ORIGINAL PAPER

Video coding of dynamic 3D point cloud data

SEBASTIAN SCHWARZ,[1] NAHID SHEIKHIPOUR,[1,2] VIDA FAKOUR SEVOM[2] AND MISKA M. HANNUKSELA[1]

Due to the increased popularity of augmented (AR) and virtual (VR) reality experiences, the interest in representing the real world in an immersive fashion has never been higher. Distributing such representations enables users all over the world to freely navigate in never seen before media experiences. Unfortunately, such representations require a large amount of data, not feasible for transmission on today's networks. Thus, efficient compression technologies are in high demand. This paper proposes an approach to compress 3D video data utilizing 2D video coding technology. The proposed solution was developed to address the needs of "tele-immersive" applications, such as VR, AR, or mixed reality with "Six Degrees of Freedom" capabilities. Volumetric video data is projected on 2D image planes and compressed using standard 2D video coding solutions. A key benefit of this approach is its compatibility with readily available 2D video coding infrastructure. Furthermore, objective and subjective evaluation shows significant improvement in coding efficiency over reference technology. The proposed solution was contributed and evaluated in international standardization. Although it is was not selected as the winning proposal, as very similar solution has been selected developed since then.

Keywords: Point cloud coding, Volumetric video, Immersive media, AR

Received 16 April 2019; Revised 11 November 2019

1. INTRODUCTION

Recent years have shown significant advances in immersive media experiences. Volumetric videos, such as dynamic point clouds, allow for new forms of entertainment and communication, like immersive tele-presence as shown in Fig. 1. However, efficient compression technologies to allow for distribution of such content are still sought after.

Volumetric video data describes a 3D scene or object with its geometry (shape, size, position in 3D-space) and respective attributes (e.g. color, opacity, reflectance, albedo, etc.), plus any temporal changes. Such data is typically generated from 3D models, i.e. CGI, or captured from real-world scenes, using a variety of solutions, e.g. multi-camera or a combination of video and dedicated geometry sensors. Common representation formats for such volumetric data are polygon meshes or point clouds. Temporal information is included in the form of individual capture instances, similar to frames in a 2D video, or other means, e.g. position of an object as a function of time. Because volumetric video describes a complete 3D scene (or object), such data can be viewed from any viewpoint. Therefore, volumetric video is a key enabling technology for any AR, VR, or MR applications, especially for providingsix degrees of freedom (6DoF) viewing capabilities.

Unfortunately, compression solutions for volumetric video representations [1–4] suffer from poor spatial and temporal compression performance. Identifying correspondences for motion-compensation in 3D-space is an ill-defined problem, as both, geometry and respective attributes may change. For example, temporal successive "frames" do not necessarily have the same number of polygons, points, or voxels. Therefore, compression of dynamic 3D scenes is inefficient. Also, current 3D data compression approaches, such as [2], do not utilize spatial prediction within an object, like intra-prediction within a video frame. Previous 2D-video-based approaches for compressing volumetric data, e.g. multiview + depth [5], have much better spatial and temporal compression efficiency, but rarely cover the full scene or object. Hence, they provide only limited 6DoF support.

In this paper, a novel volumetric video compression approach, based on coding projected 3D data in 2D images, is proposed. Employing standard 2D video coding technology, our proposal benefits from several decades of video coding research. Spatial and temporal compression efficiency is highly improved over the state-of-the-art compression technology [2]. On average, required bit rates are reduced by around 75% for geometry, respectively 50% for color attribute compression. Furthermore, software and hardware coding solutions are readily available for real-time implementation. As 2D video coding standards, such as

[1] Nokia Technologies, Hatanpään Valtatie 30, 33100 Tampere, Finland
[2] Tampere University of Technology, Korkeakoulunkatu 6, 33720 Tampere, Finland

Corresponding authors:
Sebastian Schwarz.
E-mail: sebastian.schwarz@nokia.com

Fig. 1. Example of tele-immersive experience.

H.264/MPEG-4 AVC or HEVC, are supported by billions of devices and distribution solutions, such as Dynamic Adaptive Streaming over HTTP, are widely used, such a solution could be quickly deployed in products and services.

The research presented in this paper was carried out under the scope of the ISO/IEC JTC1/SC29/WG11 (MPEG) Call for Proposals (CfP) for point cloud compression(PCC) [6]. Thus, background, approach, and evaluation will be provided in the context of point clouds. Nonetheless, described concepts are easily translated to any other form of volumetric video representation, e.g. polygon meshes. The proposed solution has been contributed to the MPEG CfP in October 2017. It was not selected as the winning technology itself, but a very similar proposal utilizing video-coded projections of 3D data has been selected and developed in standardization since then.

The remainder of this paper is structured as follows. First, an overview of previous research in the field is given in Section 2. Section 3 introduces the proposed projection-based approach for volumetric video compression. Then, evaluation methodology and results are provided in Section 4, before the paper is concluded in Section 5.

2. RELATED WORK

With the fast-rising number of 3D sensing technologies, dynamic 3D video is getting more important and practical for representing 3D contents and environments. For a long time, representing the world has been done using 2D cameras where the visualization of the world's components were possible. Nowadays, there are plenty devices which they can record and represent the world in 3D mode. Representing 3D data using polygon meshes has been employed conventionally. Nevertheless, obtaining point cloud is faster and easier because of lower complexity in their structure and computations [2]. 3D point clouds might be used widely in some environments such as virtual reality, 3D video streaming in mixed reality, and mobile mapping.

A point cloud consists of sets of points in 3D space where each point has its own spatial location along with a corresponding vector of attributes, such as colors, normals, reflectance, etc. Point cloud data produced by some high-resolution sensors, e.g. Microsoft Kinect, can be represented by high rates point clouds, e.g. a reconstructed 3D point cloud may contain several millions of points. Thus,

processing and transmitting point clouds is challenging and needs a high amount of resources because of high density of generated data.

In recent years many efforts have been devoted to promote the efficiency of compression method algorithms for such 3D point cloud data. One of the first and main classes of point cloud compressors is progressive point cloud coding. In this approach, which is mostly based on kd-tree, a coarse representation of the complete point cloud is encoded and then several refinement layers are built in order to achieve efficient compression. In this context, [7] splits a 3D space in half recursively using a kd-tree structure and points in each half are encoded. For each subdivision, empty cells are not subdivided anymore. Applying the same method and adapting it to an octree structure was proposed in [8] and [9]. These approaches achieve better results because they are also considering connectivity information. The work in [10] exploits a similar work to the mentioned octree-based approaches, while introducing a novel prediction technique for surface approximation.

Besides geometry compression algorithms, there are many methods considering attribute compression. For example, the octree structure is adopted in [11] and attributes are treated as signals. A graph in each level of octree is constructed for the leaves (connecting nearby points in a small neighborhood). Then the graph transform (Karhunen–Loeve) is employed in order to decorrelate color attributes on the graph. A hybrid color attribute compression is introduced in [12], where two different compressing modes, run-length coding and palette coding, are applied on each block. After comparing the distortion values with the defined thresholds, the final block with the best corresponding compression mode is selected. Wang *et al.* [13] introduced a method to compress both object data and environment data using three-dimensional discretecosine transform (3D-DCT). In their work, capturing signal properties in frequency domain, using DCT, attains more information from the original raw data.

While most of the above-mentioned (PCC) approaches are focusing on static point cloud, there are many efforts on compression of dynamic point cloud sequences. Thanou *et al.* [14] proposed a first work dealing with dynamic voxelized point clouds. In their work a time-varying point cloud codec is introduced, where point clouds are represented as sets of graphs. This method is using wavelet-based features to find matches between points and predicts an octree structure for adjacent frames. In [1], the geometry compression is completed by considering an octree structure for each frame of point cloud sequences. Unfortunately, 3D motion estimation is one of the challenging issues in PCC, due to the lack of information in point-to-point correspondences in a sequence of frames. One of the first work concerning motion estimation coding for dynamic point clouds is introduced in [15]. In this method, the complete point cloud is divided into occupied blocks. Then, in the coding process, it is decided if either the block should be encoded in intra, i.e. not motion compensated, mode or

it is motion compensated. In a more recent paper, [2], a progressive compression framework is introduced. The proposed method, which is mostly focusing on mixed-reality and 3D tele-immersive systems, consists of a hybrid architecture. The architecture combines two existing methods i.e. octree-based structure including motion compensation and a common video coding framework. This solution was selected as reference technology for the MPEG CfP for PCC [6]. And in [16] the initial idea of projection-based coding of dynamic 3D points using multiview image coding was explored.

3. PROJECTION-BASED COMPRESSION OF 3D DATA

The proposed solution takes the benefits of traditional multiview + depth 2D video compression of 3D video [5] and provides specific extensions to improve representation and compression for volumetric video data, similar to [16] but also taking into account attributes and a dynamic sequence of 3D data. In [5] only a single flat projection plane is considered, i.e. the camera plane. For the proposed approach a 3D video object, e.g. represented as a dynamic sequence of point clouds, is projected onto one or more simple geometries. These geometries are "unfolded" onto 2D images, with two images per object: one for texture (color attribute) and other for geometry. Two examples for such geometry projections are illustrated in Fig. 2, including the resulting 2D projection planes. Standard 2D video coding technologies are used to compress the 2D projections and relevant projection metadata is transmitted alongside the encoded video bit streams. At the receiver, a decoder decodes the video and performs an inverse 2D-to-3D projection to regenerate the 3D video object, according to the received metadata. Figure 3 depicts the overall processing chain associated with the projection-based volumetric video coding solution. Figure 4 illustrates examples of the projected texture (a) and geometry (b) images of Fig. 2(b) and respective reconstructions. The individual steps are described in more details as follows.

3.0.1. 3D-to-2D projection

Before any projection is performed, the first point cloud of a volumetric video sequence is loaded and analyzed to initialize some basic projection parameters, such as its 3D bounding box, primary orientation, best feasible projection geometry, and image plane characteristics. After initialization, each point in the point cloud is projected onto the selected 2D geometry. Figure 2 depicts examples for such a projection setups and resulting 2D texture images for a projection onto a 3D geometry, e.g. a cylinder or four rectangular planes like the sides of a box. Two projection images are created, one for the color attributes (texture image) and other for the 3D depth (geometry image). There is a manifold of 3D-to-2D projection geometries and approaches, i.e. cylinders, cubes, planes, spheres, polyhedrons, etc. Different geometries may have different benefits depending on content. For the remainder of this
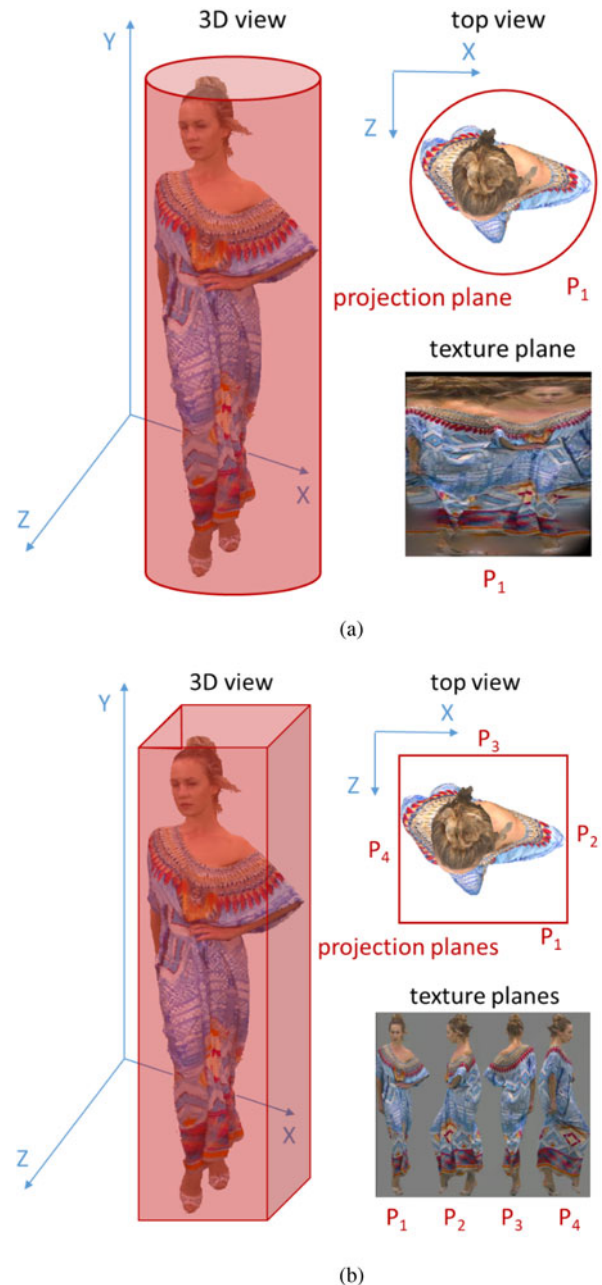

Fig. 2. Examples of 3D-to-2D projection for sequence *Longdress* provided by 8i [17]. (a) Example of 3D-to-2D projection onto a cylinder and (b) example of 3D-to-2D projection onto four rectangular planes for sequence.

paper, a series rectangular planes are chosen, where the planes rotate around the center of the 3D volume (bounding box). The number of planes, the orientation of the first plane, and orientation change between each plane can be given in the encoding parameters. The respective image resolutions are defined by the 3D bounding box and any possibly given up- or downsampling factor. Figure 2(b) illustrates this projection, the starting plane P_1 is aligned to the X-axis and there is a $90°$ rotation between each plane. The 2D projection of 3D point p at position $[x, y, z]$ with RGB texture attribute $[R_p, G_p, B_p]$ on texture image T and geometry image D for plane P1 are derived as
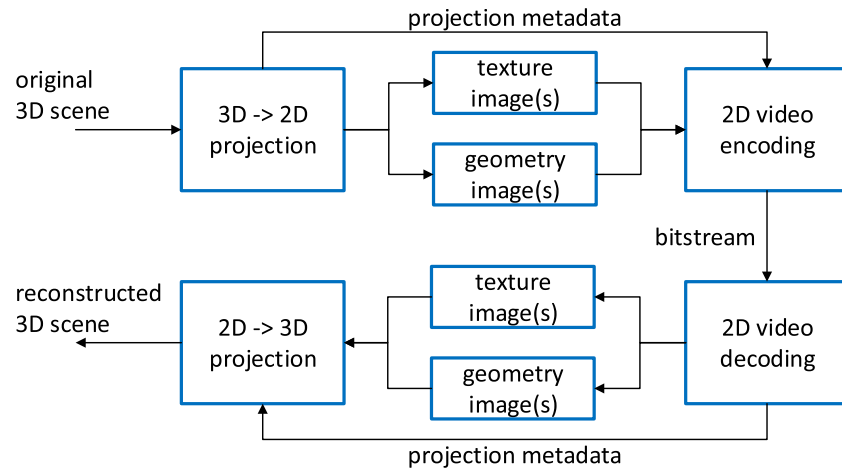
Fig. 3. Projection-based volumetric video coding workflow.

Fig. 4. Projected (a) texture and (b) geometry images for (c) original point cloud (left), as well as decoded point clouds at 13 MBit/s for the proposed solution (middle) and reference technology [2] (right).

follows.

$$T(x, y) = \begin{bmatrix} R_p \\ G_p \\ B_p \end{bmatrix} \quad \text{and} \quad D(x, y) = z \qquad (1)$$

In the case that more than one 3D point are mapped on the same 2D coordinates, depth buffering is applied to ensure that only the point closest to the projection plane is considered.

3.0.2. Reduction of projection artifacts before encoding

Whilst projection-based volumetric video coding can provide significant coding gains compared to the reference technology, it comes with some specific drawbacks, originating from the 3D-to-2D projection:

- *Occlusions*: Not all 3D points are necessarily projected onto the 2D images. Occlusions in the projections might lead to holes in the reconstructed point clouds.
- *Invalid points*: Reconstructing 3D points from decoded texture and depth images might create "invalid" 3D points due to 2D video compression artifacts.

Figure 5 depicts the effects of occluded projections (a) and invalid points (b) in a reconstructed point cloud. Both categories of artifacts can be avoided, or at least reduced, by encoder considerations.

In the case of occlusions, the extension with additional projection images seems a logical approach. However, simply adding more projection images does not increase the coverage of concave areas. Thus, *sequential decimation* is introduced: after a given number of projections, the indices of "successfully" projected points, that is points represented on at least one projection, are collected. Then it is decided to keep these points for the following projections or remove them to provide better coverage of occluded regions. By removing these already projected points, underlying points are uncovered in the following projections. An example for removing the points after each rotation is shown in
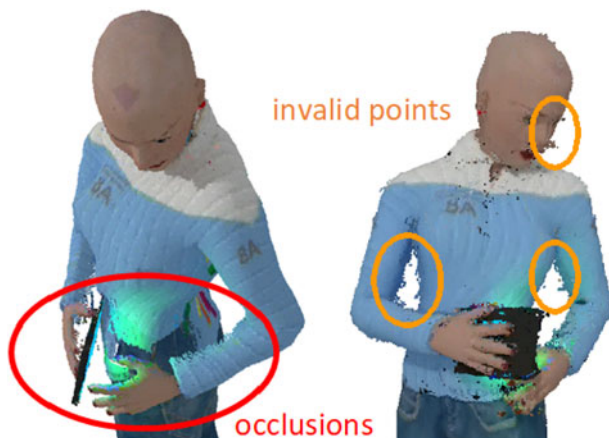

Fig. 5. Examples of projection-related artefacts: occlusions in the projected images lead to holes in the reconstructed point cloud (left), video coding distortion might lead to invalid points (right).

Fig. 6. After the first four rotations, all successfully projected points are removed and the rotation planes are shifted by $45°$ to provide a better coverage of the now uncovered remaining points, e.g. in concave regions. Figure 7 shows the resulting improved occlusion handling due to sequential decimation.

As seen in Fig. 6, *sequential decimation* can introduce a lot of small content patches, which are difficult to encode. Thus, to improve coding efficiency, the borders in the texture images can be smoothed or padded and small holes are filled by interpolation. This process is only applied to the texture image, as the geometry image is used to signal important information if a projected point is "valid" or not, i.e. should the decoder re-project this point into 3D space or not. Thus, the depth image contains high-frequency content at the patch boundaries. Quantization artifacts at these boundaries will distort the 3D reconstruction. A simple approach to dampen this effect is to introduce a depth offset z_o, i.e. (1) changed to $D(x, y) = z + z_o$, and reconstruct only values above this offset. Other, more comprehensive solutions are the transmission of dedicated validity information, e.g. a bit mask, or lossless encoding at patch boundaries. For this paper, only the depth offset was considered. Details on the texture smoothing process can be found in [18].

3.0.3. Video coding and metadata signaling

The previous steps are applied for each point cloud in a point cloud sequence. A sequence of point clouds is now represented by a sequence of texture and geometry images, i.e. a texture video and a geometry video. Standard 2D video coding is applied to encode these sequences. Additional metadata is required to signal the decoder how to reconstruct a received bit stream. The metadata is divided into sequence-dependent (static) metadata, such as projection geometry information and depth offset, and frame-dependent (dynamic) metadata, such as depth quantization. As standard 2D video coding is applied, its metadata structure, e.g. video/sequence/picture parameter sets (VPS/SPS/PPS), supplemental enhancement information (SEI) and slice headers for HEVC, can be utilized for signaling.

For this approach, texture and geometry videos are compressed as two layers using scalable HEVC (SHVC) [19]. Other implementations, e.g. 3D-HEVC or individual encodes per video are also conceivable. The SHM reference software was extended to signal the above-mentioned projection metadata. Static metadata was signaled in the SPS of the layer 0 (texture video), dynamic metadata was signaled in the slice headers of each frame of layer 0 (real-world dimensions) or layer 1 (quantization information in the geometry layer).

Some aspects of the projection geometry were hard coded, i.e. it was always assumed to have planar projections as pictured in Fig. 2(b). Only the number of projection plane, the rotation axis and offset between planes and the original orientation were signaled. No signaling for content characteristics, i.e. a compressed point cloud
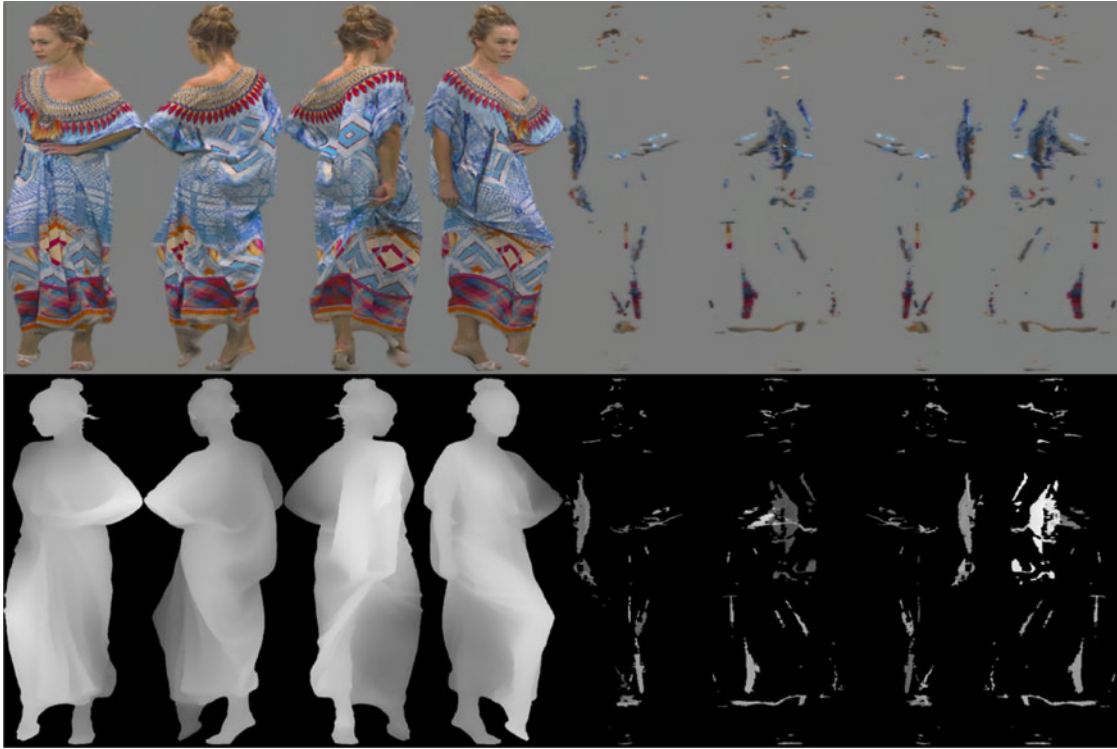
Fig. 6. Texture (top) and geometry (bottom) images for the first frame of *Longdress*, covered by eight rotations of 90° with a 45° offset and sequential decimation after four rotations.

Fig. 7. Improved occlusion handling by sequential decimation (left) versus without (right).

sequence, was implemented. Such functionality could be realized for example as SEI message. On receiving such a SEI message, the decoder would know that a volumetric video is received and would perform the relevant 2D-to-3D re-projection steps after decompressing the projection plane videos.

3.0.4. 2D-to-3D re-projection

A decoder receives the bit stream, decodes the texture and geometry images plus the projection metadata, and performs a 2D-to-3D re-projection to reconstruct the decoded 3D point cloud. Analogue to (1), the reconstructed point $\hat{p}$ at position $[\hat{x}, \hat{y}, \hat{z}]$ and RGB texture attribute $[R_{\hat{p}}, G_{\hat{p}}, B_{\hat{p}}]$ is derived from the decoded texture image $\hat{T}$ and geometry image $\hat{D}$ as follows.

$$\begin{bmatrix} R_{\hat{p}} \\ G_{\hat{p}} \\ B_{\hat{p}} \end{bmatrix} = \hat{T}(u, v), \tag{2}$$

$$\hat{x} = \frac{u}{(x_{\max} - x_{\min})} \cdot x_{\max}, \tag{3}$$

$$\hat{y} = \frac{v}{(y_{\max} - y_{\min})} \cdot y_{\max}, \tag{4}$$

$$\hat{z} = \frac{\hat{D}(u, v) - z_{o}}{(z_{\max} - z_{\min})} \cdot z_{\max}, \tag{5}$$

where u and v are the horizontal and vertical pixel coordinates in the decoded images, $x_{\min/\max}$ and $y_{\min/\max}$ are the correlating ranges to translate pixel values into real-world coordinates, and $z_{\min/\max}$ is the depth quantization range.

3.0.5. Optional inloop upsampling

In the case that there was downsampling applied during the 3D-to-2D projection described in Section 3.0.1, it might be desired to invert this by applying inloop upsampling during the 2D-to-3D reconstruction, i.e. it can be signaled to the decoder to apply an upsampling factor on one (or both) of the dimensions of the projection plane. If such a parameter is received, additional points will be reconstructed from

the decoded texture and geometry planes. Details on such a process are described in [20].

3.0.6. Optional 3D point filter

As sequential decimation is only applied after the first four rotations, some 3D points might be represented several times on different projection planes. Such points would be reprojected onto the same 3D coordinates in the reconstructed point cloud, increasing total the number of points. The effect on objective and subjective quality of such "multiple" projections is rather small. However, in terms of rendering the reconstructed point cloud, it might be beneficial to avoid multiple points. Therefore, before writing a reprojected 3D point to a PLY file, it is checked if there already exists a point at the same coordinate. If so, the new point is rejected. It would also be possible to average the color attributes of the points at the same 3D coordinate, but this is not implemented in this approach. Further post processing or filtering could be applied to the reconstructed point clouds, i.e. to reduce coding related artifacts, such as invalid points, or close remaining holes in a point cloud surface. None of such techniques are applied in the presented proposal but can be easily integrated if desired.

3.0.7. Differences to the winning MPEG CfP proposal

As mentioned earlier, the presented proposal was contributed to international standardization. Although it was not selected as the winning technology, a very similar approach proposed by Apple was selected [21]. The key differences to the technology described in this paper are the creation of 2D patches and the additional signaling of occupancy information. Instead of projecting a 3D model on 3D surfaces, the model is decomposed into 2D patches based on their surface normal. Occupancy information is signaled in the form of a video-coded binary map indicating which 2D pixels shall be reprojected into 3D space. This occupancy map is replacing the concept of depth offset discussed in Section III.2 of this paper. Figure 8 illustrates the 2D texture patches and accompanying occupancy map (the geometry image is omitted from illustration to save space). The similarity to the presented approach in Fig. 6 should be clearly visible. Thus, the overall key concept of video-based

encoding of 3D-to-2D projections remains the same and so do the main benefits of the presented idea: utilizing existing video coding technology to improve coding efficiency and reduce time-to-market. Since then, PCC technology has been developed under the acronym V-PCC, standing for video-based PCC, and is close to release as international standard [22,23].

4. EVALUATION

The evaluation was performed following the specifications of the MPEG CfP for PCC.

A total of five test sequences were evaluated at five different bit rate targets. The bit rates ranged from 3 up to 55 Mbit/s, depending on the test sequence. Figure 10 provides example views rendered from reconstructed points clouds at the four lower bit rate target points for sequence *Soldier*, using the proposed projection-based approach. Coding distortion is assessed for geometry as well as color attributes. For geometry distortion, two different metrics are applied. The first metric calculates the mean square error (MSE) between a reconstructed point and its closest point in the original point cloud. This metric is referred to as D_1, or point-to-point. The second metric calculates the MSE between a reconstructed point and a given reference surface plane. This metric is referred to as D_2, or point-to-plane. Details on these two metrics are available in [24]. Color distortion is calculated on a point-to-point level. The peak signal-to-noise-ratio (PSNR) is obtained based on the 3D volume resolution for geometry, respectively color depth for each color channel. Bjontegaard-delta (BD) metrics are derived from the distortions achieved with the provided software [25]. For a more detailed description of the evaluation process, please see the CfP document [6].

As seen in Table 1, the proposed technology achieves significant bit rate savings over the reference technology [2], with up to 90% for geometry and 50% for attribute compression in terms of the BD bit rate (BD-BR). However, the standard BD calculations are misleading in this context, as the actual rate-distortion (RD) curves often had very little overlap in the x-direction. In some cases, i.e. chroma channel distortions for sequences *Loot* and *Longdress*, there was no overlap at all and no BD-BR could be calculated. Therefore, an extension to the BD calculations was proposed in

Fig. 8. Texture (left) and occupancy (right) images for the first frame of Longdress using V-PCC.

Table 1. Bjontegaard-delta bit rate (BD-BR) results (Random Access).

Sequence	D1	D2	YUV
Queen	−87.9%	−67.3%	1.0%
Loot	−88.5%	−67.4%	−85.6%
RedAndBlack	−83.8%	−47.3%	−38.8%
RedBlack	−87.6%	−84.4%	−77.0%
Longdress	−91.5%	−75.4%	−85.38.0%
Overall	**−87.8%**	**−68.3%**	**−52.2%**

Table 2. Adapted BD-BR results and BD-PSNR results [25].

Metric	D1	D2	YUV
		Random access	
BD-BR	−75.2%	−56.5%	−49%
BD-PSNR	7.1 dB	5.4 dB	2.2 dB
		All intra	
BD-BR	−44.1%	−51.0%	6.7%
BD-PSNR	3.4 dB	3.7 dB	0.55 dB

[25]. The results using this extension, including BD-PSNR are summarized in Table 2. Figure 11 provides the RD curves for the sequences *Queen* (a), *RedBlack* (b), *Loot* (c), *Soldier* (d), and *Longdress* (e).

Apart from the objective evaluation, an extensive subjective evaluation for all CfP submissions was performed by two independent labs [26]. Only results for the test sequences *RedAndBlack* (left), *Soldier* (middle), and *Longdress* were subjectively evaluated, and only at the four lowest target rate points. An extract of their report [26] is provided as the rate-quality distortion graph is shown in Fig. 9. A significant improvement of the proposed technology (red line) in terms of subjective quality over the reference technology (blue line) is proven. For higher bit rates, the visual quality came even close the original, uncompressed data (green line). It should be noted that only the lower four bit rates were evaluated in the subjective quality assessment, e.g. the target bit rates shown in Fig. 10. It is possible that the proposed solution could achieve close to visual lossless compression at the highest bit rate target R05. Despite the good coding performance at higher bit rates, the proposed solution was clearly outperformed by the winning submission, especially at the lower bit rates. This is mainly due to the fact that the discrete occupancy information signaling has been proven superior to the depth offset solution when

higher compression is applied, as well as the better occlusion handling using 2D patch decomposition.

A key functionality of the proposed solution is the coverage of occluded points with additional projections, as described in Section 3.0.1. Without this *sequential decimation* functionality, the reconstructed point clouds will show holes which result in significantly degraded subjective quality. However, this quality degradation is not well represented in the selected objective metrics, especially D2 (point-to-plane). At lower bit rates, significantly higher D2 quality scores could be achieved by ignoring these occluded parts. Table 3 summarizes the maximum achievable objective scores if *sequential decimation* is deactivated for the lowest two bit rates.

Concluding this evaluation section, it should be mentioned that the encoder and decoder run time numbers are significantly larger for the proposed solution compared to the reference software. The numbers vary depending on the target bit rates, as the reference software performs exceptionally fast on lower bit rates. But even at higher qualities, the proposed solution takes around 100 times longer to encode and 10 times longer to decode, as shown in Table 4. However, the current implementation is based on experimental reference software [19], which is known for its poor computational performance. There exist plenty of real-time hardware and software solutions for 2D video coding. For example parallelization of texture and geometry image coding (one encoder/decoder per image) could cut down processing times further. The volumetric video specific part of this approach is rather simple and requires less than 1% of the current run times. Thus, a real-time implementation of this proposal is feasible. Still, parallel processing can also be introduced to the projecting and re-projecting part: as points in a point cloud are independent, the 3D-to-2D projection can be highly parallelized, up to one process per point. The same goes for the reverse 2D-to-3D projection at the decoder, in this context to up to one process per pixel, although other configurations such as one process per column, row or block could be more feasible.

Regarding expected memory usage, the proposed approach is relying on the underlying 2D video coding solutions and chosen coding structure. Memory efficient codecs, such as BBC's Turing codec [27], can reduce memory consumption drastically. As for the current
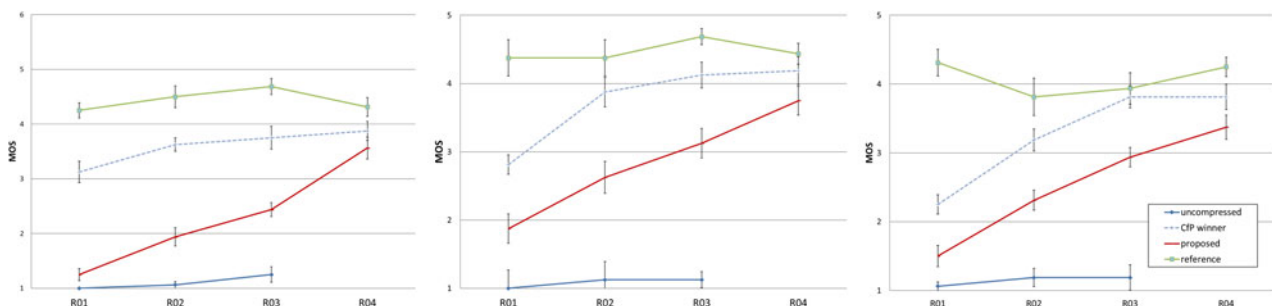

Fig. 9. Subjective MOS scores for sequences *RedAndBlack* (left), *Soldier* (middle), and *Longdress* (right) [26], where "reference" denotes [2] and "uncompressed" the original point cloud data.
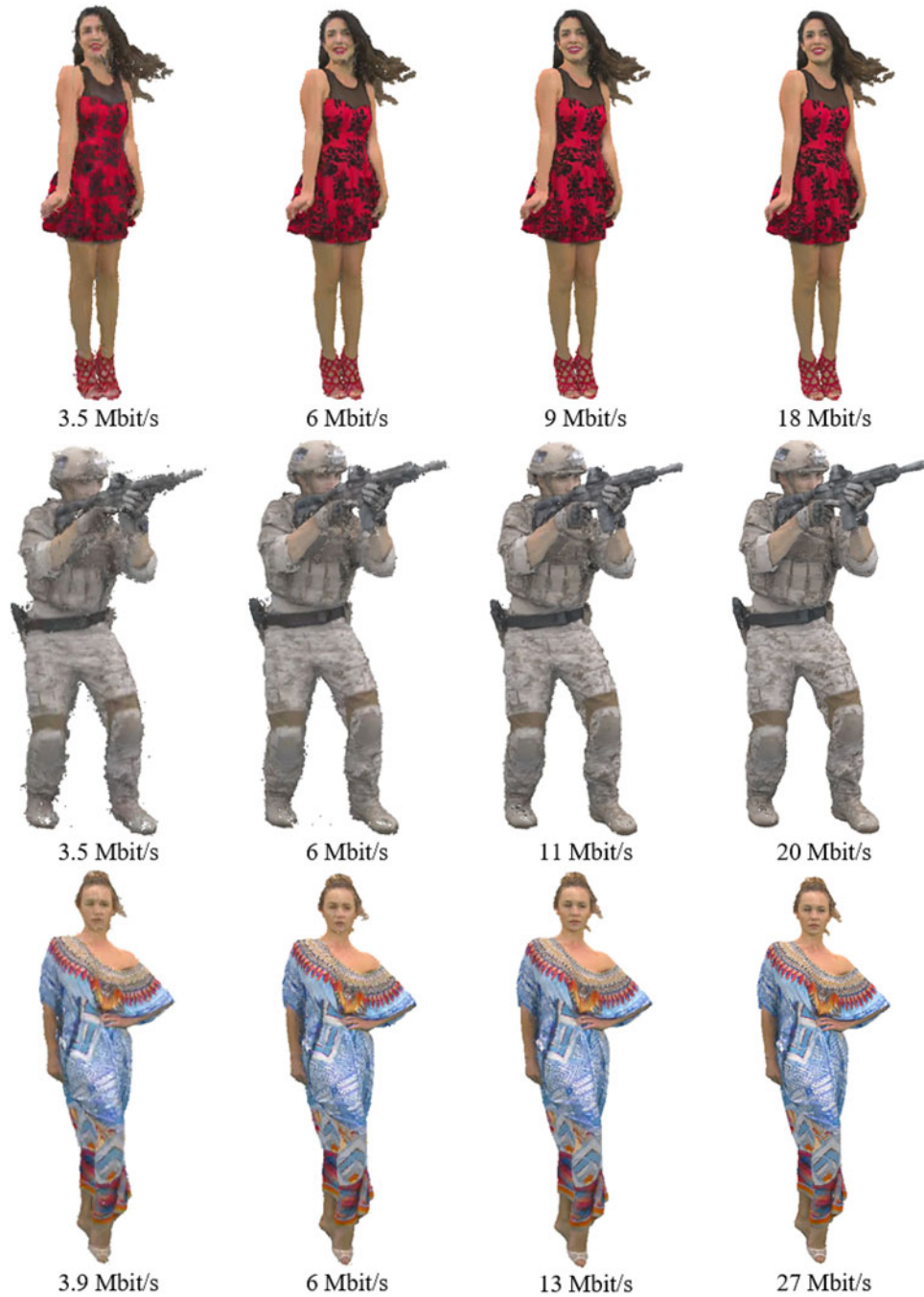
Fig. 10. Example views rendered from all bit streams submitted for subjective evaluation. Top to bottom: *RedAndBlack*, *Soldier*, and *Longdress*. Left to right: Rate R01, R02, R03, and R04.

Table 3. Objective quality without occlusion handling at low bit rates.

	Random access		
Metric	D1	D2	YUV
BD-BR [25]	−75.3%	**−85.4%**	−44%
BD-PSNR	7.1 dB	**6.9 dB**	1.9 dB

Table 4. Average coding run times in relation to anchor.

	All intra		Random access	
Condition	enc.	dec.	enc.	dec.
R05	101×	12.7×	685×	13.4×
R03	192×	112×	1531×	128.4×

implementation, two layers with roughly FullHD resolution are coded. For each layer a maximum of five reference frames is allowed for the random access configuration. One layer is 8 Bit with YUV420 chroma subsampling, one layer 10 Bit luma only. Thus, the total reference picture buffer size requirement per frame is roughly 32 MB. There is no need to buffer more than one input (encoder) or one output (decoder) point cloud, as temporal prediction is performed
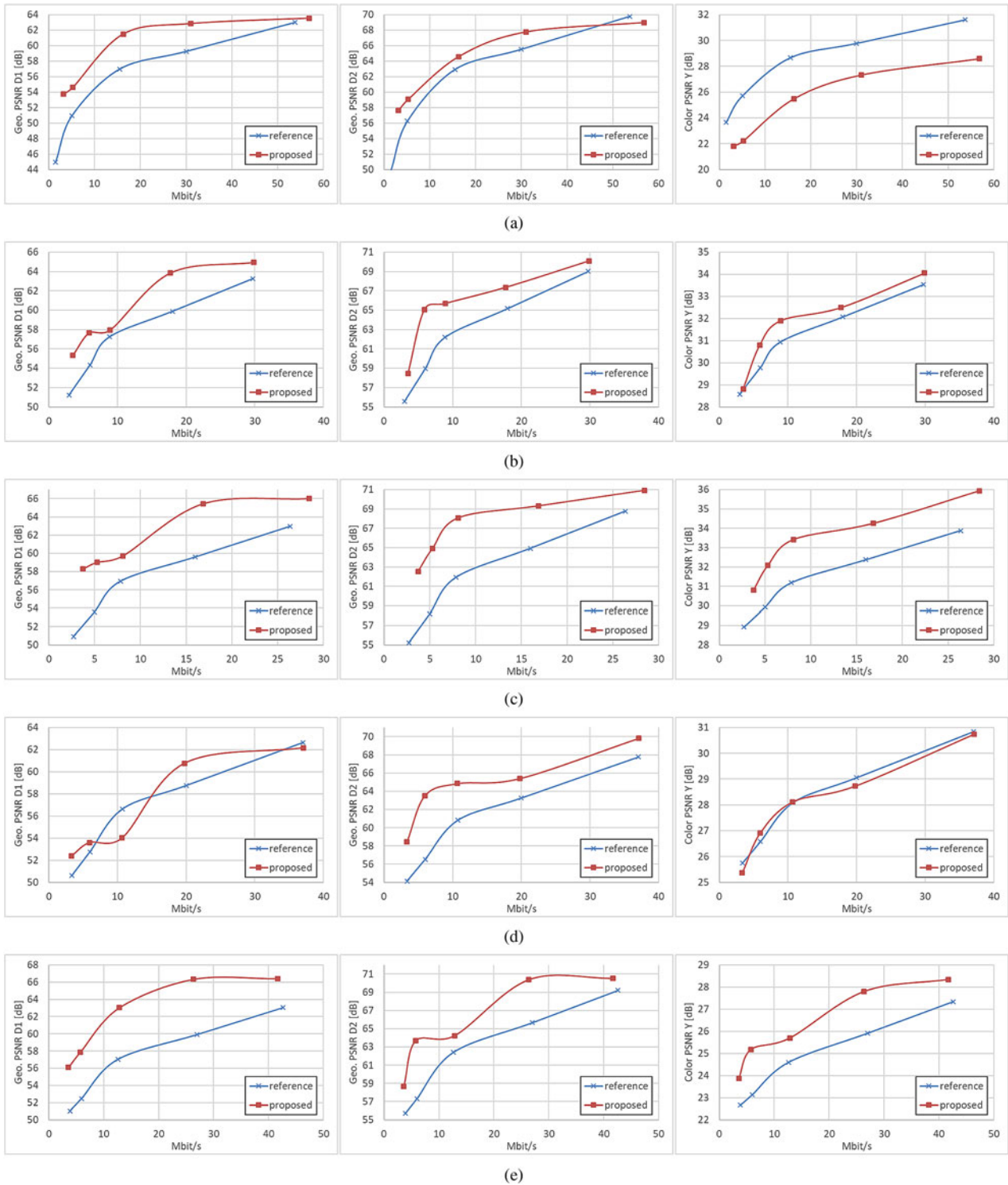
Fig. 11. Objective rate-distortion curves for all test sequences. (a) Objective rate-distortion curves for test sequence *Queen*, (b) objective rate-distortion curves for test sequence *RedAndBlack*, (c) objective rate-distortion curves for test sequence *Loot*, (d) objective rate-distortion curves for test sequence *Soldier*, and (e) Objective rate-distortion curves for test sequence *Longdress*.

on the 2D projection planes.

$$Texture : 1920 \times 1080 \times 2 \times 8Bit \times 5 \approx 20\,MB$$

$$Geometry : 1920 \times 1080 \times 1 \times 10Bit \times 5 \approx 12\,MB$$

Finally, the current implementation does not consider progressive decoding. However, such features could be easily

Table 5. Average coding run times in relation to anchor (all intra).

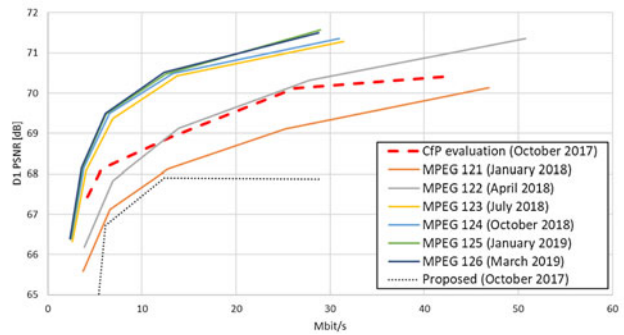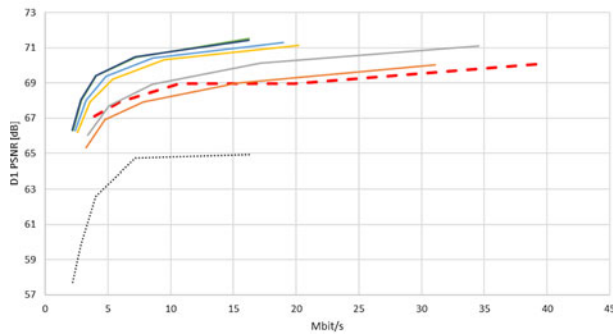| Condition	Proposed		CfP winner	
	enc.	dec.	enc.	dec.
R05	101×	12.7×	225×	12.6×
R03	192×	112×	577×	122.2×

Fig. 12. Progress of V-PCC coding performance since CfP evaluation.

implemented. With SHM as a code base, it is possible to select lower resolution texture and geometry planes as base layers, and one or more increasing resolutions as enhancement layers. Scalable decoding and reprojection from the selected resolution would provide progressive outputs. This functionality could also be extended for spatial random access, e.g. view-port dependent decoding.

4.0.8. Comparison to winning CfP and current stage of V-PCC standardization

As already shown in Figure 9, the winning CfP proposal [21] clearly outperformed the technology presented in this manuscript. However, in terms of complexity there was a clear benefit to the proposed solution. The surface normal calculations required for the 2D patch generation is significantly more complex than the simple geometry-projection proposal. Then again, this complexity only affects the encoder, as shown in Table 5.

Since the CfP evaluation V-PCC has been steadily improved and is now close to finalization [23]. To illustrate this process, Figure 12 summarizes the development in terms of coding performance based on D1 geometry distortion. The proposed technology has been added as well as reference.

5. CONCLUSION

This paper proposes a novel, projection-based approach to volumetric video compression: 3D scenes and objects are projected onto 2D images and compressed using standard 2D video technology. Key benefits of the proposed solution are improved coding performance compared to reference technology and the reliance on readily available 2D video coding solutions and infrastructure. The proposal was evaluated as part of the MPEG CfP on PCC, and objective and subjective quality evaluations proof its feasibility as a volumetric video coding solution. Although the proposed technology was not selected as winning proposal, the key concept, namely utilizing existing video coding technology to perform most of the heavy encoding operations, was also used in the winning proposal. Since then V-PCC has been steadily improved and is now close to being released as an international standard. More details on the MPEG point

cloud activity is summarized in [28]. The latest encoder description and standard specifiaction text are available in [21] and [23].

STATEMENT OF INTEREST

None.

REFERENCES

1 Kammerl J.; Blodow N.; Rusu R.B.; Gedikli S.; Beetz M.; Steinbach E.: Real-time compression of point cloud streams, in *2012 IEEE International Conference on Robotics and Automation*, May 2012, 778–785.

2 Mekuria R.; Blom K.; Cesar P.: Design, implementation, and evaluation of a point cloud codec for tele-immersive video. *IEEE Trans. Circuits. Syst. Video. Technol.*, **27** (4) (April 2017), 828–842.

3 Mamou K. *et al.*: The new MPEG-4/FAMC standard for animated 3D mesh compression, in *3DTV Conference*, 2008.

4 Google draco - 3D data compression. https://github.com/google/draco

5 Merkle P.; Smolic A.; Muller K.; Wiegand T.: Multi-view video plus depth representation and coding, in *IEEE Int. Conf. on Image Processing (ICIP)*, 2007.

6 Call for proposals for point cloud compression V2, in *ISO/IEC JTC1/SC29/WG11 N16763*, Apr 2017.

7 Alliez P.; Desbrun M.: Progressive compression for lossless transmission of triangle meshes, in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '01. New York, NY: ACM, 2001, pp. 195–202. [Online]. http://doi.acm.org/10.1145/383259.383281.

8 Peng J.; Kuo C.C.J.: Progressive geometry encoder using octree-based space partitioning, in *2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763)*, Vol. 1, June 2004, pp. 1–4.

9 Huang Y.; Peng J.; Kuo C.-C.J.; Gopi M.: Octree-based progressive geometry coding of point clouds, in *Proceedings of the 3rd Eurographics/IEEE VGTC Conference on Point-Based Graphics*, ser. SPBG'06. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2006, pp. 103–110. [Online]. http://dx.doi.org/10.2312/SPBG/SPBG06/103-110.

10 Schnabel R.; Klein R.: Octree-based point-cloud compression, in *Proceedings of the 3rd Eurographics/IEEE VGTC Conference on Point-Based Graphics*, ser. SPBG'06. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2006, pp. 111–121. [Online]. http://dx.doi.org/10.2312/SPBG/SPBG06/111-120.

11 Zhang C.; Florêncio D.; Loop C.: Point cloud attribute compression with graph transform, in *2014 IEEE International Conference on Image Processing (ICIP)*, Oct 2014, pp. 2066–2070.

12 Cui L.; Xu H. y.; Jang E.S.: Hybrid color attribute compression for point cloud data, in *2017 IEEE International Conference on Multimedia and Expo (ICME)*, July 2017, pp. 1273–1278.

13 Wang L.; Wang L.; Luo Y.; Liu M.: Point-cloud compression using data independent method - a 3D discrete cosine transform approach, in *2017 IEEE International Conference on Information and Automation (ICIA)*, July 2017, pp. 1–6.

14 Thanou D.; Chou P.A.; Frossard P.: Graph-based compression of dynamic 3d point cloud sequences. *IEEE Trans. Image. Process.*, **25** (4) (April 2016), 1765–1778.

15 de Queiroz R.L.; Chou P.A.: Motion-compensated compression of dynamic voxelized point clouds. *IEEE Trans. Image. Process.*, **26** (8) (August 2017), 3886–3895.

16 Gao Y.; Cheung G.; Maugey T.; Frossard P.; Liang J.: 3d geometry representation using multiview coding of image tiles, in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 6157–6161.

17 d'Eon E.; Harrison B.; Myers T.; Chou P.A.: 8i voxelized full bodies – a voxelized point cloud dataset, in *ISO/IEC JTC1/SC29/WG11 M40059*, Jan 2017.

18 Sheikhipour N.; Schwarz S.; Vadakital V.M.; Gabbouj M.: Efficient 2D video coding of volumetric video data, in *7th European Workshop on Visual Information Processing*, 2018.

19 HEVC scalability extension (SHVC), https://hevc.hhi.fraunhofer.de/shvc.

20 Sevom V.F.; Schwarz S.; Gabbouj M.: Geometry-guided 3D data interpolation for projection-based point cloud coding, in *7th European Workshop on Visual Information Processing*, 2018.

21 Pcc test model category 2, in *ISO/IEC JTC1/SC29/WG11 N17248*, Oct 2017.

22 V-pcc codec description, in *ISO/IEC JTC1/SC29/WG11 N18673*, Aug 2019.

23 Text of iso/iec dis 23090-5 video-based point cloud compression, in *ISO/IEC JTC1/SC29/WG11 N18670*, Aug 2019.

24 Tian D.; Ochimizu H.; Feng C.; Cohen R.; Vetro A.: Geometric distortion metrics for point cloud compression, in *IEEE Int. Conf. on Image Processing*, 2017.

25 Tourapis A.M.; Singer D.; Su Y.; Mammou K.: BD-Rate/BD-PSNR excel extensions, in *ISO/IEC JTC1/SC29/WG11 M41482*, Oct 2017.

26 Baroncini V.; Cesar P.; Siahaan E.; Reimat I.; Subramanyam S.: Report of the formal subjective assessment test of the submission received in response to the call for proposals for point cloud compression, in *ISO/IEC JTC1/SC29/WG11 M41786*, Oct 2017.

27 Turing codec - H.265/HEVC software video encoder and decoder. http://turingcodec.org/.

28 Schwarz S. *et al.*: Emerging mpeg standards for point cloud compression. *IEEE J. Em. Sel. Top. Circuits Syst.*, **9** (1) (March 2019), 133–148.

Sebastian Schwarz received the degrees of Dipl.-Ing. in Media Technology from the Technical University of Ilmenau, Germany, in 2009, and Dr. Tech. from Mid Sweden University, Sweden, in 2014. Dr. Schwarz is currently Research Leader of the Volumetric Video Coding team at Nokia Technologies in Tampere, Finland. Before joining Nokia, he held a Marie Skodowska-Curie fellowship as Experienced Researcher with BBC R&D in London. Dr. Schwarz has authored over 20 conference and journal papers on immersive media and video coding topics. He is currently an editor of the ISO/IEC committee draft for video-based point cloud compression (ISO/IEC 23090-5) and co-chair of the MPEG Ad hoc group on System Technologies for V-PCC.

Nahid Sheikhipour received her M.Sc. degree in Information Technology from the Tampere University of Technology (TUT), Tampere, Finland, in 2018. In 2017, she has joined Nokia Technologies as Master student on Point Cloud Compression (PCC). Followed by working as an External Researcher on standardization of PCC. Currently she is a Research Intern with Nokia Technologies in Volumetric Video Coding domain and pursing her Ph.D. Her research interests include computer graphics, image and video compression, virtual reality, and augmented reality.

Vida Fakour Sevom received her B.Sc. in Biomedical Engineering majoring Bioelectric at Islamic Azad University of Mashhad, 2011. She received her M.Sc. degree in bioengineering from Tampere University of Technology (TUT), Tampere, Finland, in 2015. Currently, she is working as a software engineer at AAC Technologies. At the same time, she is pursuing her Ph.D. degree with the Department of Signal Processing, TUT. Her main research interests include computer vision, imaging, and image signal processing.

Miska M. Hannuksela received his Master of Science and Doctor of Science degrees from Tampere University of Technology, Finland, in 1997 and 2010, respectively. He is currently Nokia Bell Labs Fellow and Head of Video Research in Nokia Technologies. He has been with Nokia since 1996 in different roles including research manager/leader positions in the areas of video and image compression, end-to-end multimedia systems, as well as sensor signal processing and context extraction. He has published more than 170 conference and journal papers and hundreds of standardization contributions. He is or has been an editor in several video compression and media systems standards, including H.264/AVC, H.265/HEVC, High Efficiency Image File Format (HEIF), ISO Base Media File Format, and Omnidirectional Media Format. His current research interests include video compression and immersive multimedia systems. Dr. Hannuksela received the award of the Best Doctoral Thesis of the Tampere University of Technology in 2009. He has co-authored several papers awarded in international conferences. He was an Associate Editor of IEEE Transactions on Circuits and Systems of Video Technology from 2010 to 2015.