{R}^3$ can be represented as,

$$\mathbf{M} = (\mathbf{V}, \mathbf{F}), \tag{4}$$

where $\mathbf{V}$ is the set of vertices of the mesh and $\mathbf{F}$ is the set of faces of the mesh.

2.2. Projection of the 3D model onto 2D

Orthographic projection is used to obtain a 2D representation of 3D models. The vertices of the 3D mesh are projected onto a plane, defined as $\mathbf{P}(u, w)$, where u and w are the orthogonal vectors defining the projection plane. The decision on the orientation of the projection plane is made by looking at the histogram distribution of the vertices on a given plane. The histogram shows the number of vertices that are projected on each pixel. The plane that has the highest number of distinct non-zero values in its histogram has the most projected vertices; thus, it is selected as the projection plane (see Fig. 2). This check should be done on infinitely many different orientations of projection planes. This brings a huge cost to the algorithm; thus, we used the *XY*, *XZ*, and *YZ* planes as the candidate projection planes for the sake of reducing the computational cost.

The selected projection plane $\mathbf{P}$ is discretized using the quincunx sampling matrix [15]

$$\mathbf{S}_{quinc} = \begin{pmatrix} T & T/2 \\ 0 & T/2 \end{pmatrix}. \tag{5}$$

Experimental results show that using quincunx sampled projection planes results in small error under the projection of triangular meshes. After the quincunx sampling the projected mesh vertices are lined up in a rectangular arrangement to form an image representation.

The projection of a mesh vertex depends on two parameters: the 3D coordinate of the vertex and the orientation of the selected projection plane. In many aspects, the newly formed 2D representation does not have any difference from a grayscale image.

Mesh vertices $\mathbf{v}_i$ have orthogonal projections $\check{\mathbf{v}}_i$ onto the projection plane $\mathbf{P}$. The perpendicular distance between a mesh vertex $\mathbf{v}_i$ and the projection plane $\mathbf{P}$ is represented by $\mathbf{d}_i$ in which i represents the index of the vertex. A projection of an object is illustrated in Fig. 3. As illustrated in the figure, the respective pixel location of a mesh vertex is determined by the projection of the vertex onto the projection plane. The pixel values are the perpendicular distances of the vertices to the projection plane, $\mathbf{d}_i$.

The determination of vertex-to-grid point correspondence is the most crucial task in the projection operation. The vertices that can be assigned to a grid point $\mathbf{n} = [n_1, n_2]$ form a set of indices J defined by:

$$J = \{i \,|\, |\check{\mathbf{v}}_i - \mathbf{S}\,\mathbf{n}^T| < T/2, \forall\, n_1, n_2\}, \tag{6}$$

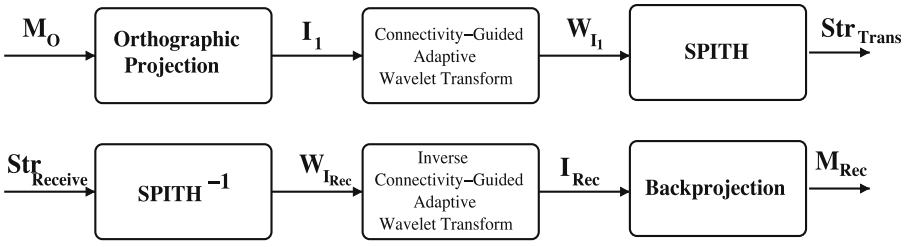

Fig. 1. The proposed framework for mesh compression.

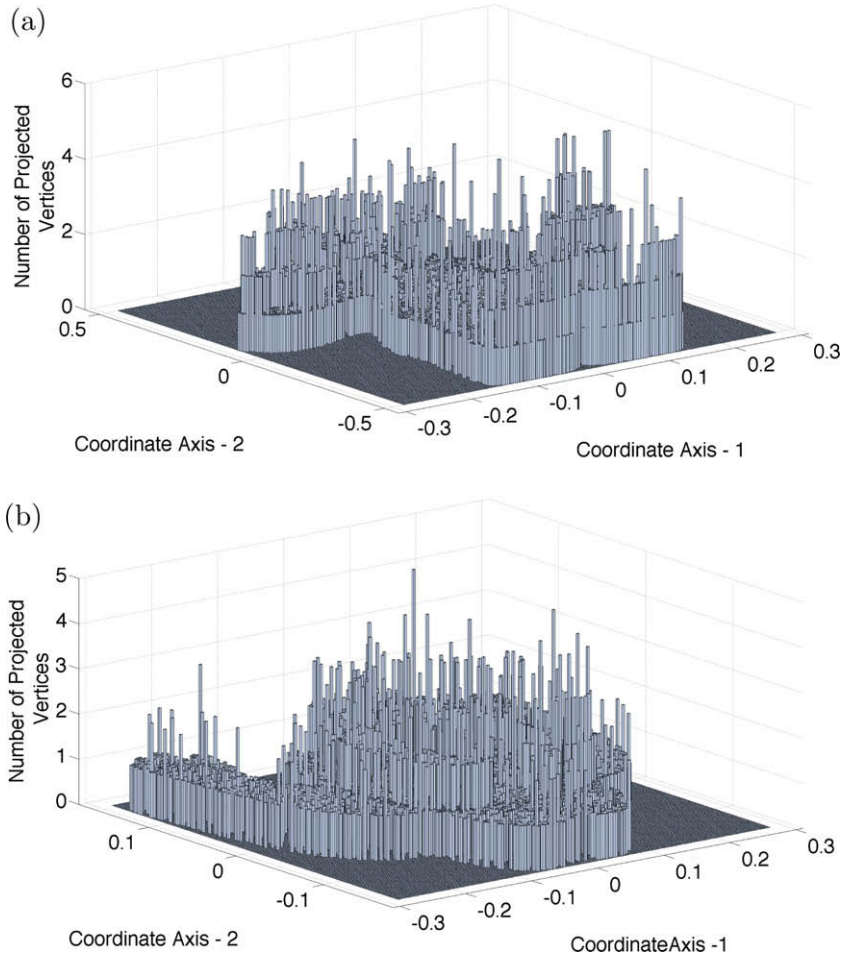

Fig. 2. The number of non-zero bins in the histogram of a projection plane gives the number of vertices that can be projected without coinciding each other. The values in the bins show how many vertices are projected to that bin. If the value is one then we have only one vertex projected to that bin. If the value of the bin is zero then no vertex is projected to that bin. If the value of the bin is greater than one then we have multiple projected-coincided-vertices in that bin. (a) *XY* plane is used as the projection plane. Four thousand and fifty-three bins have non-zero values (b) *YZ* plane is used as the projection plane. Three thousand eight hundred and thirty-seven vertices have non-zero values.

where **S** is any sampling matrix and $\mathbf{n} = [n_1, n_2]$ represents the indices of the discrete image, as shown in Fig. 3. The sampling density can be changed using the *detail level* parameter. The resolution increases with the detail level parameter. 3D Models with many vertices should be projected onto a plane whose grid resolution is high.

Here is an example of the projection operation: assume that a vertex i at $\mathbf{v}_i = (1,2,4)$ in the Cartesian coordinates is projected onto the *XY* plane. Then the projection $\tilde{\mathbf{v}}_i$ is equal to $(1,2)$ and the corresponding pixel values d_i is equal to 4. The location of the pixel on the projection image is found using $\tilde{\mathbf{v}}_i$ and the selected sampling period as given by Eq. (6).

Employing the methods described above, we transform a 3D mesh to a 2D image. The first image (projection image),

$$\mathbf{I}_1[n_1, n_2] = \begin{cases} \mathbf{d}_i, & i \in J, \\ 0, & \text{otherwise}, \end{cases} \tag{7}$$

stores the perpendicular distances of the vertices to the respective grid points on the selected planes. The second image (map image),

$$\mathbf{I}_2[n_1, n_2] = \begin{cases} i, & \mathbf{d}_i = \mathbf{I}_1[n_1, n_2], \\ 0, & \text{otherwise}, \end{cases} \tag{8}$$

stores the indices of the vertices.

The projection image is wavelet transformed using the *Connectivity-Guided Adaptive Wavelet Transform* [13]. The transformed image is then encoded using SPIHT [9] or JPEG2000 [11]. The map image is converted to a list of indices. This list is differentially coded and sent to the decoder side.

Eqs. (7) and (8) define the pixel-vertex correspondence for each vertex. Sometimes more than one vertex may be projected to the same pixel in the image-like representation. One of these vertices is chosen for the calculation of the pixel value and the others are discarded. We use a priority queue to determine the projection order for the vertices. The priority of a vertex is defined according to the number of neighbors connected to the vertex, called *valence*. The more the valence of a vertex, the more vertices can be predicted from it. In case of multiple vertex projections on the same pixel, the vertex with the highest valence is projected and the others are discarded. Another approach would be projecting the discarded vertices to the nearest, empty sampling points on the projection plane. Since the projection image is sparse, it is highly possible that some empty sampling points around the exact projection location exist. This approach may result in a minor error, due to imprecise positioning of the projection of the vertex.

More vertices can be handled either by increasing the number of the samples (pixels) taken on the projection plane or increasing the number of the projection planes. In the proposed approach, we decided to use one densely-sampled projection plane. Even in this case, some vertices of the mesh may coincide and cannot be

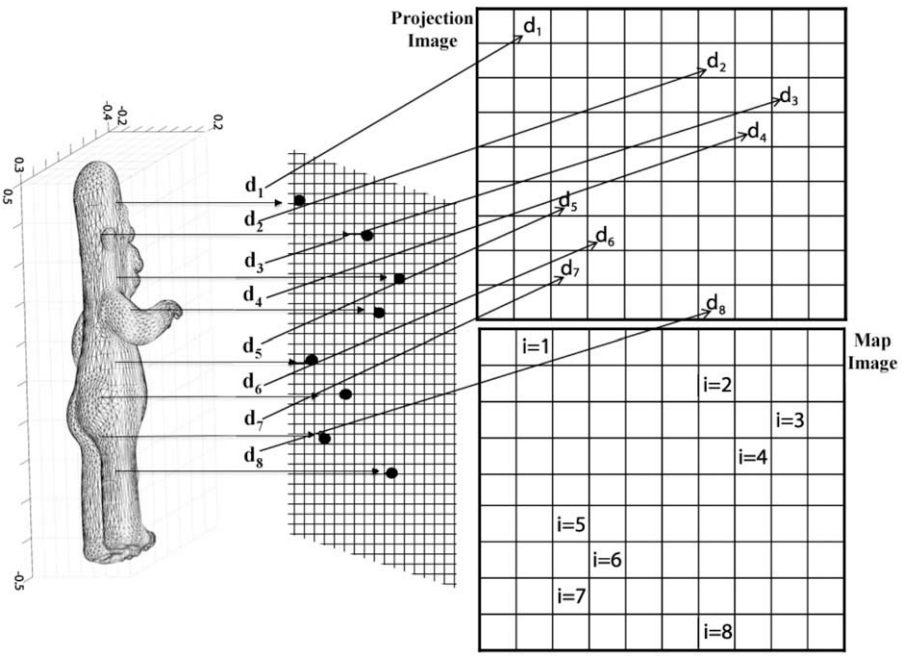

Fig. 3. The illustration of the orthographic projection of a 3D object.

projected. These lost vertices can be interpolated using connectivity-guided interpolation (cf. Section 5).

3. Connectivity-Guided Adaptive Wavelet Transform

Since the projection image (Fig. 4(a) and (b)) is a grayscale image, it can be coded using any available grayscale image coder. In the proposed algorithm, this data on a regular grid, that represents the mesh data, is transformed into wavelet domain using an adaptive wavelet transform. The idea of adaptive wavelets [16] is a well known and proved to be a successful tool in image coding. Adaptive wavelet transform is superior to its non-adaptive counterparts by exploiting the directional neighborhood information between pixels. Thus, we define an adaptive scheme. The proposed adaptive scheme takes the connectivity of the mesh into account to better exploit the neighborhood information of the mesh vertices.

The neighborhood relation in the projection images is not the same as in common grayscale images. In ordinary grayscale images, it is natural to predict one pixel from its spatial neighbors. In projection images, it is highly possible that the values of the neighboring pixels are not coming from neighboring vertices. Hence, their values are less-even not-correlated. Although the 2D projection image of a 3D mesh seems like a salt and pepper image, there is a correlation between the non-zero pixels due to the connectivity of the mesh.

There is no general 2D compression algorithm that works on salt and pepper images. Our aim here is to utilize the correlation between the non-zero pixel values of the salt and pepper like projection image. For this purpose, we modified the wavelet transformation stage of the SPIHT encoder and proposed *Connectivity-Guided Adaptive Wavelet Transform* that uses the connectivity information of the mesh and predict the pixel values from their connected neighbors. In the literature, there is no single 2D compression algorithm that is applicable to all types of 2D data. The main motivation to use 2D projection of 3D meshes is to utilize 2D image compression algorithms with minor modifications for the compression of 3D meshes.

First, $\mathbf{I}_1[n_1,n_2]$ image is wavelet transformed in the horizontal direction. This transformation results in two new images,

$$\begin{aligned}\mathbf{I}_{11}[n_1,n_2] &= \mathbf{I}_1[n_1,2n_2],\\ \mathbf{I}_{12}[n_1,n_2] &= \mathbf{I}_1[n_1,2n_2+1].\end{aligned} \tag{9}$$

The width of the downsampled images, $\mathbf{I}_{11}$ and $\mathbf{I}_{12}$ are half of the original image $\mathbf{I}$. One-level lazy wavelet transformed version of $\mathbf{I}_1$ is constructed by apposing the downsampled images, $\mathbf{I}_{11}$ and $\mathbf{I}_{12}$ as

$$\widehat{\mathbf{W}}_{I_1}[n_1,n_2] = [\mathbf{I}_{11}|\mathbf{I}_{12}]. \tag{10}$$

The resulting image have the same size as the original image. $\mathbf{I}_{11}$ is the lower subband image that will be used during the wavelet transformations at the later stages. The same transformation is applied to $\mathbf{I}_2[n_1,n_2] = i, i \in \{1,\ldots,v\}$ that results in sub images, $\mathbf{I}_{21}$ and $\mathbf{I}_{22}$. Then, a list of neighbors, $nlist(j)$, $j = 1,\ldots,v$, is constructed from the connectivity list. Each element j of the list $nlist(j)$ stores the indices of the connected neighbors of the vertex with index j. The list of vertices that are in image $\mathbf{I}_{21}$ and $\mathbf{I}_{22}$ are $list_1$ and $list_2$, respectively.

The pixel values of the vertices in $list_2$ are predicted from the vertices in lower subband image. In other words, for each element k of $list_2$, a prediction is made from its neighbors in $\mathbf{I}_{11}$ image ($list_1$). Valid neighbors of k are

$$nlist_{valid_k} = nlist(k) \cap list_1. \tag{11}$$

After finding the list of valid neighbors of a vertex, the next step is finding the pixel location of the neighbors in the lower subband. The grid location $[n_{1j},n_{2j}]$ of the jth neighbor of vertex k is found as,

$$[n_{1j},n_{2j}] = \begin{cases}[n_1,n_2], & \mathbf{I}_{21}[n_1,n_2] = nlist_{valid_k}(j),\\ 0, & \text{otherwise}.\end{cases} \tag{12}$$

The prediction of vertex $k \in list_2$ is defined as

$$\mathbf{I}_{k\ pred} = \frac{\sum_j(\mathbf{I}_{11}[n_{1j},n_{2j}])}{m}, \tag{13}$$

where $\mathbf{I}_{21}[n_{1k},n_{2k}] \in nlist_{valid_k}$ and m is the number of the elements in $nlist_{valid_k}$. The predicted pixel values become

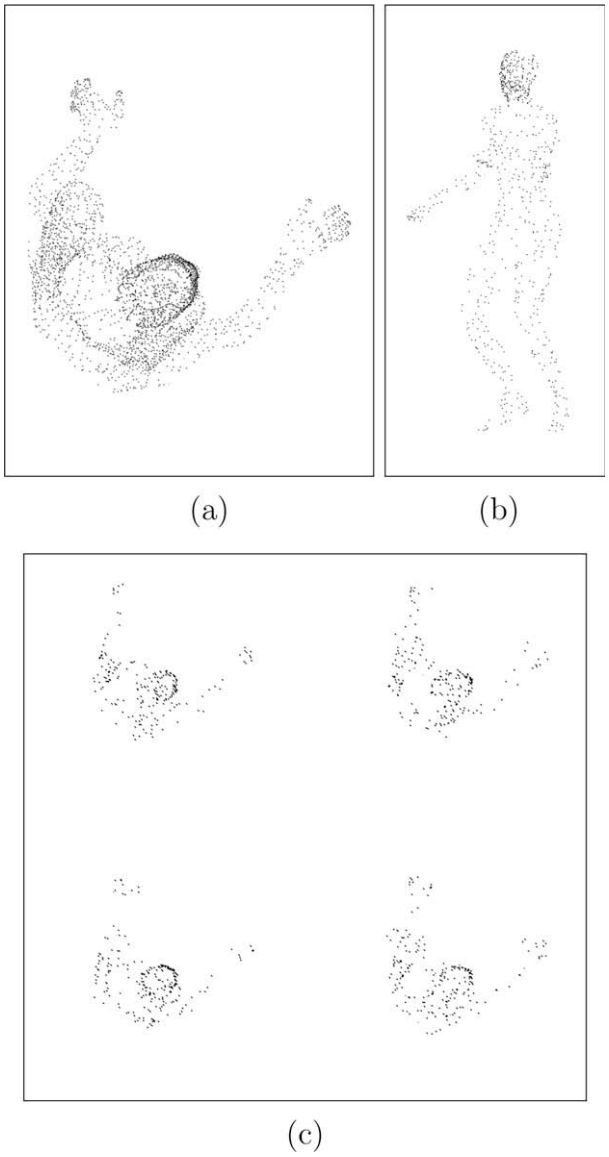

Fig. 4. The images calculated using the projection operation I_1 on (a) *XY* plane, (b) *XZ* plane. (c) After one level of wavelet transform in horizontal and vertical directions, the projection image becomes an augmented image ($W_{I_1}[n_1, n_2]$).

$$\mathbf{I}_{new_{12}}[n_1, n_2] = \mathbf{I}_{12}[n_1, n_2] - \mathbf{I}_{k\ pred},$$
$$\mathbf{W}_{I_1}[n_1, n_2] = [\mathbf{I}_{11}|\mathbf{I}_{new_{12}}], \tag{14}$$

where $\mathbf{I}_{22}[n_1, n_2] = k$. If no valid neighbors exist for a vertex, a prediction is not made. However, this situation is rare since we have a few vertices left to be predicted at the lower subbands (higher levels of transformation). Prediction and estimation are the last parts of the one level of the transform. Since the proposed transform is separable, transposing the image and applying the 1D transformation in horizontal direction is the same as applying the 1D transform in the vertical direction. In the next level, the image is transposed and the same procedure is applied on the lower subband of the image. The resulting image after two levels of transformations is given in Fig. 4(c). Typically six levels of Connectivity-Guided Adaptive Wavelet Transform (three horizontal and three vertical) are used. Since few vertices and few neighborhood relations are left in the lower subbands, using CGAWT for these subbands does not bring a significant gain. After three levels of CGAWT, ordinary lazy wavelet transform is used to decompose the image till the lowest subband. The inverse of the adaptive wavelet transform is also possible; hence, perfect reconstruction of the images is possible.

4. Connectivity-Guided Adaptive Wavelet Transform based compression algorithm

By consecutively applying the mesh-to-image transform, and Connectivity-Guided Adaptive Wavelet Transform, the mesh data is prepared for SPIHT or JPEG2000 coding. Since the resulting projection images contain mostly zero-values, these images can be coded efficiently using a zero tree wavelet coder. Thus, SPIHT is selected for coding these images. Since JPEG2000 is another widely used wavelet-based coding standard, it is used for comparison.

Before applying SPIHT or JPEG2000 coders, the transformed image is quantized. The histogram of the projection image shows that most of the pixel values are concentrated around zero in a narrow band. Thus, an 8-bit non-uniform quantization [17] whose steps are smaller around center and larger on the sides is used (e.g., Lloyd-Max Quantizer).

The quantized signal is then coded using either SPIHT or JPEG2000 coder. SPIHT coder hierarchically encodes the image. This means the lower subbands are at the initial parts of the code-stream and higher subbands follow them. As explained in Section 3, pixel information of the lower subbands is used for predicting the pixels in higher subbands. Therefore the values of the pixels-vertices-in the higher subbands can be estimated even if the part of the stream corresponding to these subbands are lost. After reconstructing the model from the leading bits, it is possible to refine it using the later parts of the stream (progressive compression).

When the quantized image is fed into JPEG2000 coder, it further quantizes the subbands of the tiles to the user-specified levels. As in SPIHT coder, JPEG2000 also has a hierarchical structure. The tiles that correspond to the lower subbands of the transformed image can be transmitted first. The reconstruction of the mesh can be done using these parts plus zero paddings instead of the other tiles. As the tiles corresponding to the higher subbands received, the reconstructed mesh can be refined.

After the bitstream is obtained by either of the encoders, it should be bitstream encoded. *gzip* software is used as a bitstream encoder [18]. It is an implementation of the *DEFLATE* algorithm [19] which uses a combination of LZ77 algorithm and Huffman coding. For comparison purposes, both the original vertex list and the SPIHT bitstream are compressed using *gzip* software.

4.1. Coding of the map image

The pixel values of the map image are the projected vertex indices (Eq. (8)). The most important issue in the compression of these images is that it should be lossless; thus, no quantization can be done on the map image. The pixel values have a wide range and they are equiprobable. Thus, a coding structure like Huffman or Lempel-Ziv is not appropriate for this kind of data.

For the compression of these map images, an algorithm based on differential coding is proposed. The basic assumption of the proposed algorithm is that near pixels of the projection image represent near vertices, so the distance between two consecutive vertices is small. This assumption is supported by Edgebreaker, the connectivity coder used in our implementation. It creates strip of faces and reindexes the mesh vertices in the order of traversal. Since the vertices forming a face are conquered consecutively, most of the time they have consecutive indices.

A list of vertex coordinates called *LCoor*, which stores the vertex locations on the image-like representation, is created. The *i*th entry

of the list $LCoor(i)$, which stores the pixel location of vertex i, is defined as

$$LCoor(i) = \begin{cases} [n_1, n_2], & \mathbf{I}_2[n_1, n_2] = i, \\ 0, & \text{otherwise}, \end{cases} \quad (15)$$

where $i = 1, \ldots, v$. Then the list of vertex coordinates is differentially coded. In the perfect case, all K vertices of the mesh are projected. Since this is not the case in general, the non-zero entries of the coordinate list are found as

$$q(j) = \{i | LCoor(i) \neq [0,0] \quad \text{and} \quad j = 1, \ldots, G\}, \quad (16)$$

where G is the number of non-zero entries of the list of vertex coordinates. A zero entry at the $LCoor(i)$ means that, ith vertex of the model is not projected. The positions of the projected vertices are updated by a first order linear predictor as

$$LCoor(q(j+1)) = LCoor(q(j+1)) - LCoor(q(j)). \quad (17)$$

In this way, an updated version of the coordinate list is obtained. The mean of this list is around 0 and variance is concentrated around the mean. Thus, the predicted version of the coordinate list can be compressed more efficiently. Then, the list is converted into a bitstream and sent to the receiver. The receiver does the inverse of the described encoding procedure for decoding.

5. Decoding

The decoding operation is the reverse of the encoding process. The received stream is first decoded by the respective coder it was encoded [9,11]. The wavelet transformed images are obtained at the end of this operation. Then, the inverse wavelet transform with connectivity-guided adaptive prediction stage is applied on those images. In this way, projection images are reconstructed. The final stage of reconstruction is the back-projection operation. The projected vertices are back-projected to the 3D space using their respective grid location and pixel values.

Since more than one vertex may project onto the same grid location, some of the vertices are discarded during the projection operation Section 2.2. The values of the discarded vertices are estimated from their projected neighbors as

$$\mathbf{v}_i = \frac{\sum_k \mathbf{v}_k}{k}, \quad (18)$$

where k is the number of elements of $n_{list}(i)$. The connectivity list is used to find the neighbors of the lost vertices, and thus to predict the values of them (Eq. (18)).

6. Results

We used several different models in our simulations. The original Cow model has 2904 vertices and a lossless compressed data size of 27.2 KB. The Nine-Handle Torus model is composed of 9392 vertices with 165 KB lossless compressed data size. The Homer Simpson model is composed of 4930 vertices with 98 KB lossless compressed data size. The Horse model is composed of 48,485 vertices with 937 KB lossless compressed data size. We also used two large models: Happy Buddha and Hand. Happy Buddha model is composed of 543,652 vertices with 9.161 MB lossless compressed data size. Hand model is composed of 327,323 vertices with 5.484 MB lossless compressed data size.

The distortion level of the reconstructed 3D mesh is checked using some quantitative tools like METRO [20] and its visual version MESH [21]. *Mean Square Error* (MSE) and *Hausdorff distance* [23] between the original and the reconstructed mesh are mostly used error measures in the literature. The *Hausdorff distance* between two given sets of points $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$ is defined as

$$d(A,B) = max\{\max_{a \in A} \min_{b \in B} |b - a|, \max_{b \in B} \min_{a \in A} |a - b|\}, \quad (19)$$

where $a - b$ is the Euclidean distance between two points. Thus, the Hausdorff distance is the maximum distance of a set to the nearest element of the other set.

Projection image coding results are significantly affected by the choice of wavelet basis. Therefore, wavelet transform basis, which can decompose the image-like meshes efficiently, should be investigated. Among several wavelet bases, such as *lazy, Daubechies-4, and Biorthogonal-4.4* [22], we obtained the best results using lazy wavelet basis with Connectivity-Guided Adaptive Transform stages. There are two reasons for this: (i) most of the neighboring pixels in the image-like representation are not neighbors of each other in the original mesh representation, and (ii) the image-like meshes contain isolated points. A wavelet transform basis with a small support (lazy wavelet basis) gives better compression results than those with larger support regions.

The comparative results of using the proposed Connectivity-Guided Adaptive Wavelet Transform and the non-adaptive wavelet transforms [24] are given in Table 1. Table 2 shows the results of the SPITH compression of the Lamp model, using non-adaptive and adaptive wavelet transforms. Table 2 shows that adaptive wavelet structure produces better results than non-adaptive lazy filterbanks. Having the same data size, the adaptive structure causes less distortion in the reconstructed mesh. This means that better compression can be obtained for the same distortion rate. However, the computational power needed for the adaptive structure is higher since multiple searches must be performed in con-

Table 1
Compression results for the Cow model. The used bitstream parameter shows what percentage of the encoded bitstream is used in the reconstruction of the original model. The detail level parameter defines grid resolution of the projection image (the size of the projection image). As the detail level increases, the grid points of the projection plane gets closer (the projection image gets larger).

Filters	Used (%)	Detail level	Size (KB)	Original to quantized Hausdorff distance	Original to reconstructed Hausdorff distance
Lazy (non-adaptive)	30	3.0	9.51	0.013993	0.014547
Lazy (non-adaptive)	60	3.0	10.4	0.013993	0.014219
Lazy (adaptive)	60	3.0	10.6	0.013993	0.013936
Haar	40	3.0	11.5	0.013993	0.016085
Haar	60	3.0	13.8	0.0139930	0.014102
Daubechies-4	40	3.0	12.0	0.013993	0.018778
Daubechies-10	40	3.0	12.1	0.013993	0.014194
Biorthogonal-4.4	60	4.0	16.0	0.009446	0.014549

Table 2
Compression results for the Lamp model using SPIHT with lazy wavelet filterbank. The results in each row have the same detail level.

Size non-adaptive (KB)	Size adaptive (KB)	Original to reconstructed Hausdorff distance (non-adaptive)	Original to reconstructed Hausdorff distance (adaptive)
9.05	10	0.069620	0.023367
9.64	10.6	0.050859	0.019023
14.7	16.5	0.027902	0.018638
16.4	18.6	0.029870	0.018551
17.8	20.5	0.025991	0.018425
17.9	19.8	0.033990	0.011436
19.1	21.6	0.041133	0.012037

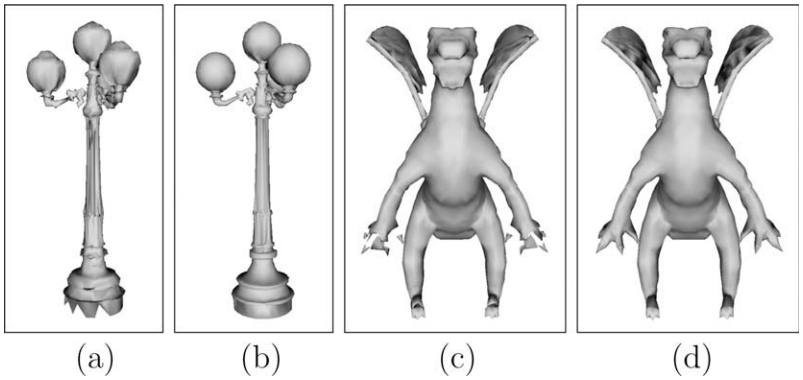

Fig. 5. The qualitative comparison of the meshes reconstructed without adaptive prediction (a and c) and with adaptive prediction (b and d). Lazy wavelet basis is used. The meshes are reconstructed using 60% of the bitstream with detail level = 5 in the Lamp model and 60% of the bitstream with detail level = 5 in the Dragon model.

nectivity data. This search operation adds complexity to the algorithm but enables better predictions.

Fig. 5 gives a qualitative comparison between the reconstructed meshes with and without adaptive prediction. The quality difference of the reconstructed meshes is especially noticeable at sharp features, such as the paws in the dragon mesh and the base of the lamp mesh.

Detail level and the bitstream used are the two other factors that are affecting the Hausdorff distance between the original and the reconstructed models (Table 1). Detail level parameter affects the size of the projection image and the closeness of the grid samples on the projection image. As the detail level parameter increases, the projection image size also increases, and thus, the grid samples become closer in distance. This increases the quality of the reconstructed model but, decreases the compression ratio. If the full size of the SPIHT bitstream is used for reconstruction, the model can be perfectly reconstructed. The used bitstream parameter shows what percentage of the SPIHT bitstream is used in the reconstruction of the original model. The quality of the reconstructed 3D model is proportional to the bitstream used for reconstruction. A lower length bitstream, which means more compression, naturally results in a higher distortion (Table 2).

The issue of how much of the SPIHT stream should be taken and what should be the quantization levels of JPEG2000 tiles are also closely related to the detail level parameter used in the orthogonal projection operation. If the detail level is low, either the percentage of the used bit-stream must be increased or finer quantization should be applied to reconstruct the 3D mesh without much distortion.

The number of the used wavelet decomposition levels is another factor affecting the compression ratio. Increasing the number of wavelet decomposition levels as much as possible brings in better compression in SPIHT. This is because the data contains mostly insignificant pixels and as we get higher on the scales, the chance of having larger zero-trees becomes higher. Larger zero-trees leads to better compression. However, increasing the decomposition level also increases the computational cost due to the computation of the adaptive wavelet transform. This situation adversely affects the performance in terms of computational complexity.

Another drawback of increasing decomposition levels is the degraded neighborhood relationship information between the pixels at lower subbands. As we go to lower subbands, it is less likely to find connected neighbors for a vertex. Thus, the prediction quality degrades as we go to lower subbands; i.e., the adaptivity in wavelet transform is lost.

After the Connectivity-Guided Adaptive Wavelet Transform, quantization takes place in the framework. Nonuniform quantization improves the rate-distortion values of SPIHT and JPEG2000 since the residuals are concentrated around a mean with a low variance. Fig. 6 gives a qualitative comparison between the original and the reconstructed meshes that are compressed using SPIHT (a and b) and JPEG2000 (c). The mesh quality compressed with

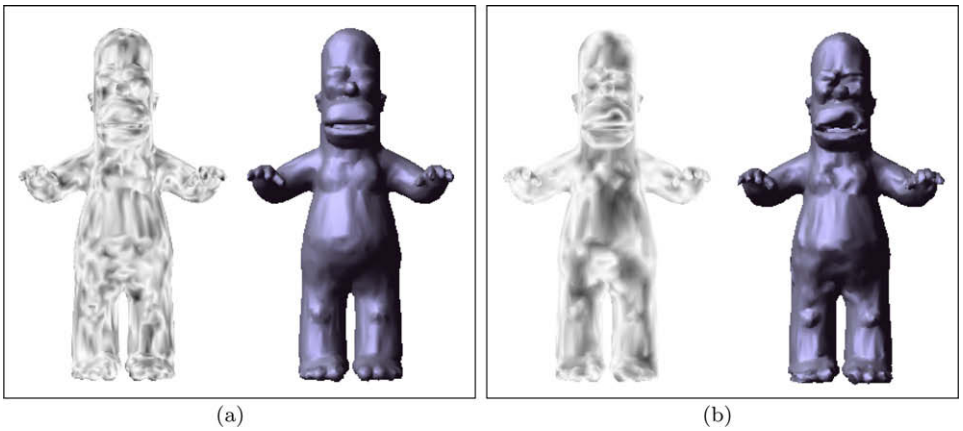

Fig. 6. Distortion measure between the original (the images on left side of the figures) and the reconstructed Homer Simpson models using MeshTool [21] software. (a) SPIHT at 11 bit per vertex (bpv); (b) JPEG2000 at 10.5 bpv. The grayscale colors on the original image show the distortion level of the reconstructed model. Darker colors mean more distortion.

Table 3
Compression results for the Homer Simpson model using SPIHT and JPEG2000. Hausdorff distances are measured between the original and reconstructed meshes.

SPIHT size (KB)	Hausdorff distance (SPIHT)	JPEG2000 size (KB)	Hausdorff distance (JPEG2000)
7.99	0.001529	6.58	0.076215
9.42	0.001256	9.64	0.076488
10.7	0.000627	9.28	0.076374
12.2	0.000419	12.2	0.075699

SPIHT is superior to JPEG2000 compressed mesh. The same is also observed in Table 3 that gives rate distortion values for compressed Homer Simpson model.

The reason why JPEG2000 has a lower performance than SPIHT is the tiling issue. The image is partitioned into tiles at the beginning of the JPEG2000 compression, and then, these tiles are wavelet transformed. Partitioning projection images into tiles adversely affects the neighborhood relationship information of the projected vertices. Connected vertices may fall in different tiles, and thus, the performance of CGAWT degrades. Another possibility can be taking the whole image as a single tile. In this way the neighborhood relationship problem can be solved but the compression performance of JPEG2000 decreases.

7. Discussion

In progressive mesh coders, different resolutions of the 3D model can be reconstructed from different sizes of received bitstream. Since the bitstream constructed by SPIHT coding has an hierarchical structure, different resolutions of the 3D model can also be reconstructed from SPIHT bitstream. In most progressive mesh coders, the difference between two resolutions of a 3D model is the number of vertices used to represent the object. Each new refinement coefficient causes a new vertex to be added to the model and refine its position. In this way, a higher resolution of the 3D model is obtained. In the proposed SPIHT algorithm, the difference between two resolutions comes from the distortion caused by the inexact positioning of the vertices. Using any size of the SPIHT bitstream, the 3D model with the same number vertices as the original model can be reconstructed. The new-coming parts of the bitstream fine-tune the positions of the vertices.

JPEG2000 algorithm is not implemented as a progressive encoder in our work. Different resolutions of the mesh could be reconstructed by using different quantization levels in JPEG2000. In the wavelet subband decomposition, high bands are suppressed by coarse quantization so that the lower resolution meshes are obtained. SPIHT based compression gives lower distortion than JPEG2000 based compression for nearly the same bit rates (see Table 3 and Fig. 6).

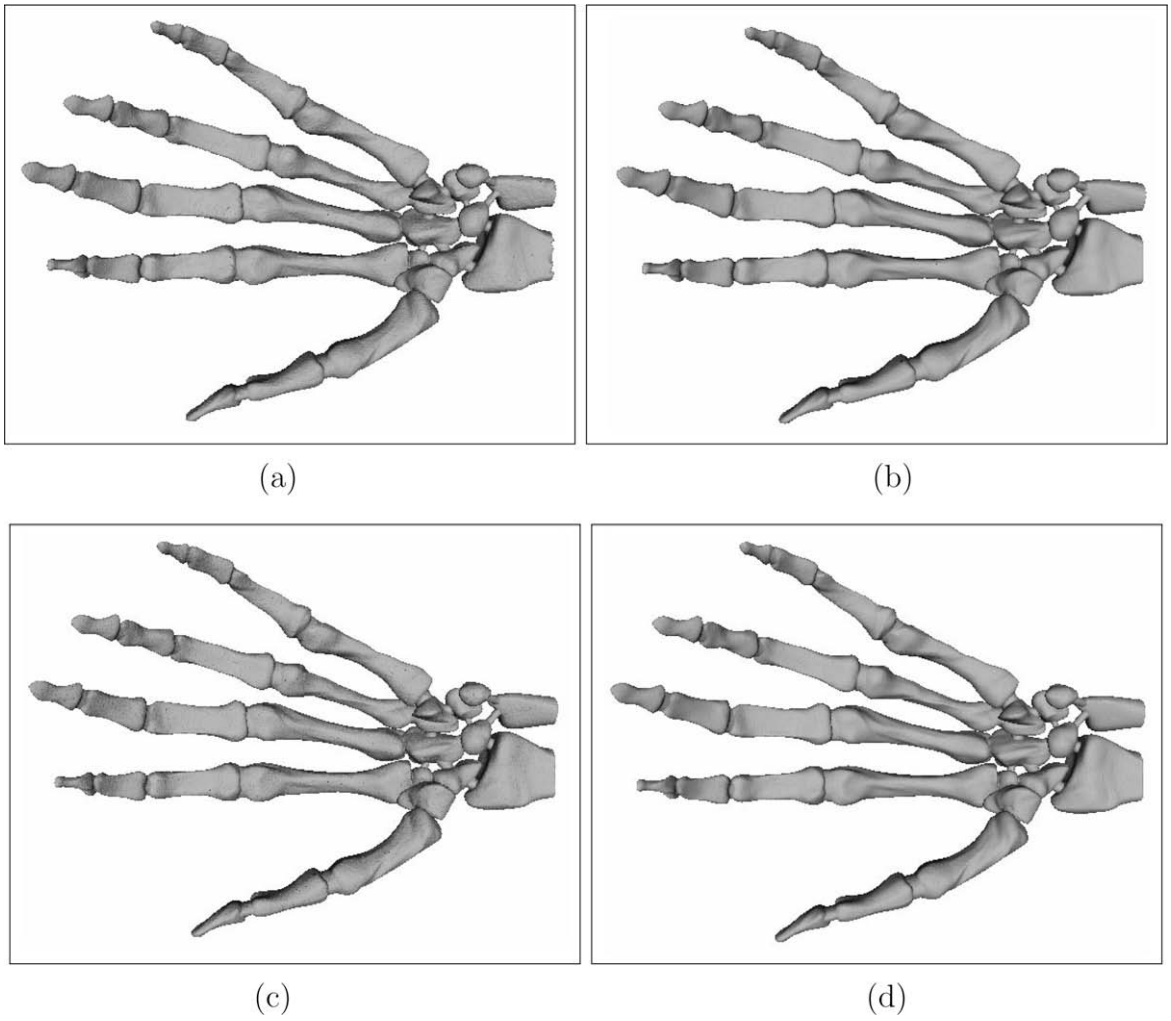

Fig. 7. The Hand model compressed using the proposed method with (a) 12.73 bpv, (b) 18.93 bpv, (c) 20.03 bpv. The original Hand model is given at (d). The distortions of the reconstructed models with respect to the Hausdorff distance metric are (a) 0.037921, (b) 0.03225, (c) 0.02737, respectively.

Geometry images (GI) [7] approach performs computationally-costly tasks, such as parameterization, and remeshing, to construct a 2D image from a 3D model. GI uses the signal-specialized parameterization algorithm [31,32]. This scheme first simplifies the input mesh progressively, creates a base mesh with $V_0 \ll V_N$, where V_0 to V_N are the number of vertices at each simplification level, and parameterizes this base mesh. During the parameterization process, the Jacobians of the elements of the base mesh are calculated and a linear system of size $V_0 \times V_0$ is solved, which is a $O({V_0}^3)$ operation. In the second step, the simplification process is reversed by split operation and the parameterization of the base mesh is refined according to the new vertices. During this refinement, the geometric-stretch error metric [32] should be calculated at each level and the parameterization should be updated accordingly.

Models with small number of vertices may be parameterized without the simplification process; however, the parameterization process becomes very complex for large models without simplification. Dividing the model into groups of faces, called *charts*, is another solution proposed for the parameterization of large models [32]. However, dividing the model into charts and parameterizing them is also a complex issue.

In our method, there is no need for dividing or simplifying the model. The model itself can be directly projected on a projection plane. The projection operation is simple and needs $2V$ multiplications for the calculation of the grid positions of the vertices. The complexity of the projection operation is $O(V_N)$. The rest of the compression algorithm is using wavelet transformation based compression algorithm SPIHT with adaptive prediction stages. The calculation of adaptive prediction stages does not have a complexity more than the ordinary prediction used in other wavelet based encoders. The complexity of the encoder part of the proposed algorithm is not much higher than ordinary wavelet based encoders.

Gu et al. [7] use a remeshing stage to map an arbitrary surface onto a completely regular structure. This remeshing causes a change in the connectivity of the model, and therefore, the original model cannot be restored. Besides, the remeshing is a computationally costly process. Our approach does not perform parameterization and remeshing. It solves only two linear equations for projection; one for determining the pixel positions of the mesh vertices on the image and another for finding the values of the projected pixels. Even large models, e.g., Hand and Happy Buddha (cf. Figs. 7 and 8), can easily be compressed using the proposed algorithm. On the other hand, to parameterize these models without any simplification and remeshing, a $543K \times 543K$ matrix for Buddha model and a $327K \times 327K$ matrix for Hand model should be solved.

The proposed mesh compression framework is lossless in the sense that it can compress an arbitrary 3D surface (e.g., irregular, semi-regular, regular) without changing the connectivity. Theoretically, the proposed mesh compression algorithm may obtain a lossless compression by using a very high-resolution projection grid, but practically it is lossy. However, the loss in mesh quality is insignificant. Reconstructed Hand, Happy Buddha and Horse models using different bitrates can be found in Figs. 7–9, respectively. A high resolution projection plane of 512×512, or

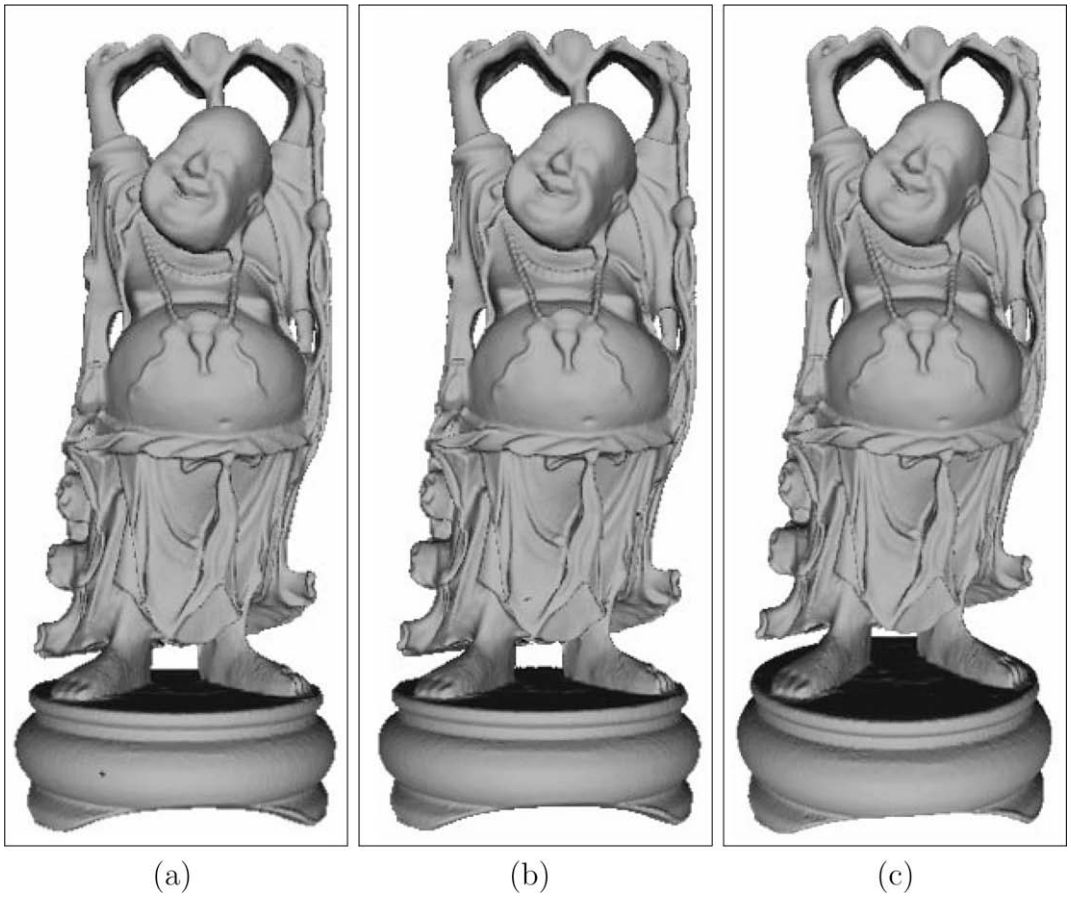

Fig. 8. The Happy Buddha model compressed using the proposed method with (a) 21.39 bpv, (b) 22.4 bpv. The original Hand model is given at (c). The distortions of the reconstructed models with respect to the Hausdorff distance metric are (a) 0.000328, (b) 0.000228, respectively.

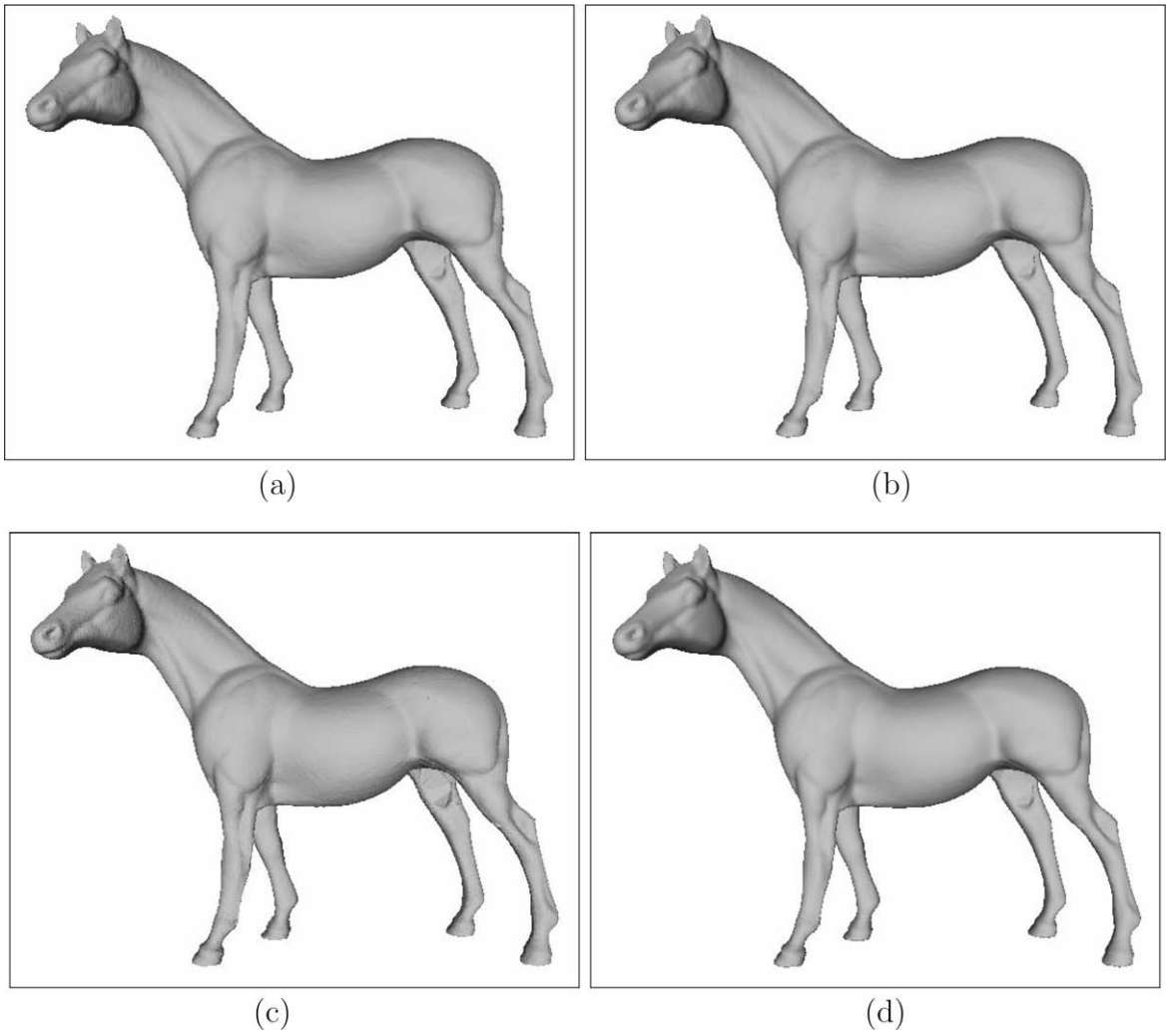

Fig. 9. The Horse model compressed using the proposed method with (a) 14.88 bpv, (b) 15.08 bpv, (c) 16.76 bpv. The original Horse model is given at (d). The distortions of the reconstructed models with respect to the Hausdorff distance metric are (a) 0.000453, (b) 0.000275, (c) 0.000131, respectively.

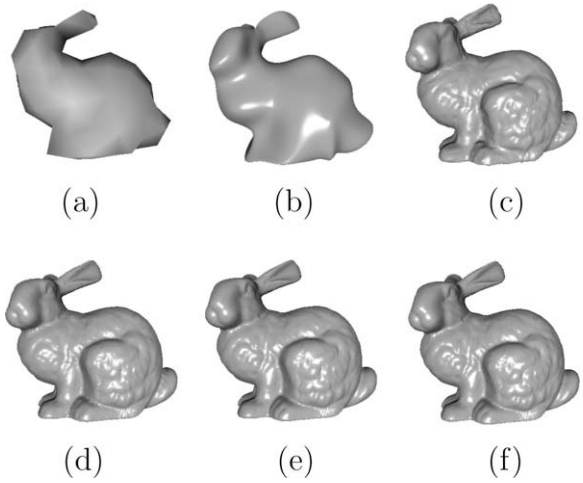

Fig. 10. (a) Base mesh of Bunny model constructed by the PGC algorithm (230 faces). The models reconstructed from (b) 5%, (c) 15%, (d) 50%, (e) 75% of the compressed stream. (f) The original Bunny model. The original model has a size of 6 MB and the compressed full stream has a size of 37.7 KB.

1024×1024, is used while compressing these large models. The results show that we can achieve a Hausdorff distance of <0.025 at bitrates around 16–21 bpv. As seen from Figs. 7–9, the difference between the original and reconstructed models cannot be perceived easily at these bitrates.

As opposed to the GI method, some vertices may get lost during the projection operation. Generally, the 3D meshes are not homeomorphic to a disk, and sometimes it may not be possible to find a correspondence between each vertex in a mesh and an image pixel. To handle the problem of discarded vertices, a neighborhood-based interpolation scheme that predicts the values of the lost vertices from its projected neighbors is defined.

It is observed that if many vertices are lost, the proposed approach may not work well. This problem can be solved by projecting the coinciding vertices to the nearest, empty sampling points of the projection plane. For complex models, another solution would be taking many projections. This would increase the bit rates but may decrease the distortion level of the reconstructed mesh. As a future work, we will investigate ways to find the best projection, by which the maximum number of mesh vertices are projected.

PGC (Progressive Geometry Compression) [25], which is another wavelet based mesh compression algorithm, is one of the most successful mesh coders in the literature. It can be seen from Fig. 10 that using only the 15% of a stream of 37.7 KB results in reconstructed model with very high quality. As the percentage of the used stream increases, the quality of the model increases too (Fig. 10). However, PGC works only on semi-regular or regular

meshes. Thus, before using wavelet transform on a mesh, it must be remeshed, as in [7]. Remeshing is also a computationally-intensive task; especially for large models. Our scheme can be applied to any mesh regardless of its regularity.

In Fig. 11, visual results produced by MPEG-3DGC/MPEG-4 and the proposed SPIHT based mesh coder are given. The proposed algorithm gives comparable results with the MPEG-3DGC/MPEG-4 (cf. Fig. 12). When the same *mean distance* between the original and reconstructed models are taken into account, the data size of the proposed coder is superior to MPEG-3DGC/MPEG-4.

Table 4 shows the bitrates of other lossless and lossy progressive mesh compression algorithms in the literature [26]. The major difference between the lossless and lossy compression schemes comes from remeshing. Lossy compression methods first remesh the 3D model and obtain a regular or semi-regular version of it; thus, the original model cannot be obtained [3]. On the other hand, lossless compression methods do not remesh the model; thus, the original model can be reconstructed. Table 4 shows that our method gives comparable results with the lossless compression methods in the literature. The bitrates of the lossy compression algorithms are better than both our method and the lossless compression algorithms in the literature. Their gain in bitrate can be attributed to the increased regularity in geometry [26]. Due to the obtained regular geometry (by remeshing), the prediction schemes used in lossy compression algorithms work more efficient.

8. Conclusion

In this paper, a mesh compression framework that uses Connectivity-Guided Adaptive Wavelet Transform based image coders is proposed. Two important components of the coder are the projection operation and the Connectivity-Guided Adaptive Wavelet Transform. We showed that a mesh can be compressed using known wavelet based image compression tools. We also showed that both single-rate and progressive encoding can be achieved using these tools.

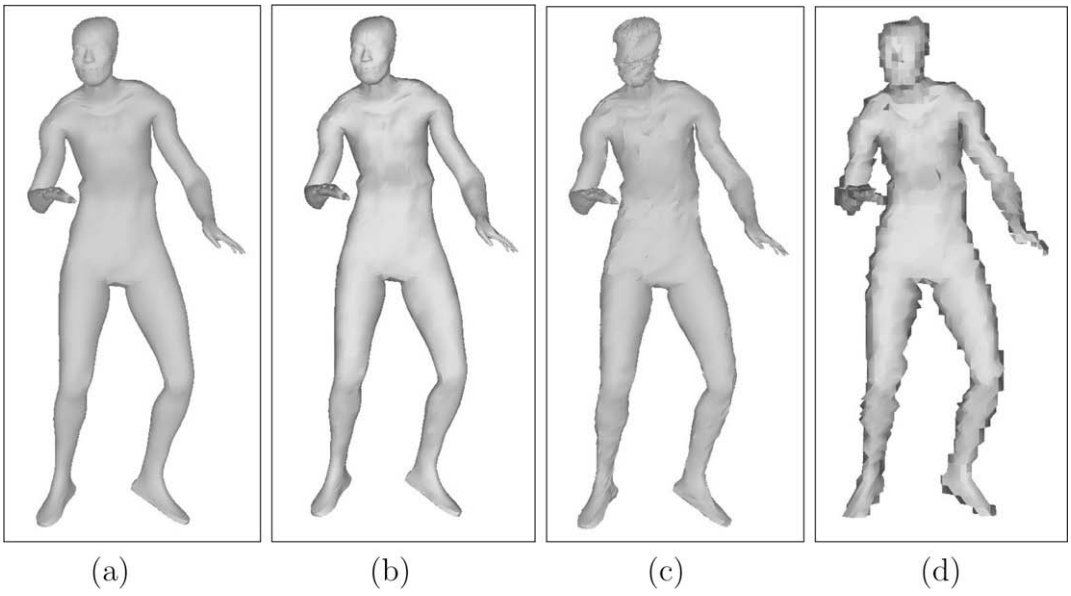

Fig. 11. The Dance model compressed using CGAWT at 21.5 bpv (a) and MPEG at 60.5 bpv (b) has the same distortion with respect to Hausdorff distance. (c) Decreasing the bit rate of CGAWT to 17 bpv causes a distortion in the face part of the model since the vertices are dense in this part. Decreasing the bit rate of MPEG to 58 bpv creates a distortion as seen in (d).

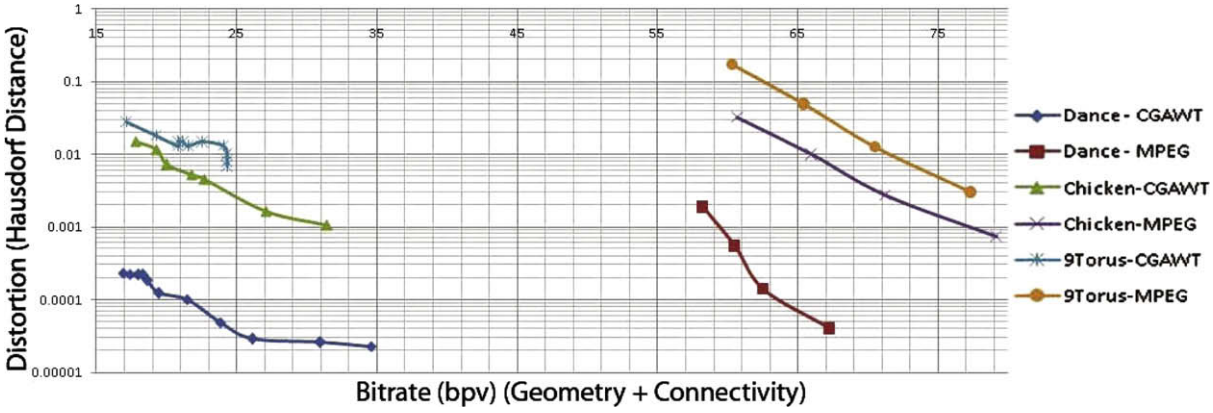

Fig. 12. The comparison between the MPEG and CGAW compression of Nine-Torus, Chicken, and Dance models. The bitrates include both the connectivity and geometry. Edgebreaker is used as the connectivity coder of CGAWT compression.

Table 4
Average bitrates of other lossless and lossy progressive mesh compression algorithms in the literature.

Algorithm name	Bitrate (bpv)
Lossless compression schemes	
Progressive simplicial complex [27]	Over 35 bpv
Progressive mesh [6]	About 35 bpv
Progressive forest split [28]	Slightly below 30 bpv
Compresses progressive mesh [29]	About 22 bpv
Valence driven compression [30]	14–20 bpv
Lossy compression schemes	
Progressive geometry compression [25]	About 5.5 bpv
Normal mesh compression [33]	3–4 bpv
Hierarchical 3DMC [4]	4–5.5 bpv

The results show that the compression of meshes using image processing techniques does not require any parameterization or remeshing on the mesh. Both remeshing and parameterization are computationally complex tasks. On the other hand, the projection operation defined here is simple to implement and needs less computation than remeshing and parameterization of the model.

The experimental results also show that the proposed Connectivity-Guided Adaptive Wavelet Transform increases the encoding efficiency compared to the ordinary wavelet transform. The neighborhood relation between the pixels of the projection image is described in the connectivity information of the model. Hence, it is necessary to take advantage of the connectivity information to predict a "pixel" from its connected pixels. Neighboring pixels in the image may not come from connected vertices of the mesh. Thus prediction structure proposed in Connectivity-Guided Adaptive Wavelet Transform performs better than the ordinary wavelet transform.

The Connectivity-Guided Adaptive Wavelet Transform is based on the prediction of grid points from its neighbors. It is observed that, SPIHT and JPEG2000 encoders provide good results when the values of signal samples are highly correlated with each other. Hence, a mesh-to-image transform providing high correlation between the grid values will lead to higher compression ratios.

Some of the best compression algorithms in the literature use remeshing as the preliminary step, since they are not applicable to irregular meshes. However, they cannot reconstruct the original mesh since they remesh the model. Another advantage of the proposed algorithm is that it is applicable to all types of meshes, including irregular, semi-regular or regular meshes.

Acknowledgments

This work is supported by European Commission Sixth Framework Program with Grant No. 511568 (3DTV NoE). The Cow and Lamp models are courtesy of Viewpoint Data Laboratories, Inc. The Skeleton Hand model is courtesy of the Stereolithography Archive at Clemson University. The Horse model is courtesy of Cyberware, Inc. Homer Simpson and Nine-Handle Torus models are obtained from INRIA Gamma Team 3D Meshes Research database. Stanford Bunny, Happy Buddha and Dragon models are courtesy of the Stanford Computer Graphics Laboratory.

References

[1] S.M. Rao, D.R. Wilton, A.W. Glisson, Electromagnetic scattering by surfaces of arbitrary shape, IEEE Transactions on Antennas and Propagation 30 (3) (1982) 409–418.
[2] J. Peng, C.-S. Kim, C.C.J. Kuo, Technologies for 3D triangular mesh compression: a survey, Journal of Visual Communication and Image Representation 16 (6) (2005) 688–733.
[3] P. Alliez, C. Gotsman, Recent advances in compression of 3D meshes, in: N.A. Dodgson, M.S. Floater, M.A. Sabin (Eds.), Advances in Multiresolution for Geometric Modelling, Springer-Verlag, 2005, pp. 3–26.
[4] F. Moran, N. Garcia, Comparison of wavelet-based three-dimensional model coding techniques, IEEE Transactions on Circuits and Systems for Video Technology 14 (7) (2004) 937–949.
[5] M. Lounsbery, T.D. DeRose, J. Warren, Multiresolution analysis for surfaces of arbitrary topological type, ACM Transactions on Graphics 16 (1) (1997) 34–73.
[6] H. Hoppe, Progressive meshes, in: Proceedings of ACM SIGGRAPH, 1996, pp. 99–108.
[7] X. Gu, S. Gortler, H. Hoppe, Geometry images, in: Proceedings of ACM SIGGRAPH, 2002, pp. 355–361.
[8] H.M. Briceo, P. Sander, L. McMillan, S. Gortler, H. Hoppe, Geometry videos: a new representation for 3D animations, in: Proceedings of ACM Symposium on Computer Animation, San Diego, CA, 2003, pp. 136–146.
[9] A. Said, W.A. Pearlman, A new fast and efficient image codec based on set partitioning in hierarchical trees, IEEE Transactions on Circuits and Systems for Video Technology 6 (3) (1996) 243–250.
[10] J.M. Shapiro, Embedded image coding using zerotrees of wavelet coefficients, IEEE Transactions on Signal Processing 41 (12) (1993) 3445–3462.
[11] JPEG 2000 Part I, Final Committee Draft, Version 1.0, 2000.
[12] J. Rossignac, Edgebreaker: connectivity compression for triangle meshes, IEEE Transactions on Visualization and Computer Graphics 5 (1) (1999) 47–61.
[13] K. Kose, A.E. Cetin, U. Gudukbay, L. Onural, Connectivity-guided adaptive lifting transform for image like compression of meshes, in: Proceedings of 3DTV Conference, Kos, Greece, May, 2007.
[14] VRML Standard, ISO/IEC 14772-1:1997.
[15] C. Guillemot, A.E.Çetin, R. Ansari, M-channel nonrectangular wavelet representation for 2-D signals: basis for quincunx sampled signals, in: Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), vol. 4, 1991, pp. 2813–2816.
[16] Ö.N. Gerek, A.E. Çetin, Adaptive polyphase subband decomposition structures for image compression, IEEE Transactions on Image Processing 9 (10) (2000) 1649–1660.
[17] N.S. Jayant, P. Noll, Digital Coding of Waveforms: Principles and Applications to Speech and Video, Prentice Hall, 1984.
[18] J.L. Gailly, M. Adler, gzip Compression Software. <http://www.gzip.org/>, 2009 (accessed at September 2009).
[19] DEFLATE Compressed Data Format Specification, Version 1.3. <http://tools.ietf.org/html/rfc1951>, 2009 (accessed at September 2009).
[20] P. Cignoni, C. Rocchini, R. Scopigno, Metro: measuring error on simplified surfaces, Computer Graphics Forum 17 (2) (1998) 167–174.
[21] N. Aspert, D. Santa-Cruz, T. Ebrahimi, MESH: measuring error between surfaces using the Hausdorff distance, in: Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), vol. 1, 2002, pp. 705–708.
[22] S. Mallat, A Wavelet Tour of Signal Processing, Academic Press, 1999.
[23] G. Rote, Computing the minimum Hausdorff distance between two point sets on a line under translation, Information Processing Letters 38 (3) (1991) 123–127.
[24] K. Kose, A.E. Cetin, U. Gudukbay, L. Onural, Nonrectangular wavelets for multiresolution mesh analysis and compression, in: Proceedings of SPIE Defense and Security Symposium, Independent Component Analysis, Wavelets, Unsupervised Smart Sensors, and Neural Networks IV, vol. 6247, 2006, pp. 19–30.
[25] A. Khodakovsky, I. Guskov, Progressive geometry compression, in: Proceedings of SIGGRAPH, 2000, pp. 271–278.
[26] A. Smolic, R. Sondershaus, N. Stefanoski, L. Váša, K. Müller, J. Ostermann, T. Wiegand, A survey on coding of static and dynamic 3D meshes, in: H.M. Ozaktas, L. Onural (Eds.), Three Dimensional Television-Capture, Transmission, Display, Springer, 2008, pp. 239–312.
[27] J. Popovic, H. Hoppe, Progressive simplicial complexes, in: Proceedings of ACM SIGGRAPH, 1997, pp. 217–224.
[28] G. Taubin, A. Gueziec, W. Horn, F. Lazarus, Progressive forest split compression, in: Proceedings of ACM SIGGRAPH, 1998, pp. 123–132.
[29] R. Pajarola, J. Rossignac, Compressed progressive meshes, IEEE Transactions on Visualization and Computer Graphics 6 (1) (2000) 79–93.
[30] P. Alliez, M. Desbrun, Valence-driven connectivity encoding of 3D meshes, Computer Graphics Forum 20 (2001) 480–489.
[31] P.V. Sander, S.J. Gortler, J. Snyder, H. Hoppe, Signal-specialized parameterization, in: Proceedings of the 13th Eurographics Workshop on Rendering Techniques, 2002, pp. 87–98.
[32] P.V. Sander, J. Snyder, S.J. Gortler, H. Hoppe, Texture mapping progressive meshes, in: Proceedings of ACM SIGGRAPH, 2001, pp. 409–416.
[33] A. Khodakovsky, I. Guskov, Normal mesh compression, in: Proceedings of ACM SIGGRAPH, 2000, pp. 95–102.