Contextual Homogeneity-Based Patch Decomposition Method for Higher Point Cloud Compression

SUNGYEUL RHYU, JUNSIK KIM, JIHEON IM, AND KYUHEON KIM
Department of Electronics Engineering, Kyung Hee University, Yongin 17104, South Korea
Corresponding author: Kyuheon Kim (kyuheonkim@khu.ac.kr)

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP), Korea Government (MSIT) (Development of Adaptive Viewer centric Point cloud AR/VR Streaming Platform) under Grant 2020-0-00452.

ABSTRACT
Point cloud content is widely used to store and represent 3D volumetric objects with a complex and detailed representation from any direction of view. However, the amount of data needed for point cloud content is much larger than that of 2D representations. To overcome this difficulty, MPEG has started to develop a Video-based Point Cloud Compression (V-PCC) that is designed by projecting point cloud content into 2D content and compressing the 2D content using conventional 2D video codecs. Compression efficiency of the V-PCC can be achieved when 3D motion flow and textual conformity on 3D surfaces are preserved through 2D projections that are favorable to the 2D video codec. As mentioned above, point cloud content has a complex geometry, therefore when decomposing a point from 3D coordinates to construct a 2D patch, several situations must be considered in addition to the location of adjacent points. This paper addresses the issues in such complex geometry by proposing a method that preserves 3D homogeneity in 2D patches. Comprehensive experiments are conducted to demonstrate bitrate savings of 0.5%, 0.6%, 7.8%, 7.0% and 5.5% in random access mode and 0.1%, 0.0%, 7.0%, 4.2% and 3.3% in all intra mode for D1, D2, Y, Cb, and Cr, respectively, compared to the reference software.

INDEX TERMS
Video-based point cloud compression, contextual homogeneity, patch generation.

I. INTRODUCTION
The provision of immersive experiences is considered one of the most important requirements for multimedia content. To provide an immersive experience, VR and AR content has been widely adopted in the entertainment industry, providing not only 2D but 3D experiences [1], [2]. In particular, 3D experiences present a more realistic experience and a higher degree of freedom in the consumption of multimedia content [3]. Point cloud content is considered as one of the representative multimedia content formats that can provide this 3D experience at 6 degrees of freedom [4].

To provide realistic details, point cloud content consists of numerous points, each of which is realized in terms of the position in a 3D domain (such as through X, Y, Z coordinates) and in terms of other attributes such as color information (as in R, G, B values). For example, a common point cloud sequence used for the Call for Proposals of Point Cloud Compression in MPEG [5] has generally 800,000-1,500,000 points in each frame. Each point requires 10 bits for each of the X, Y, and Z geometry information and 8 bits for the R, G, and B texture information; however, the bandwidth of such a point cloud sequence would require up to 3 Gbps, which is not realistic for a currently deployed network environments [6]. Therefore, it is necessary to develop point cloud content compression technology for a wider range of applications and industry deployments.

II. MPEG VIDEO-BASED POINT CLOUD COMPRESSION (V-PCC) STANDARD
A. BASIC ARCHITECTURE
Compression technology for point cloud content is under standardization in MPEG. The MPEG project for coded representations of immersive media is MPEG-I, and has two types of point cloud compression: in video-based and in geometry-based approaches. The idea behind
geometry-based approaches is to break the point cloud into blocks of voxels and then encode them in octree, as proposed by Queiroz and Chou [7]. This approach has been enhanced further, for example, by exploiting different sizes of macroblocks in octree voxel space for real-time tele-immersion as proposed by Mekuria et al. [8], arithmetically encoding hierarchical sub-band transforms using Laplace or Gaussian transforms as proposed by Queiroz et al. [9], [10], and extending KNN to generate efficient graphs for sparsely-populated blocks [11]. Video-based point cloud compression (V-PCC) [12] projects point cloud content into a 2D domain and compresses the 2D projected images using conventional 2D video codecs such as MPEG-4 AVC [13] and MPEG-H HEVC [14]. Though there were other studies on estimating temporal motion in geometry-based approaches [7], [15], [16], the compression efficiency of V-PCC for dynamic point cloud content is considered to outperform the G-PCC.

Dynamic point cloud content used in V-PCC typically consists of sequences of point cloud frames. Each point cloud frame is a time instance of a dynamic point cloud content with 800,000-1,500,000 points. Fig. 1 shows the overall architecture of an encoder for the reference software of a V-PCC test model for category 2, called TMC2 [17], where a point cloud sequence is taken as input and a compressed V-PCC bitstream is generated as an output.

![FIGURE 1. TMC2 encoder architecture.](image)

As the first step, 3D points in the sequence are decomposed into patches, which are obtained by estimating a normal vector at each point from the relationship of its adjacent points [18], then grouping neighboring points with similar directions to one of the six faces of a bounding box. An individual patch generated from 3D points is orthogonally projected onto a 2D domain, and the projected patches are packed into a 2D packing image [19] for greater gain in 2D video compression by an HEVC software, HM [20]. Since a point is realized by both its texture information, such as color values, and its geometry information, such as X, Y and Z coordinates, the projected 2D images are realized in terms of their texture and geometry aspects, respectively [21]. Motion flow in V-PCC can be analyzed in terms of both 3D point clouds and 2D packing video sequences. The motion in a 3D domain has been estimated in [22] and [23]. The 2D packing image has pixels occupied by the projected patches while the rest is unoccupied. The rest of the pixels are padded with information extrapolated from the patch boundaries for bitrate savings in terms of 2D video compression. The occupancy map is a bitmap of 1 or 0 that indicates whether each pixel is from a projected patch or from padding, respectively.

When 3D points are projected into a 2D image along a projection line, a pixel of the 2D packing image corresponds to one or more 3D points due to the fact that multiple 3D points can be placed on the same projection line. Thus, it is required to have a pair of layers for 2D textures and geometry packing images for all the individual points along a projection, which results in too many images representing points in point cloud content. To overcome this difficulty, TMC2 uses only two points along a projection line, which are projected into 2D packing images such as layer 0 and 1. For example, a layer 0 image comes from a 2D projection of the nearest 3D point, usually an outer 3D surface, and the layer 1 image comes from a 2D projection of the farthest 3D point, which is usually an inner 3D surface. In the process of reconstructing decompressed point cloud content, the intermediate space between layer 0 and 1 is filled to restore multiple 3D points and thicknesses, and then painted with the interpolated colors of the layer 0 and layer 1 points [24].

### B. PROBLEMS OF PROJECTION OF CONTEXTUALLY MIXED TEXTURE REGIONS

This paper points out two major problems of TMC2. First, 3D points in patches are projected onto a 2D domain and stored as two layers of images in terms of the texture and geometry. This method of projecting 3D points onto two layers packs the image to store the thickness by taking into account the distance of each point to the projection plane, which may cause drawbacks in optimizing the packing image process for 2D video compression. This showcases the problem of preserving only the closest and the farthest points from the projection plane in layers 0 and 1 of the 2D domain images, which cannot cover all the points in this thickness. After encoding and decoding a texture packing image, only the two points from either layer 0 or 1 have texture values among the reconstructed points, and the rest of the points have texture values by interpolating those two texture values. The larger the texture value difference is between layers 0 and 1, the larger the PSNR error will be. This is the reason that a patch should be characterized by not only its geometry but also its color information. Thus, we define a term, contextual region, for a group of points that are contiguous with neighboring points in terms of both geometric and contextual homogeneity. For example, in Fig. 2, the Queen sequence has multiple different contextual regions. Each point is projected to a projection plane, thus the cross section of points and normal projection lines can easily show the cascaded
plane. For these unprocessed points for projection in Fig. 3a, the grey colored area) to not be projected to the projection defined in TMC2 as indicated by the dotted line, which causes the #0 and #1 layer images of the #0 patch. There is a pre- and are mapped to pixels in each 2D patch image such as are projected onto a projection plane along a projection line, the outer contextual region and the inner contextual region such as the color of a point. In this decomposition, points from different contextual regions into one patch results in more complex 2D images with high-frequency components. Despite their differences in color from different contextual regions, points are processed only with regard to their geometric relevance.

Fig. 3a shows TMC2 operation on complex geometries. Note that the color of each line is for identification of a contextual region on 2D patches of 2D packing images, which results in less efficient 2D video coding performance. This paper is organized as follows. Chapter III analyzes the characteristics of patch generation from a point cloud sequence in TMC2. Based on this analysis, chapter IV proposes a method of higher video-based point cloud compression by contextual patch decomposition. Chapter V shows the compression efficiency improvements by providing the results of applying the proposed method to various point cloud sequences. Finally, chapter VI describes the benefits realized by the proposed method.

III. ANALYSIS ON 2D PATCH PACKING IMAGES IN TERMS OF LAYERS
As described in chapter II, the current patch generation method in TMC2 has the limitations of having a patch with multiple different contextual regions. In this case, layer 0 and layer 1 of a patch are composed of parts from surfaces of different contextual regions, and thus, comparing the layer images can reveal the association of its original contextual region. Compression efficiency will decrease as well because it is difficult for the 2D video encoder to efficiently perform motion estimation with scattered texture.

A patch in TMC2 is stored as layer 0 and layer 1. To identify the discontinuity of a contextual region, it is considered that a patch with more than two contextual regions has lower similarity in terms of contextual homogeneity. The five test data sequences, Long dress, Soldier, Loot, Red and Black, and Queen, provided by [5] are used for this analysis. Since layer 0 and layer 1 texture packing images in a patch are obtained along the same normal projection line, those two images from a single contextual region would have similar texture values, which would have lower variance in terms of the texture values. The distribution of sum of absolute differences is obtained using equations (1), (2) and (3). $D_{ij}$ is the absolute difference of luminance of a pixel at position $(i, j)$ in the layer 0 and layer 1 images. Each image has occupied and unoccupied pixels. $O_{ij}$ is either 1 or 0 depending on if the pixel at position $(i, j)$ is occupied or unoccupied, respectively. $C(x)$ is the number of pixels having the same luminance difference, $x$.

$$D_{ij} = |Y_{layer0}(i, j) - Y_{layer1}(i, j)|$$

$$O_{ij} = \text{Occupancy}(i, j)$$

$$C(x) = \sum_{i=0}^{\text{width}} \sum_{j=0}^{\text{height}} \delta(x - D_{ij}) \times O_{ij}$$

When most of the pixels have small differences, there is a higher probability that a single contextual region is represented by a patch. However, a larger number of pixels with larger differences may indicate that there are multiple

---

**Equations:**

1. $D_{ij} = |Y_{layer0}(i, j) - Y_{layer1}(i, j)|$
2. $O_{ij} = \text{Occupancy}(i, j)$
3. $C(x) = \sum_{i=0}^{\text{width}} \sum_{j=0}^{\text{height}} \delta(x - D_{ij}) \times O_{ij}$
contextual regions in a patch. Fig. 4 shows the distributions from the test data sequences. By analyzing the distribution of the number of differences in each sequence, we can see that the Queen sequence shown in Fig. 4e has a lot of values with particularly larger differences compared to the other sequences from Fig. 4a-4d. In other words, as mentioned earlier, this sequence can be considered to have many patches, each of which with multiple contextual regions.

To obtain the contextual region, we exploit the color distance in terms of luminance and chroma information. The CIELab color space [25] is defined for a uniform color space and color differences [26] based on the perceived magnitude of human color stimuli, and provides the Euclidean distance $\Delta E$, which can be used as a quantified distance between two different colors. The value of distance 1.0 implies imperceptible color difference [27], and the difference may become perceptible over 4.0 [28].

IV. CONTEXTUAL HOMOGENEITY-BASED PATCH DECOMPOSITION (CHD)

As explained in chapter II, the current patch generation method provides multiple patches, each of which has multiple contextual regions. Thus, this paper provides a new patch decomposition method based on contextual homogeneity, which prevents one patch from having more than one contextual region in terms of both color and geometry information.
This method helps a packed image with contextual patches to be more consistent and have a higher compression rate. As described in chapter III, two colors with a color distance value larger than 4.0 become perceptible as different colors to human eye. Additionally, a patch is composed of a group of points in a space, and thus a color distance value should be considered in terms of a patch rather than that of an individual point. This is because a smaller color distance value in terms of a point causes a lack of neighboring points for generating a patch. Thus, it needs to find a color distance value larger than 4.0 and composition of a patch as a group of points.

This paper proposes a novel method called the Contextual Homogeneity-based patch Decomposition (CHD), which decomposes patches using the distance in terms of both geometry and color information. The proposed CHD method has the following sequences to obtain contextual homogeneity in a patch. First, from the patch generated by geometry information, the color distance of each pair of points between both projected textual images of layer 0 and layer 1 is measured to measure the degree of discrepancy between those textual images. Next, the average of the color distance of all points in the patch is evaluated to determine whether this patch can be considered as homogeneous or not. When the evaluated average color distance is larger than a certain threshold, this patch is considered to have multiple contextual regions, and thus, needs to be decomposed.

For example, as shown in Fig. 5a, points in a patch are projected to layers 0 and 1 through the current patch generation method as illustrated in Fig. 5b, which is conducted using only their geometric distances. As a result, layer 0 has the A, B, C, and D points, and layer 1 has the I, J, K and L points. The points E, F, G, and H are not preserved. The CHD method proposed in this paper measures the color distance of point pairs on the same projection line, such as A and I, B and J, C and K, and D and L. Then the CHD method compares the average value of color distance with the threshold value. When the average value of color distances is larger than the threshold value, the points in the patch are separated into two patches according to their homogeneity as shown in Fig. 5d and Fig. 5e.

Since layer 0 and layer 1 are judged to have different contextual regions, the current patch is separated as shown in Fig. 5c. All the points in layer 1 in Fig. 5b are moved to a newly created patch as shown in Fig. 5d. The newly created patch is regarded as one complete patch with one contextual region. The CHD method is performed for the separated patch in Fig. 5c until the average color distance of all layers is smaller than the threshold value. The algorithm of the proposed CHD method is illustrated in Fig. 7.

As previously described, a patch is a group of points with similarities along a normal direction at each point. Thus, a patch cannot be generated when there are not enough neighboring points. The threshold value for the CHD method should be selected not to lose too many of the neighboring points. As an example, the CHD method may decompose points E and F in Fig. 6, and then classify them as candidate points for a new patch. However, the lack of neighboring points around the points E and F prohibits calculating a normal vector for each point using the geometric relationship of neighboring points. Thus, the points are ignored by the TMC2 configuration and are neither compressed nor reconstructed. In TMC2, such points are called missed points.
For each patch, the following applies:

widthPatch = GetPatch2DSizeX(thisPatch)
heightPatch = GetPatch2DSizeY(thisPatch)

For u = Range(0, widthPatch), v = Range(0, heightPatch), the following applies:

pointArray = GetPointsWithSameUV(points_in_patch, u, v)
pointL0 = pointArray[0]

For thickness = Range(1, preconfiguredThicknessLevel), the following applies:

colorDistanceL0candidate = pointL0[thickness]

colorDistanceL0candidate =
sqrt((pointL0candidate.r - pointL0.r)**2
+ (pointL0candidate.g - pointL0.g)**2
+ (pointL0candidate.b - pointL0.b)**2)

sumColorDistanceArray[thickness] +=
colorDistanceL0candidate

countColorDistanceArray[thickness] += 1

For thickness = Range(1, preconfiguredThicknessLevel), the following applies:

if (sumColorDistanceArray[thickness] / countColorDistanceArray[thickness] > CHD_threshold_value)
return (thickness - 1)

return preconfiguredThicknessLevel

Figure 7. Pseudo code algorithm of the proposed CHD method.

Table 1 shows the comparison in terms of the number and the ratio of missed points between the V-PCC reference software and the proposed CHD method depending on different CHD threshold values. These missed points cause geometric quality degradation in reconstructed point cloud content.

Table 2 and Fig. 8 show Bjontegaard Functions (BD-Rate) [29] of the Queen sequence with different CHD threshold values, where the BD-Rates in D1 and D2 show the geometry position error, and the Luma, Cb and Cr show the color error as described in [30], [31]. As the CHD threshold value increases, the D1 and Y PSNR values increase as well. Since increment of the CHD threshold above 12 are not significant, it is therefore reasonable to select 20 as the CHD threshold value to have a good balance between quality degradation and 2D compression gain.

V. EXPERIMENTAL RESULTS

The CHD method proposed in this paper is implemented and compared with the reference software TMC2. In chapter IV, we found a trade-off relationship between missed points and the BD-rate, as shown in table 1 and table 2, and concluded a CHD threshold value of 20 is appropriate. To identify the effectiveness of the proposed method, the experiment is conducted in terms of a whole test sequence for an overall performance as well as a part of the test sequence for motion estimation effects, as shown in table 3 and table 4. As was discussed in chapter III, the Queen sequence has a larger number of patches with multiple different contextual regions and thus is more suitable for evaluating motion estimation effects. To analyze the effectiveness of the proposed CHD method in terms of motion, the Queen sequence is divided into 5 sub-sequences, where each sub-sequence has 50 frames. Further, the CHD threshold of 20 is used in this experiment.
TABLE 3. Performance comparison of TMC2 and the CHD method in Random Access mode with a threshold of 20.0.

| Sub-sequence id | D1   | D2   | Luma | Cb  | Cr   |
|-----------------|------|------|------|-----|------|
| 1               | -1.4 | -1.6 | -13.8| -13.0| -9.4 |
| 2               | -2.2 | -2.3 | -9.5 | -6.7 | -7.2 |
| 3               | -0.4 | -0.4 | -7.5 | -4.9 | -3.9 |
| 4               | 0.7  | 0.5  | -3.2 | -6.0 | -1.8 |
| 5               | 0.6  | 0.8  | -5.0 | -4.2 | -4.9 |
| Average         | -0.5 | -0.6 | -7.8 | -7.0 | -5.5 |

TABLE 4. Performance comparison of TMC2 and the CHD method in All-Intra mode with a threshold of 20.0.

| Sub-sequence id | D1   | D2   | Luma | Cb  | Cr   |
|-----------------|------|------|------|-----|------|
| 1               | -0.8 | -0.7 | -11.7| -9.7 | -5.9 |
| 2               | -0.3 | -0.4 | -6.5 | -2.9 | -2.9 |
| 3               | 0.0  | 0.0  | -6.9 | -2.4 | -2.5 |
| 4               | 0.9  | -0.1 | -4.4 | -1.9 | -1.2 |
| 5               | 0.7  | 0.8  | -5.4 | -4.2 | -3.9 |
| Average         | -0.1 | 0.0  | -7.0 | -4.2 | -3.3 |

FIGURE 9. Comparison of outputs from (a) TMC2 and (b) the proposed CHD method.

The objective comparison of random access mode and all-intra mode are shown in Table 3 and Table 4 respectively. The CHD method has achieved overall better improvements in all criteria when compared to TMC2. As shown in Table 3, the required bit rate in random access mode for D1 is 0.5%, D2 is 0.6%, Luma is 7.8%, Cb is 7.0% and Cr is 5.5% smaller than TMC2 on average. Also note that Luma for the first sub-sequence has a 13.8% smaller bit rate than TMC2. As shown in Table 4, the required bit rate in all intra mode for D1 is 0.1%, D2 is 0.0%, Luma is 7.0%, Cb is 4.2% and Cr is 3.3% smaller than TMC2.

A subjective comparison of the output is shown in Fig. 9, where the two images are texture packing images from TMC2 and the CHD method. In Fig. 9a, the output from TMC2 shows black dots on the sleeves and collar patches, which are the shadows behind the cloth. Sleeves, collar and shadow are different contextual regions, however they are taken as one patch by TMC2. Fig. 9b, the output from the CHD method, shows fewer black dots on both the sleeves and collar because the proposed CHD method distinguishes shadows as different contextual regions, then decomposes this region from the cloth.

Therefore, it is shown that the CHD method proposed in this paper allows significant performance gain versus TMC2. The CHD method was proposed for MPEG and has been adopted into TMC2.

VI. CONCLUSION

Point cloud content is widely used to capture, store, and represent 3D volumetric objects in media. However, the amount of data for point cloud content is much larger than that of a 2D projected media, which prevents various industries from developing a wider range of applications. Thus, it is necessary to develop point cloud content compression technology for a wider range of applications and corresponding industry deployments. This paper proposes a method based on contextual homogeneity of a point cloud, which improves the compression rate of a reference software for MPEG V-PCC.

The reference software decomposes 3D points into 2D patches by referring to the geometry information only, which causes drawbacks in decomposing 3D points, while each patch may consist of multiple contextual regions. Thus, this paper provides a new patch decomposition method based on contextual homogeneity, which prevents a single patch from having more than one contextual region in terms of both color and geometry information. This method then generates a packing image with those contextual patches, allowing for more consistency and a higher compression rate.

The proposed CHD method is used to obtain contextual homogeneity in a patch as follows. First, the color distance of each point pair between both projected textual images of layer 0 and 1 is measured to determine the degree of discrepancy between those textual images. Next, the average color distance of all points in the patch is evaluated to determine whether this patch can be identified as homogeneous or not. When the evaluated average color distance is larger than a certain threshold, this patch is considered to have multiple contextual regions, and needs to be decomposed. The results presented in Table 3 and Table 4 in Chapter V show bitrate savings of 0.5%, 0.6%, 7.8%, 7.0% and 5.5% in random access mode and 0.1%, 0.0%, 7.0%, 4.2% and 3.3% in all intra mode for D1, D2, Y, Cb, and Cr, respectively, compared to the reference software. A subjective comparison also shows successfully separated contextual regions. Therefore, we have demonstrated that the CHD method proposed in this paper brings significant performance gain versus the reference software.

Further investigation on additional attributes, such as material ID or reflection in the V-PCC, provide good research directions to improve the compression performance of the reference software.
