Abstract—This paper introduces a novel scalable 3D mesh compression technique based on a shape approximation prediction strategy. The proposed approach, so-called Shape Approximation Compression (SAC), directly compresses the levels of detail (LoDs) defined by the content creators, while exploiting their inter-correlations. Here, the geometry of each LoD is used in order to compute a smooth approximation of the next layer. A progressive mesh hierarchy is then built on the top of the approximated version making it possible to efficiently predict and progressively transmit the geometry approximation errors. The SAC codec was evaluated within the framework of the MPEG core experiments on Multi-Resolution 3D Mesh Coding (MR3DMC) and was preliminarily accepted for future standardization.

I. INTRODUCTION

Recent years have seen the widespread adoption of 3D rendering capabilities in a variety of devices, from personal computers and game consoles to mobile devices such as tablets and smartphones. This trend is fostering the development of applications exploiting ever richer 3D contents. Obviously the video game industry is leading the pack, but many applications can benefit from using 3D contents, e.g. virtual textbooks using 3D illustrations, car navigation systems showing nearby buildings in 3D, and e-shopping applications allowing the visualization of digitized products.

The various combination of devices and networks used to access these contents have different computing powers, display resolutions and bandwidths, thus efficient compression and adaptation of the 3D contents to the client/network is becoming a crucial challenge. A straightforward approach is to create multiple versions of the 3D contents at different levels of resolution; and quality and to send those versions simultaneously, the client deciding which version is best suited to its capabilities. Such a simulcast approach comes at the cost of lower compression efficiency since the inter-layer correlation is not exploited. A more efficient approach consists in designing a scalable bitstream, meaning that some parts of the stream can be removed, resulting in a valid substream with a lower reconstruction quality but still high considering the reduced data size.

In the context of 3D mesh coding, we consider two modes of scalability: spatial scalability, i.e. adapting the mesh resolution to the terminal rendering performances and to the available bandwidth and quality scalability, i.e. progressively refining the precision of the vertex coordinates and attributes as the bitstream is decoded. 3D mesh compression schemes supporting spatial and/or quality scalabilities have been widely proposed in the literature. The following section provides an overview of the state-of-the-art techniques in this field. In this paper we only consider multi-resolution lossless connectivity 3D mesh compression techniques (i.e. they do not perform re-meshing).

II. PREVIOUS WORKS

In his pioneering work [1], Hoppe introduced the Progressive Mesh (PM) technique, which represents a detailed mesh $M$ as a coarse mesh $M_0$ together with a set of vertex split refinement operations. The PM representation produces high quality Levels of Detail (LoDs), which makes it particularly suitable for view dependent rendering applications. However, it requires $10-20$ times higher bitrates than [2] to encode the connectivity information.

Since its introduction in [1], the PM principle has been constantly exploited to design more compact representations by introducing either a different refinement operation (e.g. forest split [3] or vertex insertion [4]) or a more constrained decimation strategy (e.g. octree [5] and connectivity-based approaches [6], [7]).

In practice the codecs [7], [6], [4], [8] and [3] achieve a bitrate of $2-7$ bits per vertex (bpv) when compressing the mesh connectivity. However, in the case of non-uniformly sampled meshes, they produce poor quality LoDs because of their underlying topological and geometric constraints. Furthermore, content creators are most often reluctant to use these techniques because they do not offer any control over the generated LoDs. Instead, a discrete set of $N$ LoDs, denoted $(L_i)_{i \in \{0, ..., N-1\}}$, each manually optimized for a target platform or device is produced. The LoDs are then compressed by using a static mesh encoder and separately transmitted. Such a simulcast transmission strategy ensures optimal quality LoDs for each platform. However, it requires high bitrates and large storage capabilities.

The Shape Approximation Compression (SAC) approach proposed in this paper overcomes the limitations mentioned above by directly compressing the LoDs defined by the content creators, while exploiting their inter-correlations. Here, the LoD $L_i$ is predicted from the LoD $L_{i-1}$ by computing a smooth approximation $L_i^*$, which is further decimated by applying the Quadric Error Metric (QEM) simplification technique [9] in order to generate a PM hierarchy denoted $PM(L_i^*)$. By computing the PM hierarchy on $L_i^*$ instead of $L_i$, the SAC encoder guarantees that the decoder is able
to build $PM(L_1^*)$ without having to explicitly include any additional information in the bitstream. $PM(L_1^*)$ is finally exploited in order to further predict and progressively transmit the approximation errors.

The SAC technique offers high compression performances (cf. Section IV), while supporting both quality and spatial scalabilities. SAC was recently considered for future standardization within the framework of the MPEG activities on Multi-Resolution 3D Mesh Coding [10].

The remainder of this paper is structured as follows. Section III describes the proposed technique. The compression performances of the SAC codec are objectively evaluated in Section IV.

III. SHAPE APPROXIMATION COMPRESSION

Figures 1 and 2 present the synopsis of both the SAC encoder and decoder, respectively.

The SAC encoder compresses the lowest LoD $L_0$ by using the standardized static mesh encoder TFAN [11]. TFAN offers a lower computational complexity and competitive compression performances w.r.t. the state-of-the-art mono-resolution codecs, while handling general topologies (e.g. non-manifold meshes).

Only the connectivities of the LoDs $(L_i)_{i \in \{1,...,N-1\}}$ are compressed using TFAN. The geometry information, which represents more than 80% of the bitstream, is compressed by exploiting a shape approximation-based prediction strategy. More precisely, the SAC encoder exploits the connectivity information of the current layer $L_i$ together with the geometry information of the previous layer $L_{i-1}$ in order to compute a smooth approximation $L_i^*$ of $L_i$. Here, a mapping information $Map((i-1) \rightarrow i)$ describing a correspondence between the vertices of $L_{i-1}$ and a subset $C_i^*$ of the vertices of $L_i$ is explicitly encoded in the bitstream. $Map((i-1) \rightarrow i)$ makes it possible to associate the 3D positions and attributes of the vertices of $L_{i-1}$ to the vertices $C_i^*$, which are exploited as control points to compute $L_i^*$ (cf. Section III-A).

As in [12], the encoder may optionally choose a set $C_i^a$ of additional control points (i.e. $C_i^a \cap C_i^* = \emptyset$), and send explicitly their positions/attributes and indices. In the case of meshes with salient features, an auxiliary information $S_i$ specifying the salient edges of $L_i$ is also included in the bitstream. The positions of the additional control points are quantized and arithmetically encoded together with their indices, the mapping and the edge saliency information.

$L_i^*$ is computed by exploiting: $C_i$, $G_{i-1}$, $S_i$, $C_i^*$, $C_i^a$ and $Map((i-1) \rightarrow i)$. The progressive mesh hierarchy $PM(L_i^*)$ is then constructed by decimating $L_i^*$ as described in [9], allowing only half-edge collapse operations. $PM(L_i^*)$ is exploited in order to predict the approximation errors of $L_i^*$ w.r.t. $L_i$ (cf. Section III-B). Finally, the predicted approximation errors $(e_{vi})_{v \in \{1,...,V\}}$ ($V$ being the number of vertices) are arithmetically encoded and progressively transmitted to the decoder.

The SAC decoder decodes the LoD $L_0$ by exploiting the TFAN decoder. The remaining LoDs are decoded as follows.

First, the connectivity $C_i$, the positions/indices of the additional control points, the mapping and the edge saliency information and are decoded and exploited to compute $L_i^*$. Then, the decoder builds exactly the same progressive mesh structure $PM(L_i^*)$ computed by the encoder. Note here that the PM hierarchy is obtained at no extra cost since it is entirely derived from $L_i^*$. Finally, the predicted approximation errors $(e_{vi})_{v \in \{1,...,V\}}$ are progressively decoded and used to reconstruct the different quality LoDs as described in Section III-B.

A. Laplacian-based Mesh Approximation

The SAC codec extends the Laplacian-based approximation technique described in [12] to meshes with salient features. In this work, we have considered as salient all the edges exhibiting a dihedral angle higher than $\frac{\pi}{2}$ radians. Figure 3.a illustrates the salient edges detected for the “Fandisk” model.
We define the Laplacian matrix $L_i$ as follows:

\[
L_i(k,j) = \begin{cases} 
1 & \text{if } j = k \\
-\frac{\alpha}{|k|} & \text{if } k \leq V_i, j \in k^* \\
-\frac{\beta}{|k|} & \text{if } k \leq V_i, j \in k^*_s \\
1 & \text{if } k > V_i, j \in C_i \\
0 & \text{otherwise}
\end{cases}
\]

(1)

where,
- $C_i = (c^0_i, ..., c^{C_i-1}_i)$ is the set of control vertices and $\Gamma_i$ its cardinality,
- $(k,j) \in \{1, ..., V_i + \Gamma_i\} \times \{1, ..., V_i\}$,
- $k^*_s$ is the set of topological neighbours of the vertex $k$ sharing with it either a boundary edge (i.e. adjacent to exactly one triangle) or a salient edge (i.e. belonging to $S_i$),
- $k^*_s$ is the set of neighbours of the vertex $k$ not belonging to $k^*_s$,
- $|k^*_s|$ and $|k^*_s|$ are the numbers of elements of $k^*_s$ and $k^*_s$ respectively,
- $\alpha$ and $\beta$ are the weights associated with the special edges (i.e. salient or boundary edges) and to the non-special ones, respectively.

As in [12], the $V_i \times 3$ matrix of approximated vertex positions, denoted $P^* = (P^*(d))_{d \in \{1,2,3\}}$, is computed by solving the following sparse linear system [13]:

\[
(L_i^T L_i) \times P = L_i^T B.
\]

(2)

The k-th line $B_i(k)$ of the $(V_i + C_i) \times 3$ matrix $B_i$ is given by:

\[
B_i(k) = \begin{cases} 
P^T_{k^*V_i} & \text{if } (k > V_i) \\
0 & \text{otherwise}
\end{cases}
\]

(3)

where $P^T_{k^*V_i}$ represents the 3D position of the control vertex $c^{k^*V_i}$.

Note that if the weights $\alpha$ and $\beta$ are equal or if there are no special edges, we obtain exactly the definition of the Laplacian matrix proposed in [12]. In this work, $\alpha$ was set to one hundred and $\beta$ to 1. This implies that a vertex $k$ located on a mesh boundary or on a salient edge is a 100 times more influenced by its special neighbours $k^*_s$ than by the non-special ones (i.e. $k^*_s$). As illustrated in Figure 3, this modified version of $L_i$ better preserves the salient features of the mesh.

### B. Approximation Errors Prediction

The SAC encoder compresses the predicted approximation errors in the reverse order of $PM(L^*_i)$. At each step, the predicted approximation error $e_v$ associated with the vertex $v$ is computed as follows:

\[
e_v = (P_v - \frac{1}{|v^*|} \sum_{w \in v^*} \hat{P}_w) - (P_v^* - \frac{1}{|v^*|} \sum_{w \in v^*} P_w^*)
\]

(4)

where $v^*$ represents the set of topological neighbours of $v$ in the current LoD of $PM(L^*_i)$ and $(\hat{P}_w)_{w \in v^*}$ are the positions reconstructed by the decoder as described below (cf. equation (7)).

The obtained error $e_v$ is then decomposed into a normal component $e_v^n$ and two tangential ones $e_v^t$ and $e_v^r$, defined by:

\[
e_v^n = e_v \cdot n_v, e_v^t = e_v \cdot t_v, e_v^r = e_v \cdot r_v
\]

(5)

where $n_v$ is the normal of $L^*_i$ at vertex $v$ and $t_v^*$ and $r_v$ are two vectors chosen to form a direct orthonormal basis with $n_v$.

Finally, $e_v^n$, $e_v^t$, and $e_v^r$ are quantized and arithmetically encoded. As noted in [6], for smooth meshes, the normal component $e_v^n$ contains more information than the tangential ones (i.e. $e_v^t$ and $e_v^r$). Therefore, we quantize more finely $e_v^n$ than $e_v^t$ and $e_v^r$.

The SAC decoder progressively decompresses the approximation prediction errors starting from the lowest LoD to the highest one. Here, at each step, the three components $(e_v^n, e_v^t, e_v^r)$ are arithmetically decoded, de-quantized and then used to reconstruct the approximation error $e_v$, as follows:

\[
\hat{e}_v = e_v^n \hat{n}_v + e_v^t \hat{t}_v^* + e_v^r \hat{r}_v^*
\]

(6)

Finally, the decoded positions $(\hat{P}_v)_v$ are given by:

\[
\hat{P}_v = \hat{e}_v + \frac{1}{|v^*|} \sum_{w \in v^*} \hat{P}_w + (P_v^* - \frac{1}{|v^*|} \sum_{w \in v^*} P_w^*)
\]

(7)

Note that by encoding/decoding the vertices in the reverse order of $PM(L^*_i)$, the SAC encoder/decoder guarantee that when processing the vertex $v$, the positions $(\hat{P}_w)_{w \in v^*}$ of all its neighbours have already been reconstructed.

### C. Spatial and quality scalabilities

The SAC supports the spatial scalability by directly exploiting the LoDs $(L^*_i)_i$ defined by the content creator. The quality scalability is obtained by progressively transmitting the predicted approximation errors as illustrated in Figure 4. By setting to zero the non-decoded predicted approximation errors $\hat{e}_v$ in equation (7), the decoder is able to compute a smooth version of each LoD $L_i$ at each stage of the transmission process. Note that the QEM simplification strategy used to build the progressive mesh structure $PM(L^*_i)$ ensures that the errors are sent with a shape-based priority (i.e. starting from the most relevant features to the least important ones), which ensures high Rate-Distortion (RD) performances.
D. Computational complexity

The SAC encoding/decoding complexity is determined by the mesh decimation procedure which is in $O(V \log(V))$. As a reference, on a 2.4 GHz Core2 CPU with 3 GB of RAM, the SAC codec processes 10K vertices per second on average.

IV. EXPERIMENTAL RESULTS

A. Evaluation dataset and criteria

The SAC codec was evaluated in the context of the MPEG’s 3D Geometry (3DG) group activities on Multi-Resolution 3D Mesh Coding (MR3DMC) [10]. The MR3DMC evaluation dataset is composed of 85 dense 3D models stored as triangular meshes in the VRML 2.0 format. The considered models cover a large set of applications (e.g. GIS, CAD, cultural heritage and medical imaging) and show a high variability in terms of complexity and topology: their number of vertices ranges from 20K to 564K (average: 133K); their number of connected components varies from 1 to 168 (average: 5). 52% of the models are manifold, 68% are closed and 73% are orientable. The compression distortions were measured by using the $L^2$ error evaluated by the MESH tool$^1$. The bitrates are reported in bpv.

B. Comparative evaluation

Figure 5 compares the RD curves of the proposed SAC technique to those of the single rate approach [2] and to the multi-resolution codecs Wavemesh [7], AD [6], OCT [14] and BAQ [15]. Here, four LoDs were generated by applying the simplification approach [9] to each 3D model: (1) $L_0$ having 1.5% of the vertices of the original mesh, (2) $L_1$ having 10%, (3) $L_2$ having 20% and $L_3$ having 100%.

The compression performances reported in Figure 5 clearly show that the SAC technique offers the best RD performances for all the bitrates. Furthermore, the proposed technique is highly flexible since it allows the content creator to specify and manually optimize the spatial LoDs. The SAC high compression performances are mainly obtained thanks to its approximation-based prediction strategy which outperforms the simple predictors considered by the other techniques.

Because of its high RD performances, its flexibility and its support for both quality and spatial scalabilities, the SAC codec was selected for future MPEG standardization.

REFERENCES


$^1$http://mesh.berlios.de