Zerotree Coding of Subdivision Wavelet Coefficients in Dynamic Time-Varying Meshes

Maja Krivokuća [1], Tomás Malheiros Borges [2], and Ricardo de Queiroz [2]

[1]InterDigital R&D France
[2]Affiliation not available

December 7, 2023

Abstract

We propose a complete system to enable progressive coding with quality scalability of the mesh geometry, in MPEGâ\euros state-of-the-art Video-based Dynamic Mesh Coding (V-DMC) framework. In particular, we propose an alternative method for encoding the subdivision wavelet coefficients in V-DMC, using a mesh-based zerotree coding approach. The proposed method works directly in the native 3D mesh space. It allows us to identify parent-child relationships amongst the wavelet coefficients across different subdivision levels, which can be used to achieve an efficient and versatile coding mechanism. We demonstrate that, given a starting base mesh, a target subdivision surface and a desired maximum number of zerotree passes, our system produces an elegant and visually attractive lossy-to-lossless mesh geometry reconstruction with no further user intervention. Moreover, lossless coefficient encoding with our approach is shown to require almost the same bitrate as the default displacement coding methods in V-DMC. Yet, our approach provides several levels of quality resolution within each target bitrate, while the current solutions encode a single quality level only. To the best of our knowledge, this is the first time that a zerotree-based method has been proposed and demonstrated to work for the compression of dynamic time-varying meshes, and the first time that an embedded quality-scalable approach has been used in the V-DMC framework.

Zerotree Coding of Subdivision Wavelet Coefficients in Dynamic Time-Varying Meshes

Maja Krivokuća, Tomás M. Borges and Ricardo L. de Queiroz

***Abstract*—We propose a complete system to enable *progressive coding* with *quality scalability* of the mesh geometry, in MPEG's state-of-the-art *Video-based Dynamic Mesh Coding* (V-DMC) framework. In particular, we propose an alternative method for encoding the subdivision wavelet coefficients in V-DMC, using a mesh-based *zerotree coding* approach. The proposed method works directly in the native 3D mesh space. It allows us to identify parent-child relationships amongst the wavelet coefficients across different subdivision levels, which can be used to achieve an efficient and versatile coding mechanism. We demonstrate that, given a starting base mesh, a target subdivision surface and a desired maximum number of zerotree passes, our system produces an elegant and visually attractive lossy-to-lossless mesh geometry reconstruction with no further user intervention. Moreover, lossless coefficient encoding with our approach is shown to require almost the same bitrate as the default displacement coding methods in V-DMC. Yet, our approach provides several levels of quality resolution within each target bitrate, while the current solutions encode a single quality level only. To the best of our knowledge, this is the first time that a zerotree-based method has been proposed and demonstrated to work for the compression of dynamic time-varying meshes, and the first time that an embedded quality-scalable approach has been used in the V-DMC framework.**

***Index Terms*—Dynamic time-varying meshes, V-DMC, subdivision surfaces, subdivision wavelets, mesh compression, zerotree.**

I. Introduction

THE compression of three-dimensional (3D) mesh models has been a topic of interest in the research community for a number of decades [1], but the recent appearance of *volumetric video* content has given rise to a new challenge: the efficient encoding of *dynamic time-varying meshes* (TVMs) [2]. Unlike dynamic *animated meshes*, for which a number of compression algorithms have already been developed in the past [1], [2], the biggest challenge with TVMs is that they can have a variable number of vertices, different connectivity and topology, and different attribute data, for each frame of a volumetric video. Such meshes are more representative of a real-world capturing system, where the 3D mesh in each frame is usually constructed (semi-)independently. Hence, volumetric videos tend to have a heavy raw representation and require efficient compression techniques in order to be usable in practice. For this reason, the "Coding of 3D Graphics and Haptics" subgroup of the Moving Picture Experts Group (i.e., MPEG-3DGH) recently began to address the problem of compressing TVMs, in the context of the Video-based Dynamic Mesh Coding (V-DMC) activity. Following a Call for Proposals (CfP) in 2021 [3], the solution proposing Video- and Subdivision-based Mesh Coding (VSMC) [4] was selected to become the basis of the V-DMC Test Model (TM) that is currently being used for the development of this standard [5]. This method is now considered the state-of-the-art in TVM compression. While the V-DMC system considers all aspects of a dynamic mesh sequence for compression, in the current paper we will focus on the coding of the mesh *geometry* (i.e., the (x, y, z) positions of the mesh vertices).

VSMC [4] is based on a variation of the well-known *subdivision wavelets* scheme [6], [7], which is implemented using a Lifting Wavelet Transform [8]. In VSMC and in the current V-DMC TM, the default method for encoding the resulting wavelet coefficients and the *base mesh displacements* (explained in Section II) is to quantize and pack them into 2D images, then compress these images using a standard 2D video encoder such as HEVC [9]. However, this way of encoding the wavelet coefficients does not take into account their natural hierarchical structure, which could be exploited to achieve a very efficient coding mechanism. Furthermore, the organisation of wavelet coefficients into 2D images does not easily lend itself to a fully progressive reconstruction at the decoder, where both the *resolution scalability* across different subdivision levels and the *quality scalability* within a resolution level are possible[1]. These functionalities are expected to be very important for applications requiring TVM coding, as we cannot expect different users to have the same bitrate and quality needs. For these reasons, in this paper we propose the following new contributions to improve the functionality and versatility of the V-DMC framework:

- Instead of packing the wavelet coefficients into 2D images, we propose an organisation of these coefficients into an edge-based tree hierarchy in 3D space, starting from the base mesh and extending to the highest available subdivision level.
- A method for zerotree encoding of the wavelet coefficients based on the above-mentioned hierarchy.
- An embedded bitstream that includes the base mesh

M. Krivokuća is with InterDigital, Rennes, France, e-mail: maja.krivokuca@interdigital.com.

T. M. Borges is with the Electrical Engineering Department at Universidade de Brasília, Brasília, Brazil, e-mail: tomas@divp.org.

R. L. de Queiroz is with the Computer Science Department at Universidade de Brasília, Brasília, Brazil, e-mail: queiroz@ieee.org.

This work was partially supported by the Brazilian National Research Council (CNPq) under Grant 88887.600000/2021-00 and Grant 301647/2018.

[1] *Resolution scalability* here refers to the number of vertices and faces in a mesh; *quality scalability* means the precision or accuracy of the reconstructed vertex (*x, y, z*) positions *within* a resolution level.

displacements and zerotree-coded wavelet coefficients, which allows for an elegant lossy-to-lossless progressive encoding and reconstruction of the mesh geometry at different quality resolution levels.

II. Subdivision Wavelet Coefficients in V-DMC

A key component of the V-DMC framework is *surface subdivision*, a process that is well known in the field of computer graphics [10]–[14]. In V-DMC, the mesh in each frame of the input sequence first goes through a pre-processing stage, where it is iteratively decimated [15] to obtain a lower-resolution version of the original mesh, called a *base mesh*. The base mesh is then re-parameterized (to adapt the input texture map coordinates to the new, simplified geometry), and finally subdivided again by using a simple *mid-point subdivision* scheme. In such a subdivision scheme, new vertices and faces are added to the simplified mesh by inserting a new vertex at the midpoint of each existing edge and uniformly splitting each triangular face into four sub-triangles. This subdivision is applied in an iterative manner, in order to upsample the base mesh to reach approximately the same spatial resolution (number of vertices and faces) as the original input mesh[2]. The intermediate resolution (subdivision) levels provide different *levels of detail* (LoDs) for the mesh. The *displacement* vector field is then determined by finding, for every vertex of the final subdivided mesh, the distance to the closest point on the original input mesh surface, as illustrated in Fig. 1. The final base mesh is generated by adjusting the 3D positions of the re-parameterized mesh to minimize the distance between the subdivided mesh and its displaced counterpart, in search of piecewise smooth surface reconstruction [4]. This process of decimation, subdivision, and finally vertex position adjustment is needed because the original input meshes usually do not have *semi-regular* connectivity [2], which is needed for predictable surface subdivision.

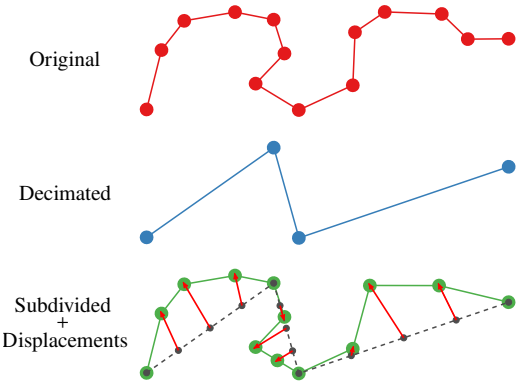


Fig. 1: Illustration of the computation of displacements in V-DMC. This makes it possible to reconstruct the original mesh geometry in a lossy way, encoding only a small decimated base mesh and a set of displacement vectors, since the subdivision is regular and can be performed at the decoder without any extra connectivity data being sent.

[2]Although in the current V-DMC TM, the number of vertices and faces in the upsampled mesh is usually much larger than in the original mesh.

For high-bitrate targets in V-DMC, the base mesh resolution is already close to the original mesh resolution, so there is very little (if any) decimation performed and the displacement vectors are not encoded at all. In such cases, only the geometry of the base mesh is encoded, using some state-of-the-art static mesh encoder, such as the MPEG implementation of EdgeBreaker [16]. We are more interested in the case where the displacement vectors are actually encoded. In this case, the base mesh in V-DMC is still encoded using the MPEG EdgeBreaker, and the displacements are transformed by using the Lifiting Wavelet Transform [8], exploiting the subdivision structure. Because the connectivity of the base mesh can be refined in a predictable manner by using a set of standard subdivision rules known to both the encoder and decoder, the only connectivity data that needs to be transmitted is that of the base mesh, which is usually relatively small. The geometry of the base mesh must also be encoded and sent to the decoder, as well as a set of encoded wavelet coefficients. In V-DMC, the wavelet coefficients, along with the base mesh displacements (which are used only for the surface fitting and are not transformed), are quantized and packed into 2D images for encoding, e.g., by using a standard video encoder such as HEVC. There is also an alternative option to encode the displacements by using arithmetic coding (AC). The reconstructed geometry is formed by the decoded base mesh (losslessly encoded), which is subdivided and added to the decoded displacement field.

The simplest form of *subdivision wavelets* [6], [7], with linear time complexity for analysis and synthesis, can be implemented by a piecewise linear subdivision that is basically equivalent to the Lifting Wavelet Transform [8]. In this simple form, the lowest-resolution approximation of the mesh geometry is represented by the base mesh (LoD_0), and the wavelet coefficients between any two successive resolution (subdivision) levels represent the differences between the "child" vertex (x, y, z) positions at LoD_N and the prediction of these child positions using the positions of their "parents" at LoD_{N-1}. For example, in Fig. 2, the wavelet coefficient w associated with the displaced vertex v (where v is a higher-resolution level vertex that was originally predicted at the midpoint $\hat{v}$ of a lower-resolution level edge pq) can be computed as:

$$w = v - 0.5(p + q) = v - \hat{v}. \tag{1}$$

We see in (1) that the resulting wavelet coefficient w is thus the difference between the final (displaced) position of v and its prediction $\hat{v}$. The base mesh can therefore be progressively refined by successive subdivision and addition of more and more wavelet coefficients at higher resolution levels.

Fig. 3 shows the Lifting Wavelet Transform scheme used in V-DMC. The displacements (*disp*) are split into "high" (LoD_N) and "low" (LoD_{N-1}) samples. The high-resolution samples H_N are transformed into wavelet coefficients W_N by subtracting the prediction P_N, as in (1)[3], and then quantized.

[3]Except that, in V-DMC, the Lifting Wavelet Transform is applied on the *displacement vectors* and not on the vertex (x, y, z) positions directly, so the wavelet coefficients represent the prediction errors for the displacements and not for the positions directly.

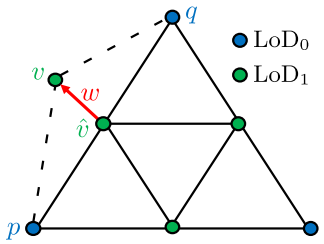


Fig. 2: Illustration of wavelet coefficient computation in linear polyhedral subdivision. The blue vertices are in the base mesh (LoD_0), and the green vertices (LoD_1) are the new vertices inserted at the edge midpoints of the base mesh.

The low-resolution samples L_N can optionally be "updated" by adding to them (a fraction of) the sum of the wavelet coefficients of neighbouring vertices. The process in Fig. 3 is repeated until the base mesh (LoD_0) is reached [4]. As mentioned earlier, the decoded displacements must be added back iteratively to the decoded base mesh vertices, to obtain the final vertex positions of the higher-resolution meshes.

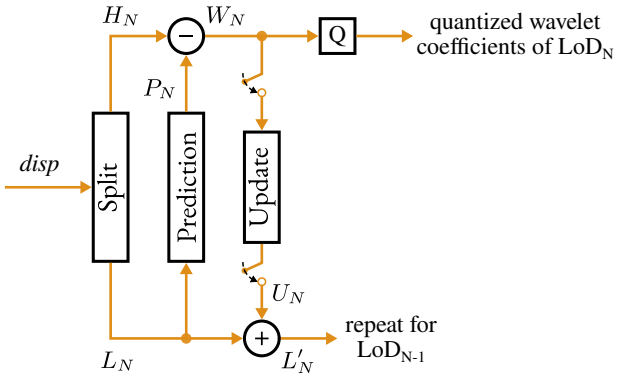


Fig. 3: Lifting scheme used in V-DMC for computing the wavelet coefficients for the input displacement (*disp*) data.

III. Zerotree Coding

The zerotree algorithm is a lossy-to-lossless compression algorithm for a Discrete Wavelet Transform or other hierarchical subband decomposition [17]. It provides a compact multi-resolution representation of the "significance maps" of a transform coefficient hierarchy. This coding mechanism was first proposed by Shapiro for wavelets on images [17] and later extended to subdivision wavelets on static 3D meshes [18]. In [18], Loop [12] subdivision wavelet coefficients on semi-regular meshes are first organised into an edge-based tree hierarchy and then encoded using an image-based SPIHT algorithm [19]. The base mesh scaling coefficients are uniformly quantized. In [20], the same SPIHT-based coding system is used as in [18] to compress unlifted *butterfly* [13] wavelet coefficients of *normal* meshes. In [21], *butterfly* subdivision wavelet coefficients of segments of *normal* meshes are organised into modified versions of the tree structures from [18], in order to be encoded using SPIHT. In [22], the "details" of *butterfly* subdivision are similarly encoded using SPIHT. In [23], the authors propose to scale the subdivision wavelet coefficients by applying different weights at different resolution levels, before applying the SPIHT coder.

Zerotree coding begins with a choice of an initial threshold T_0 to determine a "significant" wavelet coefficient magnitude. Wavelet coefficients are then traversed following an order known to both the encoder and decoder, and are labelled using a binary decision tree, shown in Fig. 4, with the codes:

- **ZTR** (Zerotree Root): A wavelet coefficient that is insignificant (i.e., its magnitude is below the chosen threshold) and all of its descendant coefficients are also insignificant. Specifying a ZTR allows the decoder to zero out all the related descendant coefficients, which do not need to be reconstructed as they are deemed insignificant.
- **POS** (Positive Significant Coefficient): A wavelet coefficient whose magnitude is significant and its sign is positive.
- **NEG** (Negative Significant Coefficient): A wavelet coefficient whose magnitude is significant and its sign is negative.
- **IZ** (Isolated Zero): A wavelet coefficient that is insignificant, but has one or more significant descendants.

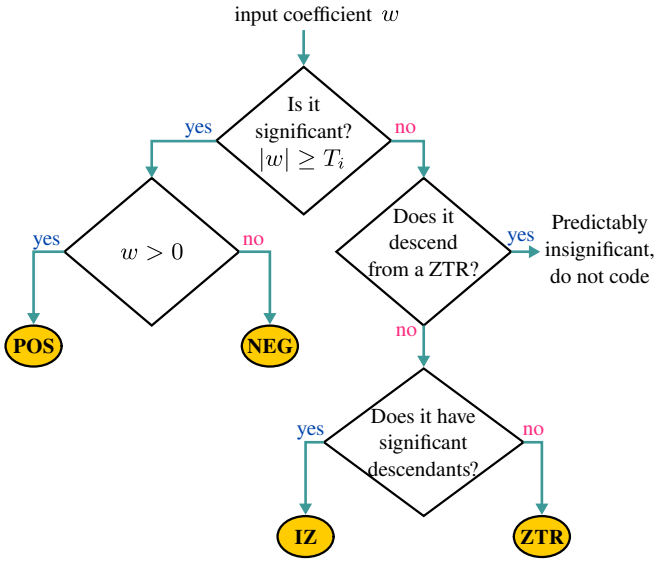


Fig. 4: Binary decision tree for encoding the significance of a wavelet coefficient using zerotrees.

The coded binary tree using the aforementioned symbols is constructed at the encoder through a series of *dominant* (or *significance*) passes and *subordinate* (or *refinement*) passes, and, at each iteration, the significance threshold T_i is updated (usually reduced by a factor of 2). At each dominant pass, the significance of the coefficients that have not yet been found significant in earlier iterations is coded by scanning the hierarchical coefficient tree and emitting one of the four possible symbols. The children of a coefficient are only coded if the parent coefficient is found to be significant, or if the coefficient is an isolated zero. The subordinate pass then emits one bit (representing the next most significant bit that has not yet been emitted) for each coefficient that has been deemed significant so far. The latter are known as *refinement bits*, as the subordinate pass is a form of bitplane

coding. The maximum possible number of refinement passes depends on the quantization level of the coefficients. The final bitstream includes data arranged in order of importance: the most significant bit (MSB) of the most significant wavelet coefficient first, the least significant bit (LSB) of the least significant coefficient last. This ensures that the most important (highest-order) bits of the most important (largest-magnitude) wavelet coefficients are transmitted first, as these usually make the most significant contributions towards reducing reconstruction error. The encoder can terminate the encoding at any point, for example when some target bitrate and/or distortion measure has been met, or some desired number of passes has been completed. The decoder can choose to terminate the transmission when satisfied with the decoded result. It can reconstruct the signal associated with any prefix of the bitstream by running an inverse wavelet transform on the significance and refinement bits received so far. Generally, the greater the number of bits that are encoded and decoded, the better the signal reconstruction.

IV. Proposed Approach

The hierarchy that we use to construct zerotrees on meshes is inspired by the work in [18]. It was observed in [18] that if one associates the wavelet coefficients with a quadrilateral face (as in images), the hierarchical coefficient trees naturally follow from the face quadtree. A similar concept can also be used for *dual* subdivision schemes (i.e., where the wavelet coefficients are associated with the mesh faces). When the subdivision scheme and the associated wavelet transform are *primal* (i.e., the coefficients live on the mesh vertices, not on the faces), as is the case in V-DMC, the tree needs to be constructed differently. In the latter case, the *vertices* of a coarser-level mesh are associated with the *scaling coefficients* of a wavelet transform, while the *wavelet coefficients* are associated with the mesh *edges* – that is, the subdivision wavelet coefficients at a given resolution level have a one-to-one association with the edges of the mesh at the previous resolution level, since the wavelet coefficients are computed on vertices that are inserted at the edge midpoints of the coarser-resolution mesh (as illustrated in Fig. 2). Assuming a triangular mesh and a recursive subdivision of each face into four sub-faces, we can thus consider each edge at a given resolution level to be the parent of four child edges (or fewer than four if the edge is on a mesh boundary) of the *same orientation* in the next higher-resolution level mesh. Such a parent-child hierarchy, which we have implemented in our source code, is illustrated in Fig. 5.

In Fig. 5, the *root* parent edges (*AB*, *BC*, *AC*, *BD* and *CD*) represent edges of the base mesh, and triangles *ABC* and *BDC* represent two triangles at the base mesh resolution (level of detail 0, or LoD_0). Each of these triangles is divided into four triangles at LoD_1. In an edge-based tree associated with root edge *BC*, its children are the four sub-edges of the *same orientation* at the next higher resolution level – that is, the sub-edges *BG*, *GC*, *EF* and *HI*. The only wavelet coefficient at LoD_1 that is associated with the base edge *BC* is the one corresponding to the midpoint of that edge (i.e., at vertex *G*).

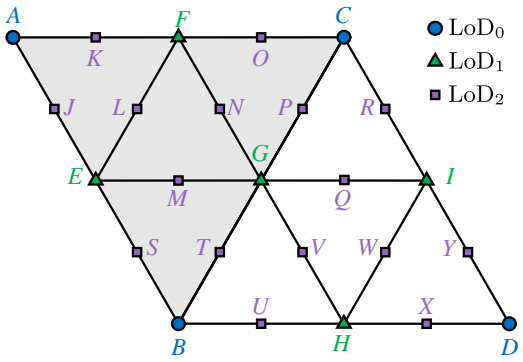


Fig. 5: Edge-based parent-child relationships for wavelet coefficients in a subdivision surface hierarchy for a triangular mesh.

The wavelet coefficients at LoD_2 that are associated with the base edge *BC* are the ones corresponding to the midpoints of its child edges (i.e., at vertices *T*, *P*, *L*, *W*). A similar process can be followed to determine the wavelet coefficients associated with the other base edges of the triangles *ABC* and *BDC*. Note that edge *AB* is a boundary edge, so it has only 3 child edges at the next resolution level (*AE*, *EB* and *FG*); similarly for edge *AC*. For each base mesh edge, we only consider descendant edges of the *same orientation* as the parent edge, such that no wavelet coefficient is accounted for multiple times or left out.

Our proposed modifications to the V-DMC intra-frame encoder and decoder frameworks are shown in Fig. 6. At the encoder, we replace the "Image Packing/Unpacking and Video Encoding/Decoding" blocks with our "Zerotree Coding" and "Wavelet Coefficient Reconstruction" blocks. At the decoder, the "Video Decoding and Image Unpacking" block is replaced by our "Zerotree Decoding and Wavelet Coefficient Reconstruction" block. The details of our modifications are explained in the following sub-sections.

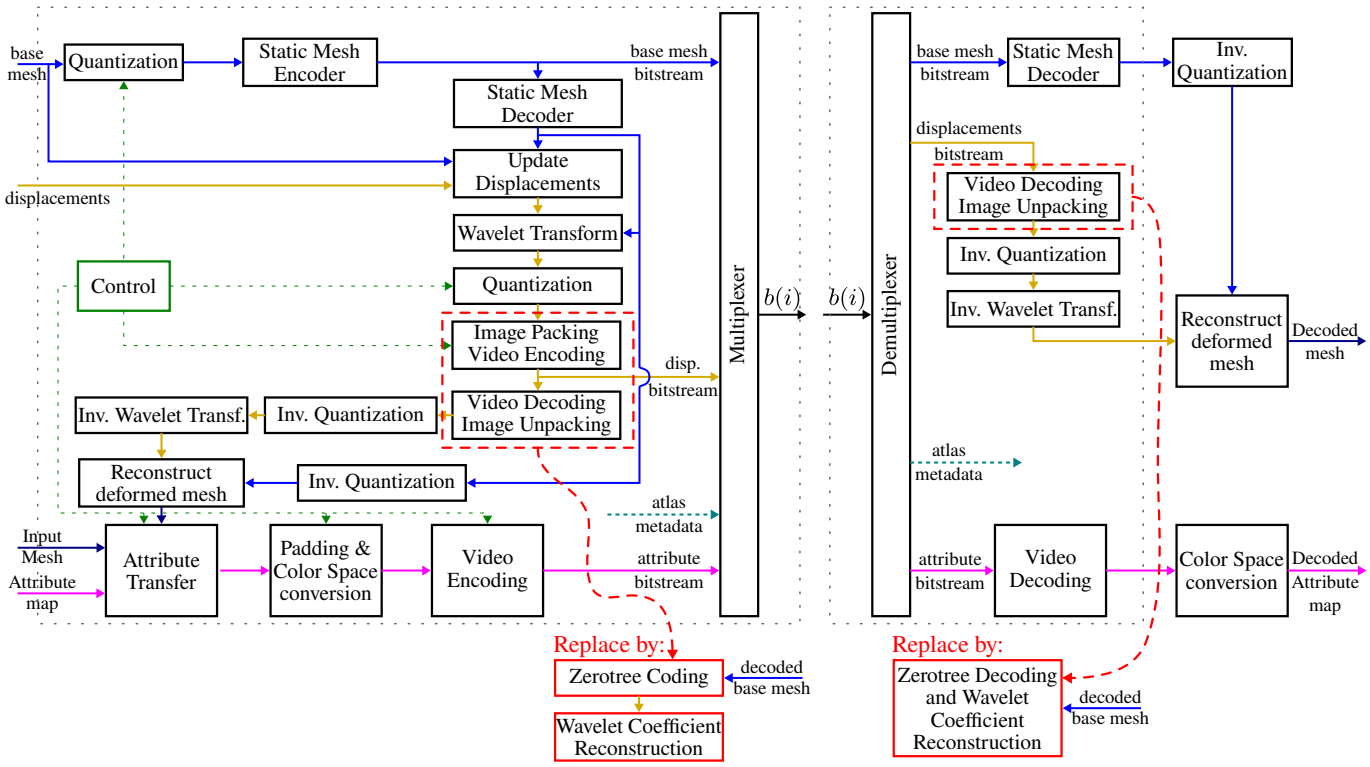
A. Encoder: Zerotree Encoding Process

The following steps are carried out at the encoder, within the "Zerotree Coding" block in Fig. 6. We assume that the input wavelet coefficients are integers before starting the zerotree coding, which usually means a pre-quantization. This pre-quantization is done in the current V-DMC TM, anyway. Our encoder diagram is depicted in Fig. 7.

First, we separate out the base mesh displacements from the wavelet coefficients. The base mesh displacements do not represent wavelet coefficients, but low-pass scaling coefficients as they are associated with the base mesh vertices and not the edges. So, we do not code these with the zerotree coder. They are coded separately, for example using a suitable entropy coder. In our current implementation, we code these using a Run-Length Golomb-Rice (RLGR) entropy coder.

For each unique base mesh edge, we construct an edge-based tree by traversing all the available subdivision levels, as in the hierarchy structure in Fig. 5, such that the number of trees are equal to the number of unique base mesh edges.

In order to define what a "significant" wavelet coefficient magnitude is, we choose an initial threshold $T_0 =$



(a) Encoding process (b) Decoding process

Fig. 6: Our proposed changes (shown in red) to the V-DMC intra-frame encoder (a) and decoder (b).

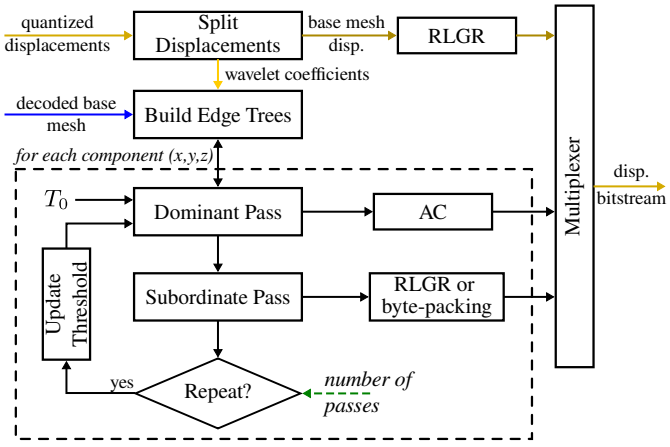


Fig. 7: The proposed Zerotree encoder.

$2^{\lfloor \log_2(c_{max}) \rfloor}$, where c_{max} is the magnitude of the largest coefficient in the list. We compute a separate threshold for each component (x, y, z)[4] of the wavelet coefficients, and these initial thresholds are transmitted to the decoder. We do this because it has been found in [18] that treating the three components separately for the dominant and subordinate passes, and separately encoding the results, tends to work better than considering a sort of vector quantization method to encode the three components together. This also means that we actually have three zerotrees per unique base mesh edge in our edge-based hierarchy.

Then we use the zerotree coding approach presented in Section III to perform the *dominant pass*, according to the threshold T_0 for the corresponding component. The chosen scanning order should be such that no child node is scanned before its parent node, and all nodes in a given subband (resolution level) are scanned before moving onto the next resolution level. Each of the dominant symbols can be represented using 2 bits by default, e.g., `00` (POS), `01` (NEG), `10` (ZTR), `11` (IZ), but we choose to entropy-code them using an adaptive arithmetic coder (AC) as this generally produces a lower bitrate. For all the coefficients that are found to be significant (i.e., POS or NEG) in the dominant pass, we put their magnitudes into a Subordinate List (separately for each component), to be refined. We then set these significant coefficients to 0 in our coefficient hierarchy, so they will not be considered again in the following dominant pass.

Because of the way we define T_i, we know that no coefficients have magnitudes greater than $2T_i$, so the "uncertainty interval" for the significant coefficient values from the i-th dominant pass is $[T_i, 2T_i)$. The *subordinate* (or *refinement*) pass refines the magnitudes of the significant coefficients by indicating whether they lie in the bottom half of the uncertainty interval (i.e., in $[T_i, T_i + T_i/2)$), or in the top half (i.e., in $[T_i + T_i/2, 2T_i)$). The output of the subordinate pass is a `0` bit if the magnitude of the corresponding significant coefficient lies in the bottom half of the interval, and a `1` bit if it lies in the top half. The order of bits that are output in the refinement pass follows the output order of the significant coefficients

[4]Throughout this paper, we use the generic 3D component names (x, y, z), but this could also be something else, such as n (normal), t (tangent), b (bitangent), if a different coordinate system is used.

from the corresponding dominant pass. If multiple dominant and refinement passes are performed, at each iteration the Subordinate List contains significant coefficient magnitudes from previous passes as well, and these are iteratively refined with each new refinement pass, using the uncertainty intervals associated with each new (narrower) threshold.

We next check if entropy-coding the sequence of bits resulting from the subordinate pass (separately for x, y, z) produces a lower bitrate than simply packing these bits into bytes directly without entropy coding, and we transmit the smaller bitstream out of the two. A corresponding flag needs to be set in our bitstream to indicate whether entropy coding is used or not. In our current implementation, we use the RLGR entropy coder for the case where entropy coding is chosen for the refinement bits.

Next, we must decide if we would like to perform additional passes. In our current implementation, we set a maximum number of desired dominant and refinement passes as encoder and decoder parameters. This allows the encoder to encode everything up to the chosen number of passes, while the decoder can continue its decoding process up to its desired number of passes. Note that our encoder can automatically figure out when no more refinement is possible for the wavelet coefficients' magnitudes (when the length of the uncertainty interval is 1), so even if the user chooses a higher number of zerotree passes, the encoder does not continue to send unnecessary bits to the decoder in this case. Alternatively, we could define a target bitrate and/or a target distortion measure (currently not implemented).

Before the N-th dominant pass, we set a new significance threshold (for each component), as: $T_N = \lfloor T_{N-1}/2 \rfloor$.

In the "Wavelet Coefficient Reconstruction" block in Fig. 6, the wavelet coefficients are updated with their corresponding reconstruction value, mimicking what happens at the decoder side (see Section IV-B), such that texture transfer can be performed on the reconstructed mesh.

B. Decoder: Zerotree Decoding Process

In Fig. 8, we show the internals of the "Zerotree Decoding and Wavelet Coefficient Reconstruction" block in Fig. 6.

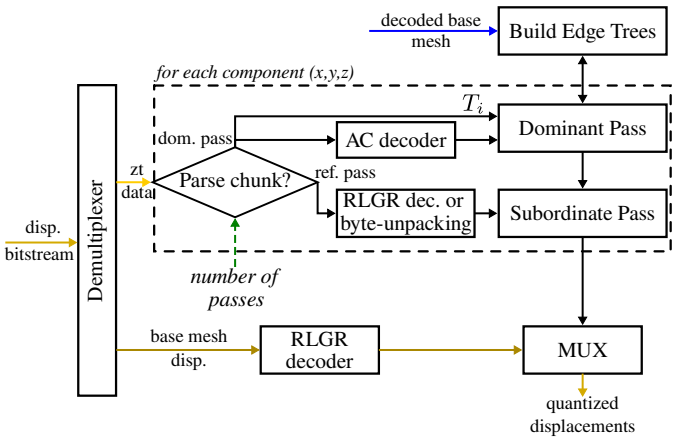


Fig. 8: The proposed Zerotree decoder.

First, we reconstruct the edge-based zerotree hierarchy using the decoded base mesh as the starting point, exactly as at the encoder. We, then, parse the received displacements bitstream and separate out the data corresponding to the encoded base mesh displacements (if these exist), the data corresponding to the zerotree dominant and refinement passes for each component (x, y, z) of the wavelet coefficients, the initial significance thresholds for the first zerotree dominant pass, and the auxiliary data needed for entropy-decoding.

We next entropy-decode the base mesh displacements (if these exist). No other reconstruction is needed, since the base mesh displacements are encoded losslessly.

For the zerotree-related data, we first entropy-decode the bits corresponding to the symbols of the first dominant pass. Using the constructed zerotree hierarchy and the traversal order that is common to the encoder and decoder, the decoder can use the decoded dominant symbols to obtain the positions of the current significant wavelet coefficients within the zerotree hierarchy. If a given tree node receives a ZTR symbol, the decoder sets the wavelet coefficients for this node, as well as for all of its descendants in the hierarchy, to 0 – this is a key aspect of the zerotree coding mechanism, which enables zeroing out insignificant coefficients at the decoder. Next, we entropy-decode the refinement bits (if they were entropy-coded at the encoder) for the first zerotree refinement pass, and use these to reconstruct the magnitude values of the significant wavelet coefficients decoded so far. In order to reconstruct the wavelet coefficient magnitudes, we use the centres of the uncertainty intervals (see Section IV-A) corresponding to the threshold values for the current dominant pass. If a threshold is equal to 1, we choose to reconstruct the corresponding coefficient magnitude as 1 (i.e., the lower limit of the interval $[1, 1.5)$) to obtain an integer value. Similarly, if we reach an uncertainty interval length of 1, we also reconstruct the coefficient as the lower limit of the interval, not as the interval midpoint, in order to maintain integer outputs. Note that once we are beyond the first refinement pass, if there exist wavelet coefficient magnitudes to refine from previous passes, for the coefficient reconstruction we first refine the uncertainty interval where the coefficient was found in the previous pass, and then use the centre of this refined uncertainty interval.

If there exist further dominant passes to decode and/or further refinements to do for the already-decoded significant wavelet coefficients, we halve the significance thresholds (as at the encoder), and repeat the above steps, except for the entropy-decoding of the base mesh displacements.

The decoder can stop the decoding process at any point when satisfied with the reconstructed coefficient values and/or when they have reached a bitrate limit or some pre-defined number of zerotree passes. Since the received bitstream is *embedded*, at each decoding of a new set of dominant and subordinate pass bits, the signal reconstruction is complete. That is, *all* the input displacement values have been reconstructed, but the quality (precision) of this reconstruction becomes progressively more accurate as more bits are decoded. The high-level layout of our embedded bitstream is shown in Fig. 9.

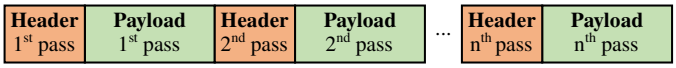


Fig. 9: The layout of our embedded bitstream. The payload for the first pass also contains the encoded base mesh displacements, while the subsequent passes contain only the zerotree data for the wavelet coefficients.

V. Results and Analysis

In Fig. 10, we present rate-distortion (R-D) curves for our results on all 300 frames of the V-DMC dynamic mesh dataset [24], without texture coding, using V-DMC TM v4.0. These R-D curves represent the following:

- Lossy intra-coding.
- HEVC and AC anchors with the standard v4.0 Common Test Conditions (CTC) rate points and parameters [24].
- A separate R-D curve for our zerotree method, for each CTC rate point. Each point on the "zt" curves represents a new zerotree pass (dominant and subordinate pass), such that the only thing that changes along the curve is that the wavelet coefficient magnitudes are progressively refined by an increasing number of passes.
- x-axes representing the total geometry bits (base mesh + displacements + motion) + metadata and any overhead bits, y-axes representing the geometry D1 PSNR as computed by the CTC metrics [24], across all 300 frames.

To obtain the results in Fig. 10, we used the V-DMC v4.0 CTC rate points as targets for a lossless-displacement reconstruction[5]. Our goal was to prove that, given a starting base mesh and a target subdivision surface (target rate point and geometry reconstruction quality), our proposed zerotree coding would enable us to achieve a smooth progression of lossy-to-lossless R-D points with an increasing number of dominant and refinement passes. This is evident in our results in Fig. 10, where we see, for each CTC rate point, a smooth curve of intermediate R-D points generated by the zerotree method, each representing a successive dominant and refinement pass. This progression of R-D points is produced automatically by our encoder and decoder, with no additional parameter setting by the user, except for a maximum number of zerotree passes.

In Fig. 11, we compare our zerotree method with the HEVC and AC anchors at the *lossless*-displacement CTC rate points. The results shown here are representative of our results for all 8 input mesh sequences [24]. The zerotree R-D curves in these plots were obtained by connecting the 5th zerotree pass points from the curves for "zt-r01", "zt-r02", "zt-r03" and "zt-r04" for the corresponding mesh sequences in Fig. 10.[6] We can observe in Fig. 11 that for all the mesh sequences, our zerotree method is either very close to, or overlapping with, the R-D curves of the HEVC and AC anchors. The corresponding *Bjøntegaard-Delta* (BD)-rates [25] between our proposed solution and the HEVC anchor are shown in Table I. While the results in Fig. 11 and Table I indicate that in some cases the zerotree method requires slightly more bits than the anchors for lossless reconstruction, this overhead is justified by the embeddedness and quality scalability of our method, which allows us to easily extract several levels of progressive geometry reconstructions within the same bitstream, while the anchors encode only one quality resolution level (lossless) at a similar bitrate.

[5]Lossless with respect to the *pre-quantized* displacement values.

[6]The number of zerotree passes needed to reach a lossless reconstruction can vary depending on the input distribution of wavelet coefficients.

TABLE I: BD-rates corresponding to the results in Fig. 11 comparing our proposed method with the default HEVC displacements coding in V-DMC.

C1-ai – lossy geometry [all-intra]				
Class	**Sequence (_voxelized)**	**D1 PNSR [24]**	**D2 PSNR [24]**	**IBSM Geom. [26]**
cat1-A	longdress	3.0%	3.0%	2.9%
	soldier	2.9%	3.0%	3.1%
	average	**2.9%**	**3.0%**	**3.0%**
cat1-B	basketball_player	3.1%	3.1%	3.2%
	dancer	2.7%	2.7%	2.7%
	average	**2.9%**	**2.9%**	**2.9%**
cat1-C	football	0.1%	0.1%	-0.1%
	levi	2.9%	2.6%	0.8%
	mitch	1.4%	1.4%	1.4%
	thomas	1.2%	1.2%	0.3%
	average	**1.4%**	**1.3%**	**0.6%**
Overall average		**2.2%**	**2.1%**	**1.8%**

In Fig. 12, we present some example progressive geometry reconstructions obtained by our zerotree system over multiple dominant and refinement passes, for a given CTC rate point. These reconstructions correspond to our "zt-r01" R-D curves in Fig. 10. Note that the number of vertices and triangles in Fig. 12 remains the same in all of the meshes from left to right; the only thing that changes is the precision (quality) of the geometry (x, y, z) reconstructions. We deliberately show these reconstructions without texture here, so that we can clearly see the finer geometric details that are progressively reconstructed, which may otherwise be hidden by the texture maps. In these reconstructions, we can truly see the benefit of the quality scalability provided by our zerotree solution – an increasingly smoother geometry reconstruction, without changing the mesh spatial resolution. In Fig. 12, we can also observe that even before reaching a lossless displacement reconstruction, one or more of the lossy reconstructions that we can achieve at a (much) smaller geometry bitrate can still be visually acceptable.

VI. Conclusion

In this paper, we presented an alternative solution for encoding the subdivision wavelet coefficients in the current state-of-the-art system for dynamic time-varying mesh coding: the MPEG V-DMC. Our solution applies zerotree coding on top of an edge-based tree hierarchy that is constructed across different resolution levels in each frame of a dynamic mesh sequence, and we code the "base mesh displacements" separately. Given a base mesh and a target subdivision surface, our proposed system produces a smooth progression of lossy-to-lossless geometry reconstructions, with very little

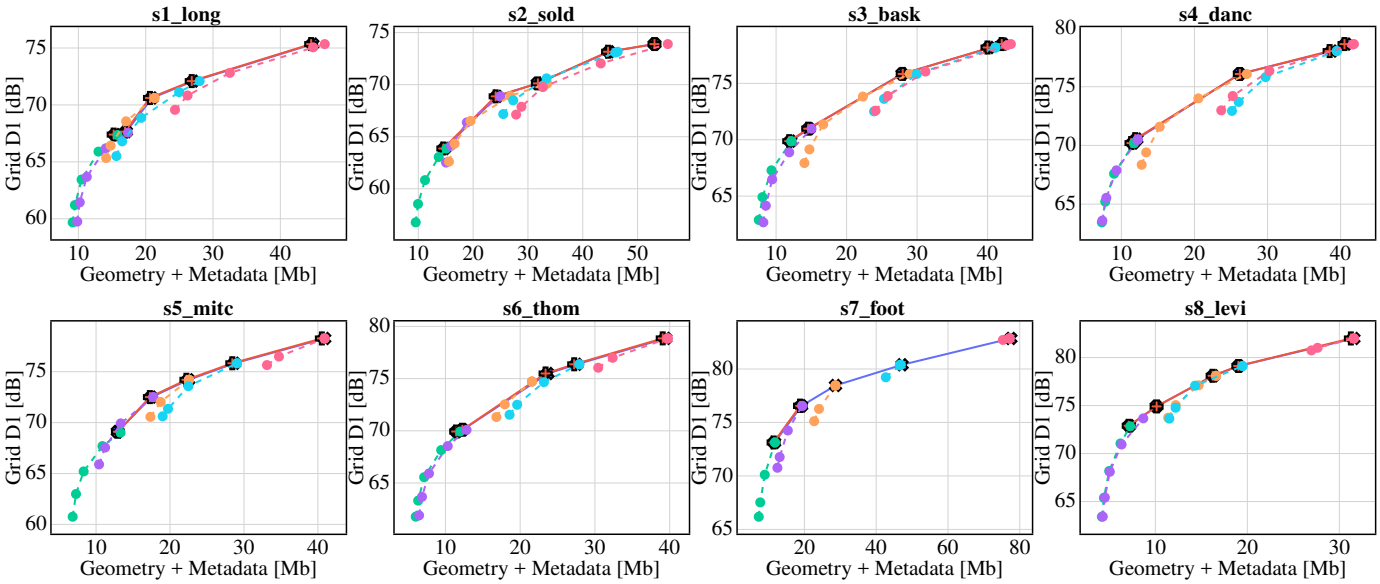


Fig. 10: R-D curves for 300 frames without texture coding, for our integration of the proposed zerotree coding method into the V-DMC TM v4.0. (✕): anchor-HEVC, (+): anchor-AC, (●): zt-r01, (●): zt-r02, (●): zt-r03, (●): zt-r04, (●): zt-r05.

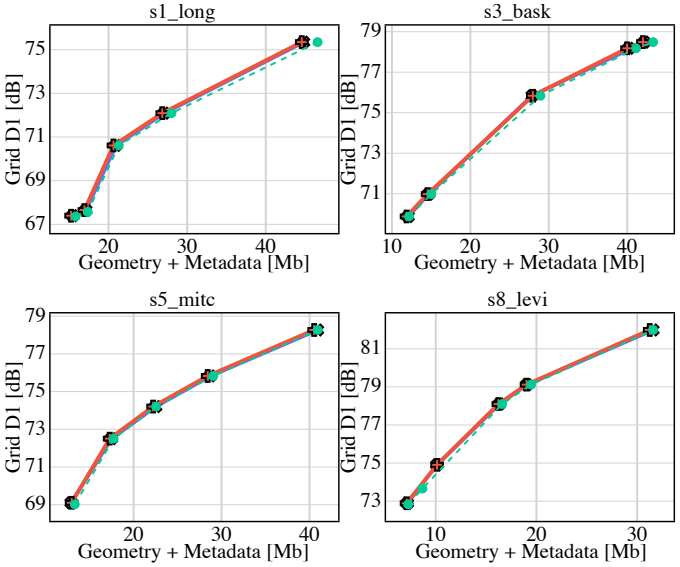


Fig. 11: R-D curves for 300 frames without texture coding, at the lossless-displacement rate points only, for our integration of the proposed zerotree coding method into the V-DMC TM v4.0. (✕): anchor-HEVC, (+): anchor-AC, (●): zt.

input from the user. This adds the desirable functionality of *quality scalability* of the mesh geometry, which is currently missing from the V-DMC framework. The embededness of our bitstream provides the encoder and decoder with the ability to tailor the bitrate and quality of the displacements bitstream according to their needs. Such codec capabilities will undoubtedly be important for a number of applications of dynamic TVMs in the future, where different users are likely to have different quality and bitrate requirements. We also believe that such capabilities would be important to ensure that the V-DMC standard is applicable to a wide range of use cases, and able to cater to heterogeneous needs of diverse users, systems, and applications. Furthermore, our proposed solution and the associated self-contained embedded bitstream have a very simple high-level parameterization that makes the solution easily integrable into the V-DMC software, or any other system that contains subdivision wavelet coefficients. The memory consumption and encoding times of our current implementation are generally comparable to that of HEVC, but further implementation optimizations are still very much possible – for example, our code is highly parallelizable. We believe that the proposed zerotree coding system would be even more effective when there is a larger number of subdivision levels in the wavelet hierarchy and/or when the base mesh is much simpler. In such cases, there would be fewer zerotrees in total and the zerotrees would be longer, which would lead to longer hierarchical sequences of wavelet coefficients that are more easily compressible.

It is important to remember that meshes are inherently different to 2D videos and should be treated as such. It is likely that for future applications of dynamic TVMs, a multi-resolution, scalable coding approach will not only be a “nice-to-have” feature, but a requirement. For example, in a volumetric video that contains more than one subject or object, it will be essential that the mesh codec is able to encode and decode different parts of this scene with different levels of both spatial and quality resolution, depending on the viewer’s interest. Our contribution in this paper provides an important step in this direction. To the best of our knoweldge, zerotree-based coding mechanisms have not previously been applied in the context of compressing dynamic TVMs. Also, differently to the known existing methods that use zerotree-like coders for subdivision wavelet coefficients on meshes, our proposed method utilises zerotree coding directly, without any prior set partitioning of the wavelet coefficients (most of the other existing methods use image-based SPIHT coding).

As future work, we can similarly envisage scalable solutions

Fig. 12: Examples of progressive geometry reconstructions corresponding to "zt-r01" from Fig. 10, using (left to right): 1, 2, 3, 4, 5 zerotree passes.

for texture compression (e.g., in V-DMC), such as image-based zerotree coding, Scalable-HEVC, or a mipmap approach.

REFERENCES

[1] A. Maglo, G. Lavoué, F. Dupont, and C. Hudelot, "3D Mesh Compression: Survey, Comparisons, and Emerging Trends," *ACM CSUR*, vol. 47, no. 3, pp. 1–41, 2015.

[2] J.-E. Marvie, M. Krivokuća, and D. Graziosi, "Chapter 14: Coding of Dynamic 3D Meshes," in *Immersive Video Technologies*. Elsevier Inc., 2023, pp. 387–423.

[3] 3DG, "CfP for Dynamic Mesh Coding," ISO/IEC JTC 1/SC 29/WG 7, Online, Tech. Rep. N00231, 2021.

[4] K. Mammou, J. Kim, A. M. Tourapis, D. Podborski, and D. Flynn, "Video and Subdivision based Mesh Coding," in *EUVIP2022*. Lisbon, Portugal: IEEE, 2022, pp. 1–6.

[5] K. Mammou, J. Kim, A. Tourapis, D. Podborski, and K. Kolarov, "[V-CG] Apple's Dynamic Mesh Coding CfP Response," ISO/IEC JTC 1/SC 29/WG 7, Online, Tech. Rep. m59281, Apr. 2022.

[6] J. Lounsbery, "Multiresolution Analysis for Surfaces of Arbitrary Topological Type," Phd Thesis, Dept. of Computer Science and Engineering, UW, Seattle, WA, USA, 1994.

[7] M. Lounsbery, T. D. DeRose, and J. Warren, "Multiresolution analysis for surfaces of arbitrary topological type," *ACM TOG*, vol. 16, no. 1, pp. 34–73, 1997.

[8] W. Sweldens, "The Lifting Scheme: A Construction of Second Generation Wavelets," *SIAM J. Math. Anal.*, vol. 29, no. 2, pp. 511–546, 1998.

[9] Z. M. Miličević and Z. S. Bojković, "Performance of High Efficiency Video Coding (HEVC) HM-16.6 Encoder," in *TELFOR*. IEEE, 2015, pp. 712–715.

[10] E. Catmull and J. Clark, "Recursively generated B-spline surfaces on arbitrary topological meshes," *Seminal graphics: pioneering efforts that shaped the field*, p. 183–188, Jul. 1998.

[11] D. Doo and M. Sabin, "Behaviour of recursive division surfaces near extraordinary points," *Seminal graphics: pioneering efforts that shaped the field*, p. 177–181, Jul. 1998.

[12] C. Loop, "Smooth Subdivision Surfaces Based on Triangles," Master's thesis, Department of Mathematics, University of Utah, Jan. 1987.

[13] N. Dyn, D. Levine, and J. A. Gregory, "A butterfly subdivision scheme for surface interpolation with tension control," *ACM Trans. Graph.*, p. 160–169, Apr. 1990.

[14] J. Peters and U. Reif, "The simplest subdivision scheme for smoothing polyhedra," *ACM Trans. Graph.*, p. 420–431, Oct. 1997.

[15] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *SIGGRAPH*, USA, Aug. 1997, p. 209–216.

[16] J. Rossignac, "Edgebreaker: connectivity compression for triangle meshes," *IEEE Trans. Vis. Comput. Graphics*, vol. 5, pp. 47–61, 1999.

[17] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3445–3462, 1993.

[18] A. Khodakovsky, P. Schröder, and W. Sweldens, "Progressive geometry compression," in *SIGGRAPH*, USA, 2000, pp. 271–278.

[19] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 243–250, 1996.

[20] A. Khodakovsky and I. Guskov, "Normal mesh compression," *Geometric modeling for scientific visualization*, pp. 189–206, 2002.

[21] J.-Y. Sim, C.-S. Kim, C.-C. Kuo, and S.-U. Lee, "Rate-distortion optimized compression and view-dependent transmission of 3-d normal meshes," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 7, pp. 854–868, 2005.

[22] F. Moran and N. Garcia, "Hierarchical coding of 3d models with subdivision surfaces," in *ICIP*, vol. 2. IEEE, 2000, pp. 911–914.

[23] W. Guan, J. Cai, J. Zhang, and J. Zheng, "Progressive coding and illumination and view dependent transmission of 3-D meshes using RD optimization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 4, pp. 575–586, 2010.

[24] 3DG, "Common Test Conditions for V3C and V-DMC," ISO/IEC JTC 1/SC 29/WG 7, Geneva, Tech. Rep. N00685, 2023.

[25] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," *VCEG-M33*, 2001.

[26] J.-E. Marvie, Y. Nehmé, D. Graziosi, and G. Lavoué, "Crafting the MPEG metrics for objective and perceptual quality assessment of volumetric videos," *Qual. User Exp.*, vol. 8, no. 1, Jun. 2023.