IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received April 3, 2020, accepted April 15, 2020, date of publication April 30, 2020, date of current version May 15, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2991478

3D Motion Estimation and Compensation Method for Video-Based Point Cloud Compression

JUNSIK KIM, JIHEON IM, SUNGRYEUL RHYU, AND KYUHEON KIM
Department of Electronic Engineering, Kyung Hee University, Yongin 17104, South Korea

Corresponding author: Kyuheon Kim (kyuheonkim@khu.ac.kr)

This work was supported in part by the Korea Electric Power Corporation (KEPCO) under Grant R18XA02, and in part by the Institute of Information and communications Technology Planning and evaluation (IITP) under Grant 2020-0-00452.

ABSTRACT A point cloud visualizes information by placing a voxel with a color value and a position value in a three-dimensional space. Since a point cloud uses hundreds of thousands or millions of points to visualize information, a large number of bits is needed compared to existing 2D media. Therefore, it is essential to compress point data for transmission and storage. The Moving Picture Expert Group (MPEG) is developing a point cloud compression method based on 2D video that takes advantage of the benefits of coding efficiency and the wide adaption of video codecs by various industries. This compression method is called video-based point cloud compression (V-PCC). Generally, video codecs use a compression method that employs a block matching algorithm. Currently, V-PCC is conducted using 2D video codecs, which means that motion information used by V-PCC is obtained from 2D video sequences. Thus, this 2D-based motion information limits the characterization of the motion in terms of 3D-points, which is also disadvantageous to compression efficiency. In this paper, we propose a method for estimating and compensating the motion in terms of a 3D object when compressing a dynamic object point cloud using a conventional video codec. The proposed 3D motion estimation and compensation technology showed higher gain overall in terms of BD-rate and was proven to effectively compress 3D point cloud content on the basis of 3D motion.

INDEX TERMS Video-based point cloud compression, 3D motion search, 3D motion estimation, compression of vector information.

I. INTRODUCTION

Conventional 2D images represent objects and scenes as a set of pixels with color values. Like a 2D image, a point cloud represents objects and scenes as a set of voxels with color values in terms of the 3D domain. In other words, a point cloud is a medium that visualizes information by placing a voxel with a color value and a position value in a three-dimensional space. However, unlike in a 2D image, in a point cloud, all of the expression space of the voxels in the 3D point cloud domain is not filled with the position and color information. Point cloud content is seen as the next generation of media in the fields of virtual reality (VR) [1], augmented reality (AR) [2], and autonomous driving [3]. Since a point cloud uses hundreds of thousands or millions of points to visualize information, a large number of bits is needed compared to existing 2D media [4]. Therefore, it is essential to compress point data for transmission and storage [5], [6], [7].

In the Moving Picture Expert Group (MPEG) under ISO/IEC JTC1, an international standardization organization, the standardization of point cloud compression is under discussion. In MPEG, a dynamic point cloud of moving objects is called a dynamic objects point cloud and is considered to be a video with a point cloud. MPEG has started to develop a compression method for dynamic point clouds that uses an existing video codec [8]. All existing video codecs were developed based on 2D images. Therefore, in MPEG, a point cloud is projected in 2D space to create a 2D image, and 2D video is compressed using an existing video codec such as Advanced Video Coding (AVC)/H.264 [9] or High Efficiency Video Coding (HEVC) [10]. This compression method is known as video-based point cloud compression (V-PCC) [11].

In V-PCC, the 2D video codec basically uses a motion-search-based compression method such as a block matching algorithm [12]. This requires that the motion search is conducted only in a projected 2D domain generated in a direction orthogonal to the 3D points. However, this 2D-based motion search method is not suitable for the 3D motion that

The associate editor coordinating the review of this manuscript and approving it for publication was Yun Zhang.

occurs in a point cloud video sequence. Thus, it is necessary to develop a 3D motion estimation and compensation mechanism for better compression efficiency. To solve this 3D motion search, Li *et al.* [13] proposed a method of transferring 3D motion to a 2D video codec and obtained better compression efficiency. This approach shows high compression efficiency, but has limitations in that the 2D video codec needs to be modified, which means to develop 3D motion estimation methods suited for each 2D video codec. To effectively apply the V-PCC technology to various 2D video codec, it is necessary to develop 3D motion estimation method regardless of a 2D video codec to be used.

In this paper, we propose a method for estimating and compensating the motion in terms of a 3D object when compressing a dynamic object point cloud using a conventional video codec. This paper is structured as follows. Chapter II describes the V-PCC technology, which proceeds as developed by MPEG. A technology for achieving motion estimation and compensation in terms of 3D objects is proposed in Chapter III. In Chapter IV, we compare the results obtained using the proposed technology and existing technologies developed by MPEG. Finally, further work being considered for better compression efficiency is suggested in Chapter V.

II. VIDEO-BASED POINT CLOUD COMPRESSION

With the development of computer graphics and image processing technologies, attention is being focused on point cloud technology that expresses real space and object information as 3D content [14]. A point cloud can visualize information by positioning a point having a color value in a three-dimensional space [15]. Point cloud technology is being actively researched in a variety of fields, including immersive media such as AR and VR and autonomous vehicles. In particular, because of the increased computing power now available, the 3D graphics processing field has also begun to use point cloud data. This trend is accelerating with the development of point cloud acquisition devices such as Kinect.

As the number of point cloud applications increase and acquisition methods become more common, there is growing need for point cloud compression. As a result, the international organization for standardization known as MPEG has begun the development of technology for compressing point cloud content in which point cloud content is classified into three categories: static, dynamic, and dynamically acquired [16]. Static data and dynamically acquired data include characteristics that represent a specific point of time in an environment or object, and dynamic data include characteristics that represent moving objects. Fig. 1 shows an example of dynamic point cloud data.

FIGURE 1. Example of dynamic point cloud data.

Since V-PCC is designed in terms of a 2D video compression scheme, the 3D point cloud must be projected into 2D space for use by existing video codecs such as AVC/H.264 and Fig. 2 and 3 show the architecture of the current V-PCC encoder and decoder, respectively. As explained previously, V-PCC requires that 3D points be projected into a 2D space for use by 2D video codecs. The patch generation function shown in Fig. 3 produces 2D patches from 3D points with the most similarities among points along the normal direction [17], then the patch packing function generates the patch location information [18]. On the basis of this patch location information, texture and geometry images are generated for the color and 3D location information of the 3D points, respectively [19]. Since 3D points can be duplicated during the projection into the 2D domain, two geometry and two texture images are used [20], one based on the minimum location value and the other based on the difference between the minimum and maximum location value (called the thickness image). To improve 2D video compression performance, high frequency in the geometry and texture images must be reduced. Thus, the image padding function in Fig. 2 is used to fill up the empty space in the 2D texture and geometry images [21], [22]. The original locations of the patches to be deleted by this function are determined by the occupancy map image, which consists of 0s and 1s indicating whether or not there are points packed in each pixel. Information related to the 3D-to-2D projection, such as projection plane information and the size of each patch, is separately compressed in the auxiliary patch information compression function. Finally, the padded geometry and texture video sequences are compressed by existing video codecs [23].

The decoding of the V-PCC is shown in Fig. 3, which is the reverse of the encoding process explained in Fig. 2.

As explained previously, V-PCC is currently conducted using 2D video codecs, which means that motion information used in the V-PCC is obtained from 2D padded video sequences. Thus, this 2D-based motion information limits the characterization of the motion in terms of 3D points, which is disadvantageous to compression efficiency.

III. 3D MOTION ESTIMATION AND COMPENSATION

As explained in the previous chapter, V-PCC performs a compression of dynamic point cloud content using a 2D video codec which allows for a highly reliable and economical compression method. Thus, V-PCC has excellent performance and has proven to be a suitable architecture for the compression of dynamic point cloud data.

However, currently, V-PCC projects 3D point cloud content into a 2D video sequence. A 2D video codec, which applies a 2D motion search and compensation mechanism, is used for compression. Thus, there exist limitations when 3D motion information is used, which can improve the

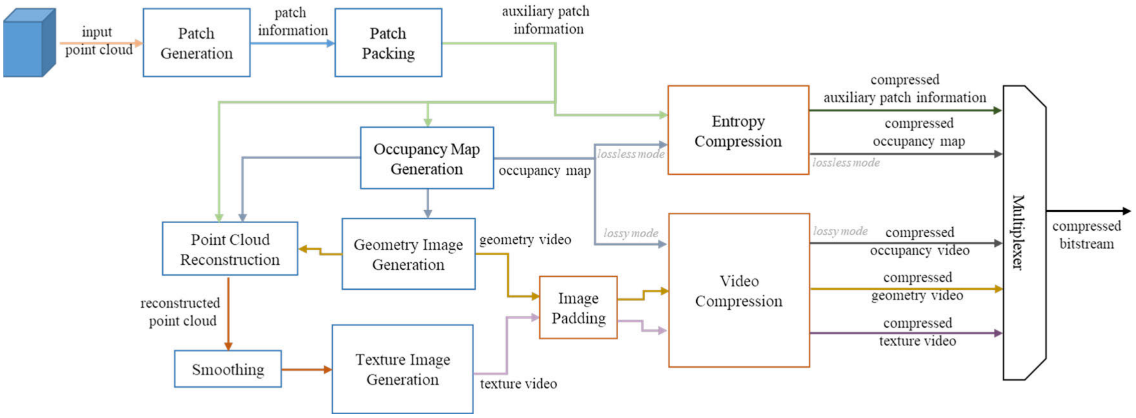

FIGURE 2. Architecture of the V-PCC encoder.

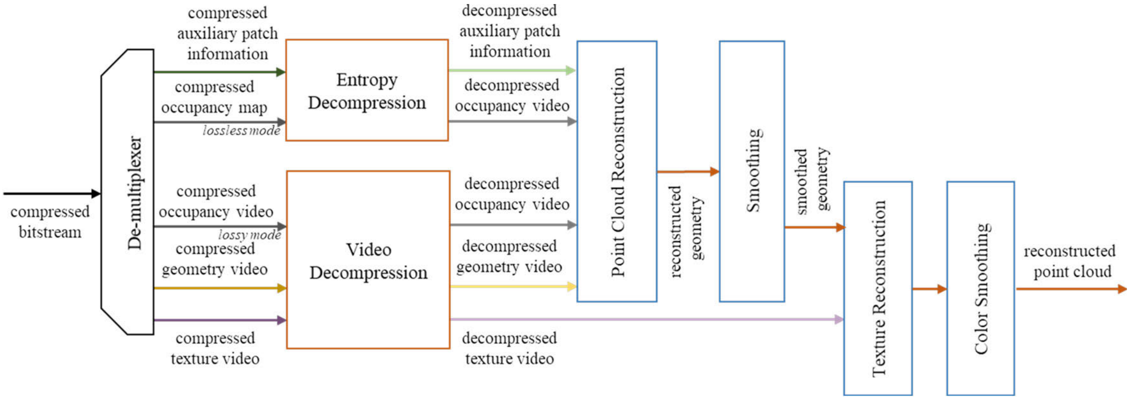
compressed auxiliary patch information
compressed occupancy map
lossless mode
Entropy Decompression
decompressed auxiliary patch information
decompressed occupancy video
compressed bitstream
De-multiplexer
compressed occupancy video
lossy mode
compressed geometry video
Video Decompression
decompressed occupancy video
decompressed geometry video
compressed texture video
decompressed texture video
Point Cloud Reconstruction
reconstructed geometry
Smoothing
smoothed geometry
Texture Reconstruction
Color Smoothing
reconstructed point cloud

FIGURE 3. Architecture of the V-PCC decoder.

compression benefits. This paper proposes an algorithm for point cloud compression that uses 3D motion estimation and compensation.

A. ARCHITECTURE

The 3D motion estimation and compensation algorithm proposed in this paper is based on the V-PCC architecture shown in Fig. 4, where the functions of 3D motion search, vector image generation, and delta image generation are additionally provided.

The current V-PCC algorithm in [11] produces the same number of geometry, texture images, and occupancy maps with all the input point cloud frames. However, the 3D-motion-based encoder proposed in this paper selects a point cloud frame as an intra-frame, and the subsequent point cloud frames are considered to be predicted frames. The intra-frame is compressed according to the current V-PCC encoding procedure, and the predicted frames are compressed using 3D motion vector and delta images from the intra-frame.

The 3D-motion-compensation-based decoder is illustrated in Fig. 5, which shows an intra-frame being reconstructed by the current V-PCC decoding procedure and a predicted frame being reconstructed by combing the 3D motion vector and delta image with the reconstructed intra-frame.

This chapter will explain how to select an intra-frame and a predicted frame, and describe how to generate a 3D motion vector, a vector image, and a delta image in terms of 3D point clouds.

B. INTRA AND PREDICTED FRAMES

To use the proposed 3D motion estimation and compensation algorithm, an intra-frame and a predicted frame are determined as follows:

$$f_{(t+\Delta t)} = \begin{cases} \mathit{Predicted}, \sum_{1}^{N} S_n\left(f_{(t)}, f_{(t+\Delta t)}\right) < \Omega \\ \mathit{Intra}, \sum_{1}^{N} S_n\left(f_{(t)}, f_{(t+\Delta t)}\right) \geq \Omega \end{cases} \tag{1}$$

where $f_{(t)}$ represents a 3D point cloud frame, and $S_n()$ represents a motion search function for N points in an intra-frame. The details of the motion search function will be explained in Section C. As explained in (1), the summation of the motion search function outputs is less than a certain value Ω, which means a frame $f_{(t)}$ and a subsequent frame $f_{(t+\Delta t)}$ have a similar point distribution, and thus the subsequent frame is defined as a predicted frame. An intra-frame is defined for a subsequent frame, which has less similarity with frame $f_{(t)}$.

An example of the processing procedure for determining an intra and a predicted frame is shown in Fig. 6. The red arrow

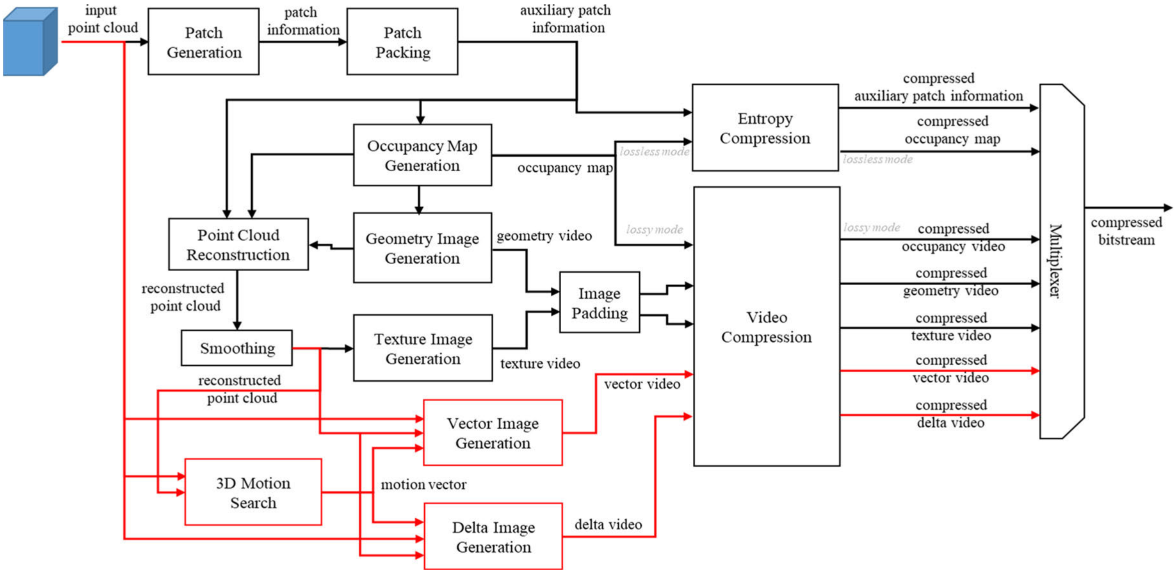

FIGURE 4. Architecture of the proposed 3D motion estimation technology.

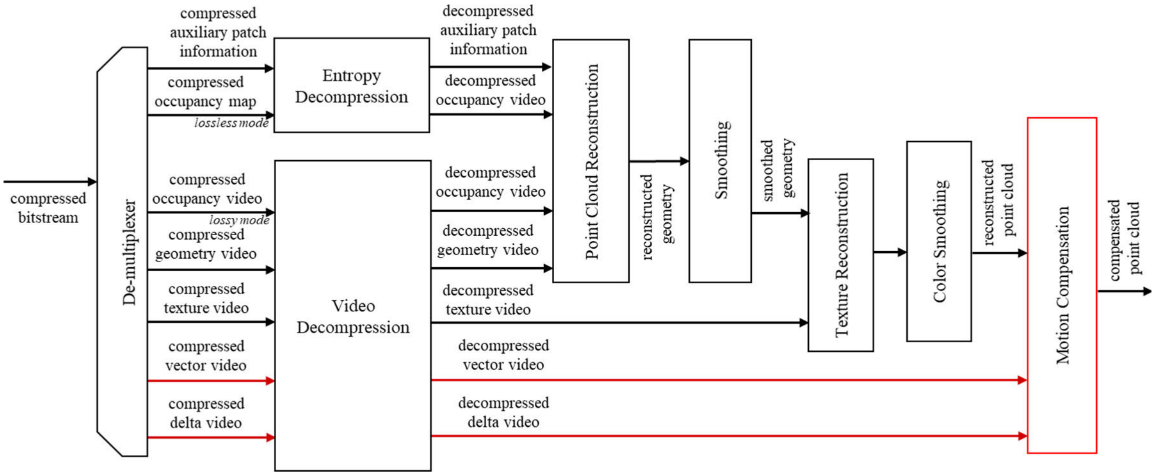

FIGURE 5. Architecture of the proposed 3D motion compensation technology.

indicates the example in which Ω is exceeded, and the blue arrow indicates an example in which it is not. If the result of motion estimation does not exceed Ω, the frame is determined to be a predicted frame, otherwise, it is determined to be an intra-frame. If the current frame is determined to be an intra-frame, the following frames are searched on the basis of the newly-determined intra-frame. As shown in Fig. 6, the intra-frames are reconstructed as a geometry image and a texture image, as is done in the existing V-PCC scheme. The predicted frame is reconstructed as a vector image and delta image using the proposed 3D motion search and estimation scheme.

C. MOTION SEARCH BASED ON THE NEAREST NUMBER OF POINTS

As described in the previous section, the determination of whether an input point cloud frame is to be an intra or predicted frame is made on the basis of a 3D motion search. The 3D motion search proposed in this paper is conducted by finding the most similar points within a specific 3D search range, where the similarity and 3D search range are obtained in terms of a texture image sequence and a geometry image sequence, respectively. However, it is impossible to perform a 3D motion search using a macro cube such as the macro block used in a 2D image motion search. This is because the interior of a point cloud dataset is not completely filled with voxels, unlike the interior of a 2D search region in a 2D image for 2D motion estimation. Thus, this paper proposes a motion search method based on the nearest number of points, called 2NoP. This method finds similar points in reference to a group of points rather than to a single point.

An example of motion search based on 2NoP is shown in Fig 7. One point within the intra-frame ($f_{(t)}$), such as a point p_n in Fig. 7, is chosen as a reference point. A corresponding point within the subsequent frame ($f_{(t+\Delta t)}$) is selected as a target point, such as a point p'_n in Fig. 7. However, the target

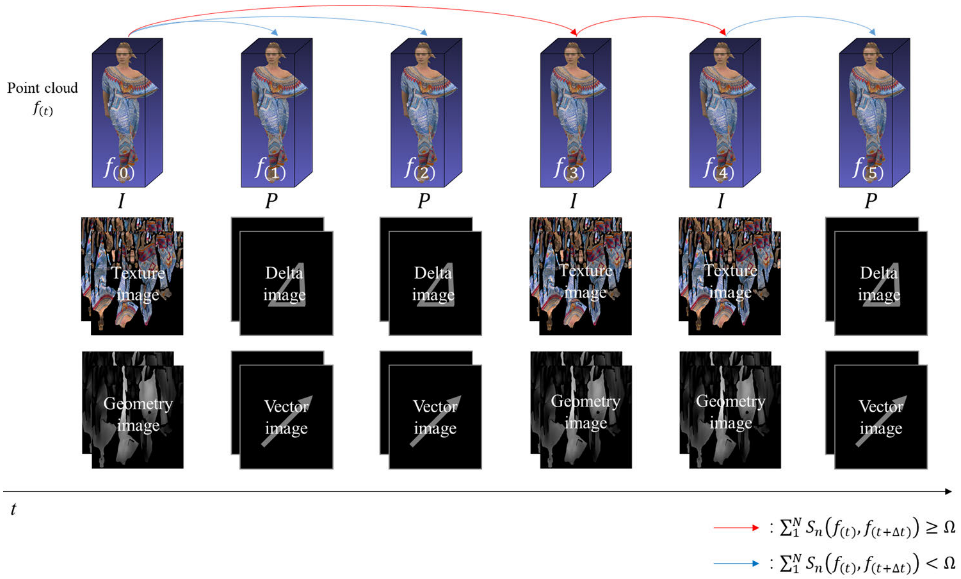

FIGURE 6. **Illustration of predicted frame determination method.**

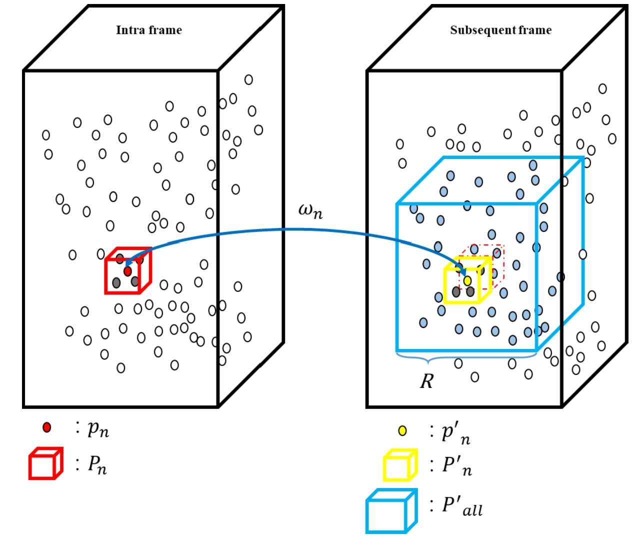

FIGURE 7. **Illustration of the motion search method based on 2NoP.**

point p'_n is not located in the same 3D coordinate position, and thus the target point p'_n should be defined for a motion search. Thus, this method uses points in a specific region R as candidate target points, such as the blue points in Fig. 7. Since the proposed 2NoP search method uses a group of points, the nearest k points around the reference and the target point, such as the points in the red and yellow boxes in Fig. 7, are selected for motion search. The details of the proposed 2NoP motion search are as follows:

$$p_n = [x, y, z, r, g, b], \quad (p_n \in f_{(t)}) \tag{2}$$

$$P_n = \{p_1, p_2, \ldots, p_k\}, \quad (n(P_n) = k,\ p_n \in P_n,\ P_n \subset f_{(t)}) \tag{3}$$

where p_n is a point in the intra-frame with its 3D position and color information, and which is spatially centered among the points of P_n. P_n, a group of k points being compared in the 2NoP motion search. The k points in P_n can be selected using various nearest neighbor searches [24]–[26] or a closest point search [27]. Similarly, p'_n is a point in the subsequent frame and is located at the center of P'_n. All points in P'_n are to be compared to all points in P_n.

$$\begin{aligned} p'_n &= [x+d_x, y+d_y, z+d_z, r, g, b], \\ &\quad (-R \leq d_x, d_y, d_z \leq R) \end{aligned} \tag{4}$$

$$P'_{all} = \{p'_n | p'_n \in f_{(t+\Delta t)}\} \tag{5}$$

$$\begin{aligned} P'_n &= \{p'_1, p'_2, \ldots, p'_k\}, \\ (n(P'_n) &= k, p'_n \in P'_n, P'_n \subset f_{(t+\Delta t)}) \end{aligned} \tag{6}$$

where p'_n is a point within a subsequent frame which belongs to a specific range R. Then, all points in a range R are defined as P'_{all}. Further, P'_n is defined as the set of k points closest to p'_n as in P_n. As described before, the 2NoP motion search compares P_n with P'_n in terms of a group of points rather than a single point. Each P'_n is to be defined according to a candidate target point in P'_{all}. The similarity ω_n between P_n and P'_n can be obtained by (7) as follows:

$$\omega_n = \sum_{j=1}^{J} \left(\begin{array}{c} \left(r_{p_j} - r_{p'_j}\right)^2 + \left(g_{p_j} - g_{p'_j}\right)^2 \\ + \left(b_{p_j} - b_{p'_j}\right)^2 \end{array} \right) \Big/ 3K, \quad \left\{p_j \in P_n, p'_j \in P'_n\right\} \tag{7}$$

As described in (7), the similarity is obtained using the difference in color information between neighboring points in a range in an intra-frame and a subsequent frame. Thus, the similarity in color and geometry information can be determined using (7). Finally, the points p_n and p'_n producing the smallest value of ω_n are considered to be reference and target points for the motion vector. If the minimum value of ω_n is not small enough, the motion search of p_n is considered failed,

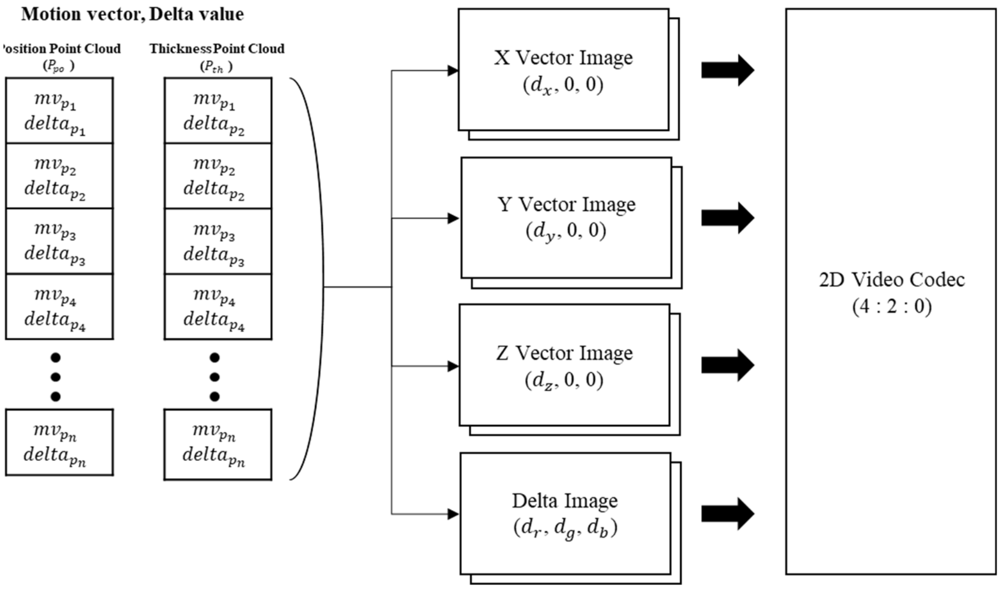

FIGURE 8. Vector image and delta image compression process.

in which case the output of $S_n()$ in (1) is 0. Conversely, if the motion search is successful, the output of $S_n()$ is 1. Thus, intra frame and predicted frame are determined by the number of points of the successful motion search.

The motion vector mv_{p_n} is obtained as follows:

$$mv_{p_n} = \left[d_x, d_y, d_z\right] = \left[x_{p_n}, y_{p_n}, z_{p_n}\right] - \left[x_{p'_n}, y_{p'_n}, z_{p'_n}\right] \quad (8)$$

The result of the 2NoP motion search proposed in this paper is a three-dimensional vector, as shown in (8). Since every point in an intra-frame will have a motion vector, it is necessary to develop a method of efficiently compressing these motion vectors, which will be discussed in Section D.

D. VECTOR AND DELTA IMAGE GENERATION AND COMPRESSION

This section describes a method for compressing the 3D motion vector obtained by the 2NoP motion search proposed in Section C. This method is based on a 2D video codec. Since the motion between an intra and a subsequent frame is continuous, neighboring 3D motion vectors will be similar, and thus a conventional 2D video codec can perform efficient compression while maintaining the similarity of input values. Additionally, the parameters used to convert a 3D value to a 2D value in V-PCC can be used to transform a 3D motion vector to 2D images. This can also improve compression.

The motion vector described in (8) can accurately move a point in an intra-frame to a point in a subsequent frame, but the color values cannot be compensated. To solve this problem, this paper proposes a method that uses the delta value. The delta value is used to compensate for the difference in color values between an original frame and a decompressed frame, and can be expressed by (9) as follows:

$$delta_{p_n} = \left[r_{p_n}, g_{p_n}, b_{p_n}\right] - \left[r_{p'_n}, g_{p'_n}, b_{p'_n}\right] \quad (9)$$

where, $delta_{p_n}$ indicates a difference in color value between a point p_n in an intra-frame and point p'_n in a subsequent frame. As described previously in Section C, motion vectors are created at every point in an intra-frame and thus, each point in an intra-frame has a corresponding motion vector that is realized in terms of 3D position values such as x, y, and z. Thus, this paper proposes using the texture image generation method of V-PCC to generate 2D vector images from the 3D motion vectors, such as mv_{p_n} described in (8).

To use a conventional 2D video codec, this paper proposes assigning each element of $delta_{p_n}$, such as d_r, d_g, and d_b, to YUV values in an image and generating delta images (Fig. 8). However, the elements of mv_{p_n} are independent of each other, and thus it is not suitable to generate a 2D vector image by assigning d_r, d_g, and d_b to the YUV domain as is done for $delta_{p_n}$. Hence, this paper proposes using three vector images corresponding to d_x, d_y, and d_z, as shown in Fig. 8.

Additionally, an intra-frame is divided into two point clouds: a position point cloud P_{po} and a thickness point cloud P_{th} [28]. Then the 3D motion vectors and delta values should be obtained in terms of those position and thickness point clouds as indicated in Fig. 8. Finally, the generated vector and delta video sequences are compressed using a conventional 2D video codec with 4:2:0 subsampling, where the video codec configuration (such as QP for vector and delta video sequences) is set to be the same as the geometry and texture video sequence configuration for the V-PCC. The method for 3D motion compensation based on this proposed 3D motion estimation is discussed in Section E.

E. MOTION COMPENSATION

As described in Section A, compressed vector and delta video sequences are decoded using a conventional 2D video

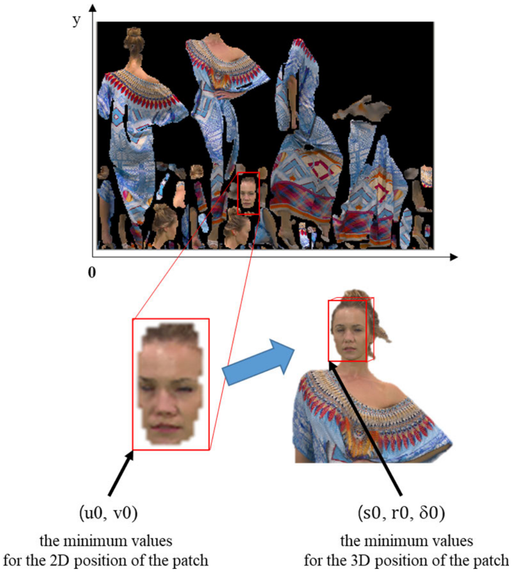
y
0
(u0, v0)
the minimum values
for the 2D position of the patch
(s0, r0, δ0)
the minimum values
for the 3D position of the patch

FIGURE 9. Example of the minimum values for the 3D and 2D position of the patch.

decoder, where the decoded vector and video sequences are used to reconstruct a point cloud of subsequent frames based on decoded intra-video sequences, such as texture and geometry video sequences. The 3D motion compensation method proposed in this paper is defined as follows:

$$p_n^* = [x, y, z, r, g, b], (p_n^* \in f_{(t+\Delta t)}^*) \tag{10}$$

where p_n^* is a reconstructed point within a decoded point cloud with subsequent frame $f_{(t+\Delta t)}^*$, and is described by its position and color information, such as x, y, z, r, g, and b. The position information for p_n^* is generated by combining the decoded geometry images of an intra-frame and the decoded vector images of a subsequent frame, and is obtained using equations (11), (12), (13), and (14):

$$x_{p_n^*} = \mathrm{s0} + u - \mathrm{u0} + Y_{V_x^*(u,v)} \tag{11}$$
$$y_{p_n^*} = \mathrm{r0} + v - \mathrm{v0} + Y_{V_y^*(u,v)} \tag{12}$$
$$z_{p_n^*} = \delta 0 + Y_{G_{po}(u,v)} + Y_{V0_z^*(u,v)} \tag{13}$$
$$z_{p_n^*} = \delta 0 + Y_{G_{po}(u,v)} + Y_{G_{th}(u,v)} + Y_{V1_z^*(u,v)} \tag{14}$$

where s0, r0, and $\delta 0$ are the minimum values for the 3D position of the patch, and u0 and v0 are the minimum values for the 2D position of the patch (as shown in Fig. 9). These are described in the auxiliary patch information that is created in the patch generation step described in Chapter II [11]. The values $x_{p_n^*}$, $y_{p_n^*}$, and $z_{p_n^*}$ are the x, y, and z position values of reconstructed point p_n^*.

Since a 2D patch is generated on the basis of a vector normal to the 3D point cloud, the normal vectors can be defined by the X, Y, or Z direction. Thus, it is necessary to use a vector oriented in the direction normal to a patch in the decoded 2D frame when reconstructing 3D point clouds. Further, a decoded 2D frame is recognized in terms of a vertical and a horizontal axis, such as U and V, respectively. The position information of p_n^* defined in equations (11) to (14) shows the case in which Z is used as the directional normal vector, where u and v represent the coordinate value in a 2D decoded frame. Since the 3D motion estimation proposed in Section D uses only Y values for vector images, $Y_{V_x^*(u,v)}$, $Y_{V_y^*(u,v)}$, and $Y_{V_z^*(u,v)}$ represent Y values in the YUV components of the decoded vector images $V_x^*(u, v)$, $V_y^*(u, v)$, and $V_z^*(u, v)$, respectively. Equations (11) and (12) can provide the position information of a reconstructed 3D point cloud in terms of the X and Y axes, respectively. Since the Z axis is used for the normal vector direction, depth information is used for the Z axis position information of the reconstructed point cloud [11]. The Z axis position information can be found using (13) and (14), where $G_{po}(u, v)$ and $G_{th}(u, v)$ are the decoded position and the decoded thickness image for an intra-frame [28], and decoded vector images corresponding to $G_{po}(u, v)$ and $G_{th}(u, v)$ are defined as $V0_z^*(u, v)$ and $V1_z^*(u, v)$, respectively. When a thickness image is not being used, (13) is used to obtain the Z axis position information. Otherwise, (14) is used.

The color value of reconstructed point cloud p_n^* is obtained using (15) and (16) as follows:

$$[r_{p_n^*}, g_{p_n^*}, b_{p_n^*}] = T0(u, v) + \Delta 0^*(u, v) \tag{15}$$
$$[r_{p_n^*}, g_{p_n^*}, b_{p_n^*}] = T1(u, v) + \Delta 1^*(u, v) \tag{16}$$

where $T0^*(u, v)$ and $T1^*(u, v)$ are the decoded texture images of an intra-frame, and $\Delta 0^*(u, v)$ and $\Delta 1^*(u, v)$ are delta images corresponding to each texture image, such as $T0^*(u, v)$ and $T1^*(u, v)$. Since texture images are generated on the basis of two geometry images as explained in Chapter II, the color information for a reconstructed point cloud is found in terms of the generated texture images, as in (15) and (16). Equations (15) and (16) are used to find the color information of a reconstructed point cloud based on position and thickness images, respectively.

Chapter III describes the process of compressing and reconstructing a point cloud using the proposed 3D motion estimation and compensation technology. The results of a comparison of this method to V-PCC of MPEG are shown in the next chapter.

IV. EXPERIMENTAL RESULTS

The performance of the 3D motion estimation and compensation technology proposed in this paper was determined using V-PCC reference software v4 [29]. HEVC was used as the 2D video codec, and the test point cloud sequences called "Soldier", "Queen", "Longdress", "Red and Black" and "Loot" were used under Common Test Conditions (CTC) in V-PCC [30]. In the CTC of V-PCC, the test sequences are encoded with quantization parameter (QP) values that vary from R1 to R5, as shown in Table 1.

The CTC uses the Bjontegaard-Delta-rate (BD-rate) as a quality verification method, which derives a quantitative value by combining the results of various qualities obtained

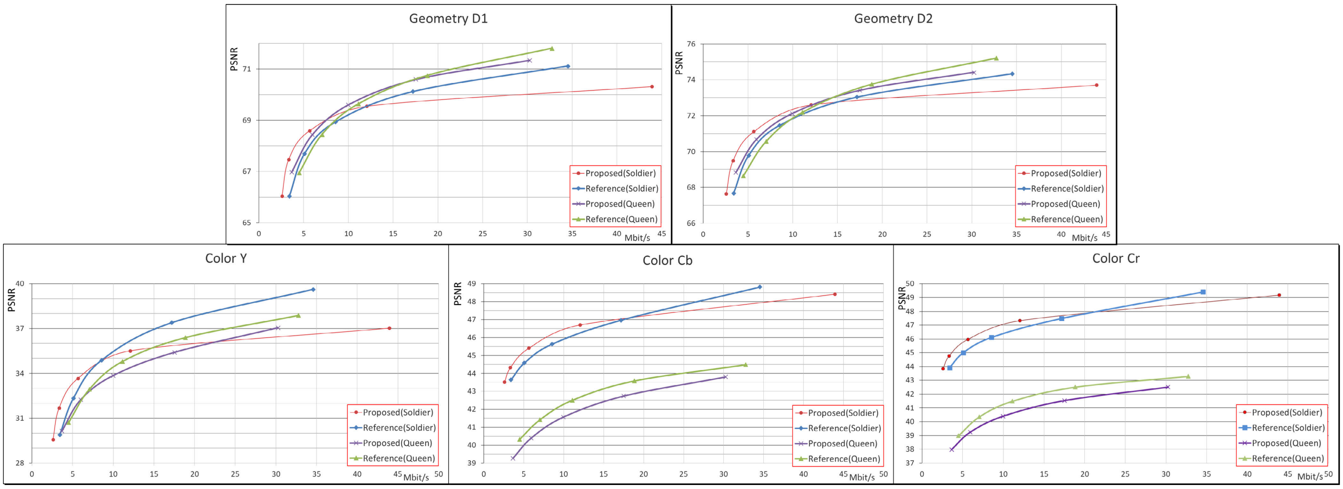

FIGURE 10. BD-rate results of "Soldier" and "Queen" data in D1, D2, Y, Cb, and Cr.

TABLE 1. Quantization parameters of the common test condition.

R1 to R5	Geometry Video	Texture Video	Vector Video	Delta Video
R01	32	42	32	42
R02	28	37	28	37
R03	24	32	24	32
R04	20	27	20	27
R05	16	22	16	22

from individual QP values using the peak signal-to-noise ratio (PSNR) and a bitrate [31]. A conventional BD-rate for a 2D video sequence is expressed in terms of color information. However, the BD-rate for a 3D point cloud should show coding efficiency not only in terms of color information, but also in terms of geometry information. This is because a 3D point cloud is realized in terms of color and 3D position values.

The PSNR of the geometry information can be obtained using point-to-point and point-to-plane distances, where the point-to-point distance is determined by the Euclidean distance between a reference point and a nearest point, and the point-to-plane distance is determined by estimating the distance between the reference point and a projected point along a normal direction. The point-to-point distance PSNR is denoted by D1, and the point-to-plane distance PSNR by D2. The color information PSNR are expressed in terms of Luma, Cb, and Cr. The coding efficiency achieved by the proposed 3D motion estimation and compensation technology was determined by comparison with the current V-PCC reference software v4 [32].

Fig. 10 shows the PSNR results of the proposed technology compared to the V-PCC reference software v4. As shown in Fig. 10, the proposed technology achieved a higher geometry PSNR value than the V-PCC reference for bit rates of up to about 15 Mbps, which indicates that the proposed technology is more suitable for a lower bitrate environment, such as mobile wireless 4G and LTE environments [33].

The proposed technology has been also applied to "Longdress", "Red and Black" and "Loot" of the test sequences, however, there is no gains obtained. This is because the 3D motion in those sequences have relatively larger between frames, and thus, cannot be found in the 3D motion search region proposed in this paper. Thus, it is needed to have further research on search region and 3D motion search algorithm.

As described earlier, the CTC uses the BD-rate as a quality verification method, and thus the proposed technology must be compared in terms of the BD-rate. Table 2 compares the "Soldier" data results of the proposed 3D motion estimation and compensation method with the V-PCC reference software v4 in terms of the BD-rate. The proposed technology has a better compressed bitstream size, as shown in the bitstream size column of Table 2. However, it produces slightly lower PSNR values, as described in Fig. 10. Since the BD-rate considers bitstream size and PSNR, these values are included in Table 2. It was confirmed that the proposed technology produces an overall higher BD-rate gain of up to 19.9%, which means that the gain in bitstream size is larger than the loss in PSNR. Therefore, the proposed 3D motion estimation and compensation technology obtains better compression gains overall than the current V-PCC reference software v4. However, it is observed that the PSNR between the V-PCC reference and the proposed are crossed at higher bitrate as displayed in Fig. 10. This is because the bitstream size from the proposed is larger than one from the V-PCC reference at R5 as being shown in Table 2. Since R5 bitstream uses lower quantization parameter, it causes the vector and delta images to have more higher frequency components compared to other bitstreams, which resulted in less efficient 2D video compression.

Compared to the V-PCC reference the proposed technology increases additional computational complexity caused by

TABLE 2. Bitstream size, PSNR, and BD-Rate results of proposed technology and V-PCC reference ("solider" data).

	Variant	Bitstream size	PSNR				
		(Byte)	D1 (dB)	D2 (dB)	Luma (dB)	Cb (dB)	Cr (dB)
Proposed technology	R01	3,226,698	66.03	67.64	29.55	43.52	43.84
V-PCC reference		4,250,427	66.03	67.68	29.88	43.64	43.91
Proposed technology	R02	4,166,592	67.46	69.48	31.67	44.33	44.77
V-PCC reference		6,349,019	67.69	69.78	32.34	44.58	45.00
Proposed technology	R03	7,075,895	68.58	71.11	33.66	45.41	45.97
V-PCC reference		10,665,074	68.93	71.46	34.88	45.63	46.10
Proposed technology	R04	15,054,879	69.54	72.60	35.50	46.70	47.32
V-PCC reference		21,460,193	70.12	73.04	37.39	46.97	47.48
Proposed technology	R05	54,919,202	70.31	73.71	37.02	48.41	49.17
V-PCC reference		43,177,752	71.12	74.35	39.62	48.81	49.41
BD-rate			D1 (%)	D2 (%)	Luma (%)	Cb (%)	Cr (%)
			-14.8%	-19.1%	-2.2%	-16.2%	-19.9%

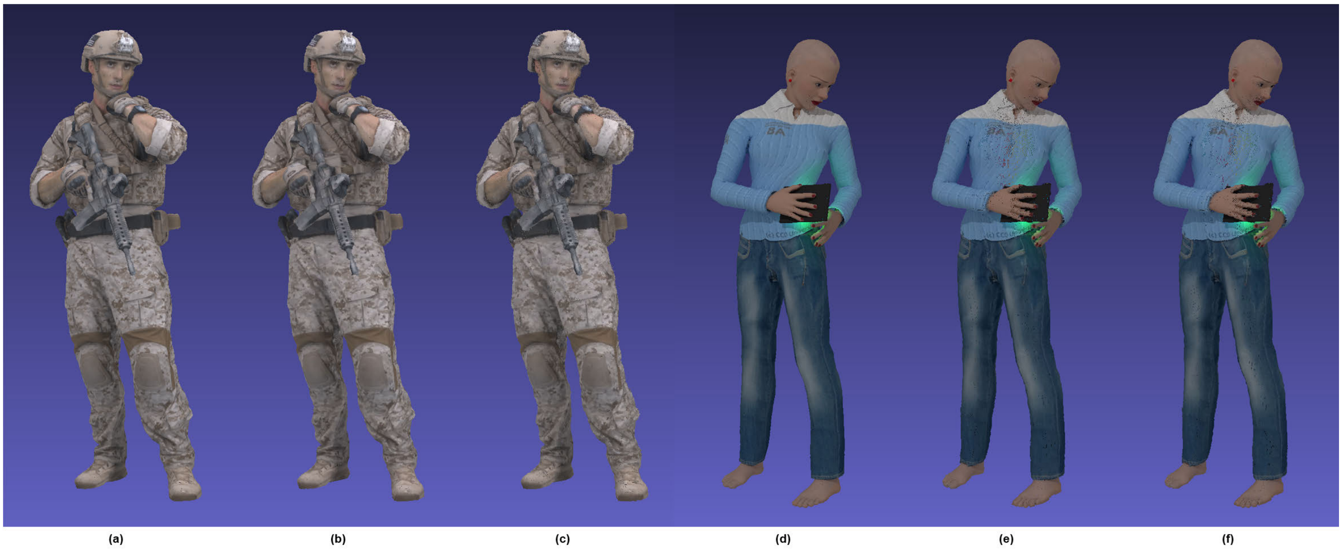

FIGURE 11. a) Uncompressed "Soldier" data, (b) "Soldier" data result of the V-PCC reference, (c) "Soldier" data result of the proposed technology, d) Uncompressed "Queen" data, (e) "Queen" data result of the V-PCC reference, (f) "Queen" data result of the proposed technology.

both the nearest algorithm used for 3D motion search, and encoding three vector images instead of a geometry image.

The results are shown visually in Fig. 11, where (a) and (d) is the uncompressed "Soldier" and "Queen" data, (b) and (e) show the "Soldier" and "Queen" R05 decompressed data using the V-PCC reference, (c) and (f) show the "Soldier" and "Queen" R05 decompressed data using the proposed technology, respectively. As shown in Fig. 11, there is no significant degradation in the proposed technology result.

These experimental results demonstrate that the proposed 3D motion estimation and compensation technology more efficiently compresses 3D point cloud content without significant degradation in quality, and also needs to further studied for higher bitrate.

V. CONCLUSION

Three-dimensional point cloud content requires more storage space than a conventional 2D image because millions of data points are needed to represent the 3D point content. Thus, it is essential to efficiently compress this point cloud content. In accordance with this requirement, the international standardization organization MEPG developed V-PCC, which compresses dynamic point cloud content using a conventional 2D video codec. However, since this compression scheme is based on a 2D video codec, the 3D motion of point clouds cannot be effectively used for compression. Thus, this paper proposes a 3D motion estimation and compensation technology suitable for use with the V-PCC architecture.

The 3D motion estimation and compensation scheme proposed in this paper is as follows. First, an intra-frame and a predicted frame are distinguished in the input point cloud sequence. Intra-frames are converted into geometry and texture video in the same manner as in V-PCC. The predicted frames are converted into vector video and delta video using the 3D motion vector obtained as a result of the 2NoP search discussed in this paper. The 2NoP search is used to perform a 3D motion search that is suitable for a point cloud sequence because it takes into account the fact that a 3D domain in a point cloud frame is not fully represented by point voxels. The 3D vector information generated from the 2NoP search is converted into a vector image to be compressed using the V-PCC architecture, and a delta image is used to compensate the color difference. Thus, the geometry and texture images of the predicted frame are replaced with vector and delta images. In particular, each vector has independent values for each axis, and thus this paper proposes a method for generating three vector video sequences corresponding to the x, y, and z axes. Each video sequence is compressed and decompressed using a conventional 2D video codec, and decompressed video sequences are combined to reconstruct a 3D point cloud. The proposed 3D motion estimation and compensation scheme produces better compression efficiency by using 3D motion in a point cloud sequence rather than 2D motion with a conventional 2D video codec.

As shown in this paper, the proposed 3D motion estimation and compensation technology achieved higher gain overall in terms of BD-rate, and effectively compressed 3D point cloud content on the basis of 3D motion. However, the proposed technology is limited in higher bitrate ranges such as R05, as shown in Table 2. In the future, our work will improve the efficiency at higher bitrates by investigating a motion search method that is more suitable for the structure of the motion estimation and compensation technology proposed in this paper.

REFERENCES

[1] Y. Ishikawa, R. Hachiuma, N. Ienaga, W. Kuno, Y. Sugiura, and H. Saito, "Semantic segmentation of 3D point cloud to virtually manipulate real living space," in *Proc. 12th Asia Pacific Workshop Mixed Augmented Reality (APMAR)*, Ikoma, Nara, Mar. 2019, pp. 1–7.

[2] K. Ma, F. Lu, and X. Chen, "Robust planar surface extraction from noisy and semi-dense 3D point cloud for augmented reality," in *Proc. Int. Conf. Virtual Reality Vis. (ICVRV)*, Hangzhou, China, Sep. 2016, pp. 453–458.

[3] P. Sun, X. Zhao, Z. Xu, R. Wang, and H. Min, "A 3D LiDAR data-based dedicated road boundary detection algorithm for autonomous vehicles," *IEEE Access*, vol. 7, pp. 29623–29638, 2019.

[4] *Call for Proposals for Point Cloud Compression V2*, document ISO/IEC JTC1/SC29/WG11 MPEG2017/N16763, Hobart, TAS, Australia, Apr. 2017.

[5] R. L. de Queiroz and P. A. Chou, "Compression of 3D point clouds using a region-adaptive hierarchical transform," *IEEE Trans. Image Process.*, vol. 25, no. 8, pp. 3947–3956, Aug. 2016.

[6] R. L. de Queiroz and P. A. Chou, "Motion-compensated compression of dynamic voxelized point clouds," *IEEE Trans. Image Process.*, vol. 26, no. 8, pp. 3886–3895, Aug. 2017.

[7] P. de Oliveira Rente, C. Brites, J. Ascenso, and F. Pereira, "Graph-based static 3D point clouds geometry coding," *IEEE Trans. Multimedia*, vol. 21, no. 2, pp. 284–299, Feb. 2019.

[8] *Text of ISO/IEC CD 23090-5 Video-Based Point Cloud Compression,* document ISO/IEC JTC1/SC29/WG11 MPEG2018/N18030, Macau, China, Oct. 2018.

[9] *Advanced Video Coding for Generic Audio-Visual Services*, document ISO/IEC 14496-10, May 2003.

[10] *High Efficiency Video Coding*, document ISO/IEC 23008-2, Jan. 2013.

[11] *V-PCC Codec Description*, document ISO/IEC/JTC1/SC29/WG11 MPEG2018/N18017, Macau, China, Oct. 2018.

[12] A. Hussain, L. Knight, D. Al-Jumeily, P. Fergus, and H. Hamdan, "Block matching algorithms for motion estimation—A comparison study," *Int. J. Sci. Eng. Res.*, vol. 264, no. 2, pp. 356–369, 2014.

[13] L. Li, Z. Li, V. Zakharchenko, J. Chen, and H. Li, "Advanced 3D motion prediction for video-based dynamic point cloud compression," *IEEE Trans. Image Process.*, vol. 29, pp. 289–302, 2020.

[14] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, and J. Wan, "An integrated framework for 3-D modeling, object detection, and pose estimation from point-clouds," *IEEE Trans. Instrum. Meas.*, vol. 64, no. 3, pp. 683–693, Mar. 2015.

[15] J. He, Z. Fu, W. Hu, and Z. Guo, "Point cloud attribute inpainting in graph spectral domain," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Taipei, Taiwan, Sep. 2019, pp. 4385–4389.

[16] *Draft Test Conditions and Complementary Test Material*, document ISO/IEC JTC1/SC29/WG11/MPEG2014/N16716, Geneva, Switzerland, Jan. 2019.

[17] H. Hoppe, T. DeRose, T. Duchamp, J. A. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," in *Proc. SIGGRAPH*, 1992, pp. 71–78.

[18] S. Sebastian, M. Miska Hannuksela, F. Vida, and S.-P. Nahid, "2D video coding of volumetric video data," in *Proc. Picture Coding Symp. (PCS)*, 2018, pp. 61–65.

[19] S. Schwarz, M. Preda, V. Baroncini, M. Budagavi, P. Cesar, P. A. Chou, R. A. Cohen, M. Krivokuca, S. Lasserre, Z. Li, J. Llach, K. Mammou, R. Mekuria, O. Nakagami, E. Siahaan, A. Tabatabai, A. M. Tourapis, and V. Zakharchenko, "Emerging MPEG standards for point cloud compression," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 9, no. 1, pp. 133–148, Mar. 2019.

[20] E. S. Jang, M. Preda, K. Mammou, A. M. Tourapis, J. Kim, D. B. Graziosi, S. Rhyu, and M. Budagavi, "Video-based point-cloud-compression standard in MPEG: From evidence collection to committee draft [Standards in a Nutshell]," *IEEE Signal Process. Mag.*, vol. 36, no. 3, pp. 118–123, May 2019.

[21] K. Mammou, J. Kim, V. Valentin, F. Robinet, A. Tourapis, and Y. Su, *CE2.12 Related: Sparse Linear Model Based Padding Method for the Texture Images*, document ISO/IEC JTC1/SC29/WG11 m44837, Macau, CH, Oct. 2018.

[22] D. Graziosi and A. Tabatabai, *[V-PCC] New Contribution on Geometry Padding*, document ISO/IEC JTC1/SC29/WG11 m47496, Geneva, China, Mar. 2019.

[23] *High Efficiency Video Coding Test Model*. Accessed: 2019. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.18+SCM-8.7/

[24] S. A. Nene and S. K. Nayar, "A simple algorithm for nearest neighbor search in high dimensions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 9, pp. 989–1003, Sep. 1997.

[25] H. Samet, "K-nearest neighbor finding using MaxNearestDist," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 243–252, Feb. 2008.

[26] K. C. K. Lee, B. Zheng, and W.-C. Lee, "Ranked reverse nearest neighbor search," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 7, pp. 894–910, Jul. 2008.

[27] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Trans. Inf. Theory*, vol. 48, no. 8, pp. 2201–2214, Aug. 2002.

[28] J. Kim, J. Im, S. Rhyu, and K. Kim, "Point cloud compression on the basis 3D motion estimation and compensation," *Proc. SPIE*, vol. 11137, Sep. 2019, Art. no. 1113719.

[29] *V-PCC Test Model V4*, document ISO/IEC JTC1/SC29/WG11 MPEG2018/N17996, Macau, China, Oct. 2018.

[30] *Common Test Conditions for Point Cloud Compression*, document ISO/IEC JTC1/SC29/WG11 N17995, Macau, China, Oct. 2018.

[31] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," in *Proc. 13th VCEG-M33 Meeting*, Austin, TX, USA, Apr. 2001, pp. 2–4.

[32] *PCC TMC2 Performance Evaluation and Anchor Results*, document ISO/IEC JTC1/SC29/WG11 N17998, Macau, China, Oct. 2018.

[33] *4G LTE Speeds vs. Your Home Network*. Accessed: 2013. [Online]. Available: https://www.verizonwireless.com/articles/4g-lte-speeds-vs-your-home-network/