# MULTI-CODEC RATE ADAPTIVE POINT CLOUD STREAMING FOR HOLOGRAPHIC-TYPE COMMUNICATION

Mahendra Suthar[1], Rui Dai[2], Junjie Zhang[3], Sasu Tarkoma[4], Ian F. Akyildiz[4]
[1]Department of Computer Science, University of Cincinnati, Cincinnati, OH 45221, USA, [2]Department of Electrical and Computer Engineering, University of Cincinnati, Cincinnati, OH 45221, USA, [3]Department of Computer Science and Engineering, Wright State University, Dayton, OH 45435, USA, [4]Computer Science Department, University of Helsinki, Finland

NOTE: Corresponding author: Rui Dai, rui.dai@uc.edu

***Abstract*** – *Point cloud videos play a crucial role in immersive applications enabled by holographic-type communication, which has been identified as an important service for 6G and beyond wireless systems and the metaverse. The significant volume of point cloud video demands efficient compression and transmission techniques to support the Quality of Experience (QoE) requirements of real-time immersive applications. A few Point Cloud Compression (PCC) techniques, such as MPEG PCC and Draco, have emerged in recent years, and studies have shown that each technique has its strengths and weaknesses under different system settings. This paper proposes a multi-codec rate adaptive point cloud streaming method to satisfy the QoE requirements of interactive and live applications considering available system resources. The proposed method leverages three common PCC techniques: MPEG V-PCC, MPEG G-PCC, and Draco. The performance of each PCC technique is evaluated under various test conditions, and then estimation models are constructed to predict the bit rate, the decoding time, and the quality of the reconstructed point cloud. Based on the user's quality requirements and available computational and communication resources, the proposed streaming method selects a codec along with appropriate compression parameters that can provide the minimum latency for streaming. Evaluation results demonstrate that the proposed method can provide better QoE than benchmark methods under various bandwidth and computation scenarios.*

**Keywords** – Compression, Draco, MPEG G-PCC, MPEG V-PCC, point cloud video, quality of experience, streaming

## 1. INTRODUCTION

Holographic-Type Communication (HTC), which can send holograms and other multisensory data through wireless and wired networks to remote locations, has been identified as an important service for 6G wireless systems and the metaverse [1]. A hologram is a recording of a light field that captures the original properties of depth and parallax of real 3D objects and humans. Nowadays, a point cloud is the most favored representation of holograms. A point cloud is a set of 3D points, where each point is associated with a geometry position and additional characteristics like color and reflectance. Point cloud videos can easily support 6 Degrees of Freedom (6DoF) immersive viewing experience, either through head-mounted displays or for naked eyes through light field displays. Point cloud videos are expected to advance the user experience over multiview videos and 360-degree videos for a wide range of HTC and metaverse applications, such as education, healthcare, entertainment, and holographic telepresence [1, 2, 3].

The significant volume of point cloud videos poses challenges in terms of processing, storage, and efficient transmission. For instance, transmitting a 0.7 million point cloud per 3D frame at 30 frames per second requires a bandwidth of around 500 MB/s [4]. Several Point Cloud Compression (PCC) techniques have emerged in recent years to address this challenge [5, 2, 6, 7]. MPEG Video-based Point Cloud Compression (V-PCC) and Geometry-based Point Cloud Compression (G-PCC) are the major standards for point cloud compression [2]. V-PCC converts point clouds into 2D patches and utilizes existing 2D video codecs (e.g., High Efficiency Video Coding (HEVC)) to compress them. G-PCC directly encodes content in the 3D space based on geometry presentation and attribute transformation. The Draco software library developed by Google compresses meshes and point clouds using KD tree formation together with quantization and entropy coding [5]. A few deep learning-based point cloud compression techniques have also been proposed, which utilize neural networks to learn the underlying patterns and structures in a point cloud to efficiently represent it with fewer bits [8, 9, 10]. In [11], these PCC methods were compared in terms of compression ratio, processing time, and quality of the reconstructed point clouds, and the results showed that each method has its strengths and weaknesses in different test conditions, which suggested the need in selecting or combining techniques to address application-specific constraints.

Adaptive streaming methods have been proposed to transmit pre-recorded or stored point cloud videos to meet users' Quality of Experience (QoE) requirements. Originating from multi-viewpoint and VR video streaming

applications, a common method for point cloud streaming is based on tiling: a 3D video stream is split into tiles in space, and a user can decide which areas of a scene to download at higher qualities to avoid wasting communication resources. Park et al. devised a method to reduce the bandwidth consumption by employing 3D tiles and adapting their level of detail based on the user's view [12]. They introduced a rate-utility optimization algorithm to allocate bits among tiles, considering representation quality, level of detail, and the user's device resolution and viewpoint while minimizing latency. Li et al. proposed a strategy for resource allocation in QoE-driven point cloud video streaming considering limited resources and selecting quality levels for each segmented point cloud video tile [13]. Wang et al. proposed an adaptive streaming approach for point cloud data transmission that enhances user QoE and reduces transmission redundancy. It incorporates factors like the user's view frustum, tile occlusion, and rendering device resolution to model the QoE of each 3D tile [14].

The success of tile-based streaming methods relies on accurate prediction of a user's behavior or viewing angle. Although viewing angle prediction has been well studied in VR and 360-degree video streaming systems, user behaviors in point cloud video systems become more complicated and difficult to predict due to the enriched 6DoF, where users can change not only the viewing angles but also the distances from the scene. Furthermore, the aforementioned tile-based methods are designed for the streaming of stored point clouds, and it is more challenging for interactive and live streaming applications to predict and adjust to user behaviors under stringent time constraints. The tile-based streaming methods are also limited for the use case of accommodating multiple users and multiple viewpoints at one destination.

In our study, we aim at providing a holistic visualization of an entire point cloud at a destination for interactive and live applications. We propose to exploit the characteristics of existing PCC methods to design a multi-codec rate adaptive point cloud streaming method to satisfy the QoE requirements of holographic-type communication. The contributions of our work are as follows:

- We conducted a systematic comparison of three common PCC technologies (MPEG V-PCC, MPEG G-PCC, and Draco) using different point cloud content at various spatial resolutions. PCC techniques could involve high computational time to achieve good compression performance, which cannot be neglected for interactive and live applications. Therefore, the performance of each PCC technique was tested in terms of computational complexity and compression performance, counting in both computation and communication delays. Based on the test results, we have developed accurate prediction models for the bit rate, the decoding time, and the quality of the reconstructed point cloud for the three codecs.
- We have designed a multi-codec rate adaptive point cloud streaming method using the developed prediction models. Based on the user's quality requirements and available computational and communication resources, the proposed streaming method selects a codec along with appropriate compression parameters that can provide the minimum latency for streaming. We have evaluated the proposed method based on various streaming scenarios considering different computational and bandwidth constraints.

The subsequent sections of this paper are organized as follows: Section 2 shows our preliminary study on comparison of point cloud codecs V-PCC, G-PCC and Draco. Section 3 introduces the proposed bit rate, decoding time, and quality estimation models followed by the multi-codec rate adaptive streaming method. Section 4 shows the performance evaluation of the proposed method, and finally Section 5 concludes the paper.

## 2. PRELIMINARY STUDY

We have conducted a comprehensive study comparing PCC methods based on coding experiments across various point cloud videos. In the rest of this section, we introduce the dataset, the evaluation methodology, and the test results for this comparison.

### 2.1 Dataset and evaluation methodology

Our study selected 10 point cloud videos from the following three public datasets: Microsoft Voxelized Upper Bodies (MVUB)[15], 8i Voxelized Full Bodies (8iVFB)[16], and Owlii Dynamic Human Textured Mesh Sequence (ODHTMS)[17]. The selected point cloud videos, which are summarized in Table 1, include varying spatial and temporal details and three different resolutions: 512, 1024, and 2048. The dataset consists of human subjects, both full-body and upper-body, and included both geometric and color properties. Prior research in the field of point cloud codec comparisons utilized these identical datasets [11, 18, 19], and they were similarly employed in examining solutions for point cloud streaming [13, 20, 21, 22].

We performed compression and streaming experiments on the point cloud videos using a testbed with client-server architecture. The server is a high-performance PC with Ubuntu 18.04.5 LTS, an Intel Core i9-7900X CPU @ 3.30GHz × 20, NVIDIA TITAN Xp/PCIe/SSE2, and 64GB RAM. It uses Python Flask and Nginx server, which is an open-source software offering support for web serving, load balancing, caching, and media streaming functionalities [23]. The client is a Python script running on an office laptop with Ubuntu 18.04.6 LTS, an Intel® Core™ i5-5200U CPU @ 2.20GHz × 4, Intel® HD Graphics 5500, and 16GB RAM. The server encodes a point cloud video and sends it to the client, and the client decodes the video and renders it for display.

**Table 1** – Point cloud videos dataset

| Name | Spatial Resolution | No. of Frames | Avg. No. of Points |
|---|---|---|---|
| Andrew9 | 512 | 318 | 279664 |
| Sarah9 | 512 | 207 | 302437 |
| Andrew10 | 1024 | 318 | 1276312 |
| Sarah10 | 1024 | 207 | 1355867 |
| Loot | 1024 | 300 | 784142 |
| Redandblack | 1024 | 300 | 729133 |
| Longdress | 1024 | 300 | 765821 |
| Soldier | 1024 | 300 | 1059810 |
| Basketball_player | 2048 | 600 | 2880057 |
| Dancer | 2048 | 600 | 2592758 |

We evaluated the performance of three point cloud compression codecs: MPEG V-PCC, MPEG G-PCC, and Draco. The encoding parameters are shown in Table 2. For each encoder, each point cloud video was encoded using five bit rate settings with lossy compression. For V-PCC, the point clouds were encoded in intra mode according to the MPEG V-PCC common test conditions [24] using MPEG V-PCC v11. V-PCC utilizes three parameters to control the resulting bit rate: GeometryQP(GQP), AttributeQP(AQP), and OccupancyPrecision(OP). For G-PCC, the G-PCC v14 with octree-predlift coding mode for lossy geometry and color coding was applied under the MPEG G-PCC common test conditions [24]. G-PCC could adjust the rate through two parameters: PositionQuantizationScale(PQS) and qp(QP). The Google Draco v1.5.6 software was used for this experiment. For Draco, the parameter QP controls the bits allocated for position quantization, and similarly the parameter QT controls the bits for coordinate texture attributes.

## 2.2 Preliminary evaluation results

We performed a comprehensive evaluation of the three PCC methods by assessing the compression ratio, the quality of reconstructed point clouds, as well as the processing time and transmission latency under different bandwidth settings.

First, for fair comparison, we selected the rate profiles of the three PCC methods that result in comparable quality of the reconstructed point clouds. We utilized the PSNR D2 (point-to-plane Peak Signal-to-Noise Ratio) metric to evaluate the quality of the reconstructed point cloud compared to the original point cloud [24] [25]. All the point cloud videos with the same spatial resolution in Table 1 were encoded using V-PCC, G-PCC, and Draco at the five rate profiles, and then the average values were considered. After averaging the results, one rate profile was selected for each codec at each spatial resolution based on similar PSNR D2 values with the minimum standard deviation. Table 3 shows the selected rate profiles.

Under the selected profiles that generate a similar quality of point clouds, the decoding times and the Compression Ratios (CR) of the three codecs are summarized in Table 4, with both average and standard deviation values obtained from our dataset. Draco has the lightest decoder, resulting in the smallest decoding time compared to V-PCC and G-PCC at all spatial resolutions, with decoding recorded below 1 second for all spatial resolutions. Compared to V-PCC, although G-PCC resulted in a smaller decoding time at a lower spatial resolution (512), it increased significantly at resolutions of 1024 and 2048. The decoding times for V-PCC did not change much when the resolution increased from 1024 and 2048. On the other hand, Draco resulted in much lower compression performance (lower CR values) compared to V-PCC and G-PCC at all resolutions. V-PCC performs impressively at higher resolutions, while G-PCC shows minor deviations without notable improvement. V-PCC's CR is over 10 times higher than G-PCC's at 2048 resolution and about five times higher at 1024 resolution. This is due to V-PCC's advanced 2-D video encoding method for 3-D point cloud compression. Therefore, an encoder with a higher CR can generate a smaller bitstream that can be transmitted faster, but at the expense of a longer decoding time.

For processing time estimation, we focused on decoding time. In on-demand point cloud streaming, the time taken for encoding is a crucial factor. We omitted it while calculating the total latency, assuming the existence of a high-performance server that can handle real-time encoding. This assumption allows us to assess the efficiency of other components in the streaming pipeline, such as network transmission and decoding, as they significantly impact on the overall user experience. The total latency, including the processing time and the transmission time, is estimated by:

$$\begin{aligned} Total\ Latency &= Transmission\ time + Computation\ time \\ &= \frac{Bitstream\ size}{Bandwidth} + Decoding\ Time \end{aligned} \quad (1)$$

To study the effect of different bandwidths on total la-

**Table 2** – Encoding parameters for V-PCC, G-PCC and Draco

| Rate Profile | V-PCC Rate parameters | | | G-PCC Rate parameters | | Draco Rate parameters | |
|---|---|---|---|---|---|---|---|
| | GQP | AQP | OP | PQS | QP | QP | QT |
| R1 | 32 | 42 | 4 | 0.125 | 51 | 7 | 6 |
| R2 | 28 | 37 | 4 | 0.25 | 46 | 9 | 8 |
| R3 | 24 | 32 | 4 | 0.5 | 40 | 11 | 10 |
| R4 | 20 | 27 | 4 | 0.75 | 34 | 15 | 14 |
| R5 | 16 | 22 | 2 | 0.875 | 28 | 15 | 14 |

**Table 3** – Selected rate profiles for comparison for 512, 1024 and 2048 spatial resolution

| Codec | 512 Spatial Resolution | | | 1024 Spatial Resolution | | | 2048 Spatial Resolution | | |
|---|---|---|---|---|---|---|---|---|---|
| | Rate Profile | Avg. PSNR D2 | Std. Dev. | Rate Profile | Avg. PSNR D2 | Std. Dev. | Rate File | Avg. PSNR D2 | Std. Dev. |
| V-PCC | R3 | 64.91 | 0.23 | R2 | 70.04 | 0.73 | R1 | 75.63 | 0.20 |
| G-PCC | R4 | 64.89 | 0.02 | R4 | 70.42 | 0.75 | R5 | 74.72 | 2.44 |
| Draco | R5 | 64.96 | 0.47 | R5 | 70.40 | 0.73 | R4 | 75.44 | 0.61 |

tency(eq. *(1)*) to stream point clouds, the average bitstream size for each of the resolution was streamed under varying bandwidths ranging from 1 Mbps to 512 Mbps, and the results of total latency are illustrated in Fig. 1. Though for an uncompressed point cloud, the decoding time is zero, it is clear that the total time to stream the entire uncompressed point cloud sequence is really high at lower bandwidths, which demands for efficient compression solutions. Although the bitstream sizes of V-PCC and G-PCC are significantly smaller than that of Draco, their total latency is very high and remains constant when the bandwidth increases. This is mainly because the high decoding times of V-PCC and G-PCC are the dominant factors in total latency. Alternatively, Draco's decoding time is almost negligible but the higher bitstream size is the dominant factor for its total latency.

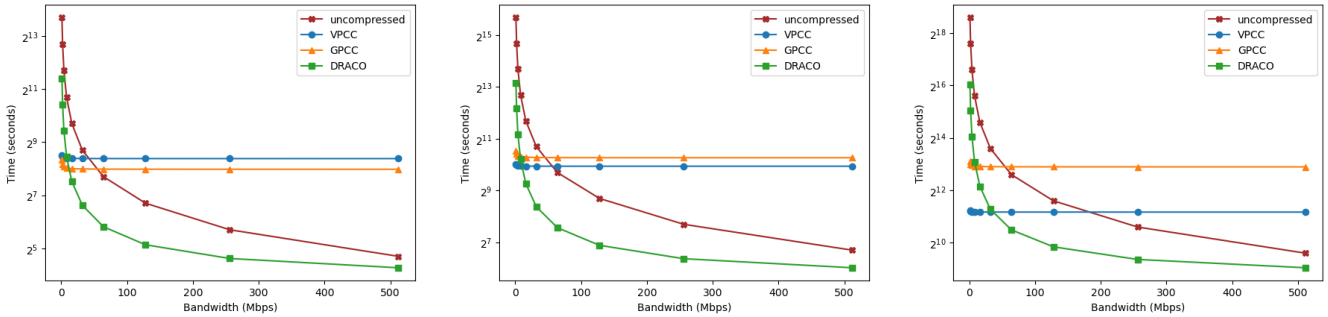
Fig. 2 shows the impact of quality representations on latency for three bandwidth settings (low, medium, and high) for point clouds with 1024 spatial resolution. The quality of a point cloud is evaluated by PSNR D1 (point-to-point Peak Signal-to-Noise Ratio). V-PCC consistently achieves higher point cloud quality compared to G-PCC and Draco, and it would be selected when there is high quality demand and the client has high computational power to reduce the decoding time. The smaller total latency of G-PCC with lower reconstructed quality could be useful for smaller resolution devices with lower bandwidth and lower computational power. Draco may be adopted to achieve lower latency when there is sufficient bandwidth to support its larger bitstream size.

## 3. MULTI-CODEC RATE ADAPTIVE STREAMING

The preliminary results in the previous section reveal the trade-off between computational complexity and compression performance, and they also indicate the need to consider both communication time and computation time for real-time applications. Depending on the available resources and the QoE requirements, different codecs may be used to optimize system performance. Our findings in the previous section indicate that V-PCC is selected for scenarios with severely constrained bandwidth, while Draco is the favored choice in cases of abundant bandwidth availability. On the other hand, G-PCC is the optimal selection when resources like computational power and bandwidth are limited. Furthermore, codec choice is also influenced by the quality requirements of the user. We propose a multi-codec rate adaptive (MC_RA) point cloud streaming solution for real-time HTC applications. The key components of the solution are illustrated in Fig. 3. After a point cloud video is acquired from a capturing system, the server runs the MC_RA optimization algorithm to decide the best codec along with the encoding parameters that result in the minimum total latency while satisfying the quality requirement. To make intelligent decisions, the algorithm leverages estimation models on bit rate, decoding time at the client, and point cloud video quality. At the client side, the point cloud video stream is buffered, decoded, and displayed, and feedback information (e.g., the end-to-end bandwidth and the client's computational power) are sent back to the server to improve the accuracy of the proposed estimation models. Our estimation models are constructed based on four datasets, each with different spatial resolutions: andrew9 (512), loot (1024), soldier (1024), and dancer (2048). We also tested the computational capability of the client device in our testbed to derive the estimation models. In the rest of this section, we first introduce the proposed estimation models on bit rate, decoding time, and quality, and then we explain details of the entire MC_RA which leverages the constructed estimation models.

**Table 4** – Comparison of decoding time and compression ratio

| Codec | 512 Spatial Resolution | | 1024 Spatial Resolution | | 2048 Spatial Resolution | |
|---|---|---|---|---|---|---|
| | Decoding Time(sec) | Compression Ratio(CR) | Decoding Time(sec) | Compression Ratio(CR) | Decoding Time(sec) | Compression Ratio(CR) |
| V-PCC | 1.27±0.04 | 601.08±85.8 | 3.41±0.63 | 1112.61±97.1 | 3.83±0.16 | 3278.15±67.5 |
| G-PCC | 0.96±0.07 | 185.87±41.9 | 4.28±1.88 | 206.34±37.5 | 12.67±0.45 | 313.81±24.3 |
| Draco | 0.05±0.00 | 4.92±0.72 | 0.16±0.03 | 5.86±2.63 | 0.66±0.05 | 5.96±0.43 |




(a) 512 Resolution (b) 1024 Resolution (c) 2048 Resolution

**Fig. 1** – Total latency under varying bandwidths

## 3.1 Bit rate estimation

The estimation of bit rate (i.e., the compressed point cloud video size per unit time) is necessary as it is directly related to the transmission time. The size of a bitstream is not only related to the content characteristics of a point cloud video but also determined by the level of compression. Similar to 2-D videos, quantization is a crucial step that controls the compression level by reducing the precision of data. We introduce bit rate models for each codec (V-PCC, G-PCC, and Draco) based on specific quantization parameters.

**V-PCC bit rate estimation:** The V-PCC bitstream is composed of four primary sub-streams: geometry video, attribute video, occupancy map video, and atlas data. The occupancy video and the atlas are encoded in a lossless manner, and they do not significantly impact the overall size of the bitstream. Thus, the most dominant parts for the compressed bitstream size are geometry video and attribute video which are encoded using the 2-D video encoder HEVC. The encoded geometry video size is controlled by GeometryQP ($GQP$), and the encoded attribute video size is controlled by AttributeQP ($AQP$). Fig. 4(a) gives an example of the V-PCC bit rate for the Loot dataset under different $GQP$ and $AQP$ values. We also observed similar relationships among bit rate, $GQP$, and $AQP$ in the other point cloud videos in our dataset. We adhere to the previous study in [26] for estimating the bit rate of V-PCC. The bit rate for V-PCC in intra mode can be estimated by summing the geometry bit rate and the attribute bit rate [26], which is given by:

$$R_{V-PCC} = v_{e1} \cdot GQP^{\,v_{e2}} + v_{e3} \cdot AQP^{\,v_{e4}} \qquad (2)$$

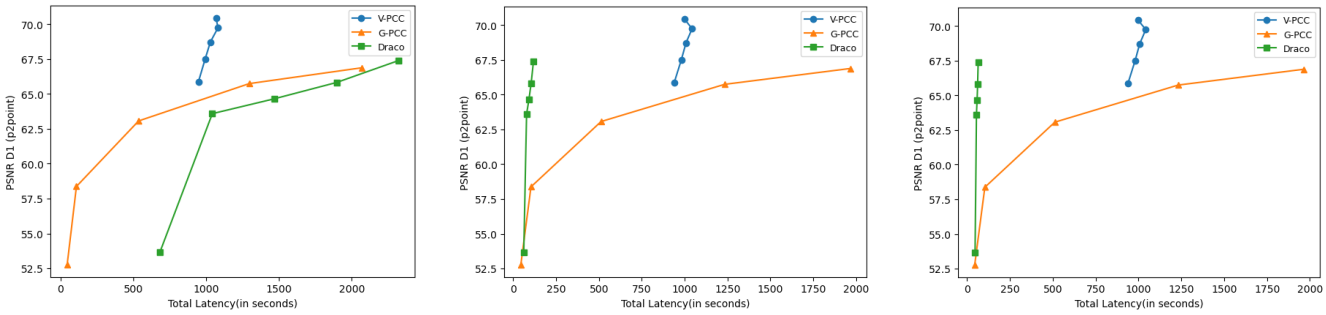
where $v_{e1}$, $v_{e2}$, $v_{e3}$, and $v_{e4}$ are content-dependent parameters. We have estimated these parameters and verified the accuracy of this model using our dataset.

**G-PCC bit rate estimation:** A recent G-PCC bit rate model was introduced in [27] based on tests on sparse point cloud datasets; however, we have found that this model does not fit dense point cloud sequences. Therefore, we have developed a new G-PCC rate model based on our dataset. In G-PCC, the parameters PositionQuantizationScale ($PQS$) and $QP$ (attribute quantization parameter) control the bit rate. The former determines the depth of the octree and affects the geometry bitstream size, while the latter controls the attribute quantization and influences the attribute bitstream size. We experimented with 10 different $PQS$ values and 10 different $QP$ values, resulting in the encoding of 100 combinations using the octree-predlift coding mode for lossy geometry and color coding. In G-PCC, the total bit rate is the sum of geometry bit rate and attribute bit rate. To estimate the geometry bit rate, we tried to fit a curve using the power model given by:

$$R_{G\_PCC,geometry} = g_{e1} \cdot PQS^{\,g_{e2}} \qquad (3)$$

The attribute coding in G-PCC is dependent on the decoded geometry, which means that the attribute bitstream size is also influenced by the geometry quantization [2]. To investigate this effect, we set the $QP$ values at 11, 21, 33, and 43 and analyzed its impact on the total attribute bit rate using the model which is given by:

$$R_{G\_PCC,attribute} = g_{e3} \cdot \frac{PQS^{\,g_{e4}}}{QP^{\,g_{e5}}} \qquad (4)$$




(a) Low Bandwidth - 4 Mbps (b) Medium Bandwidth - 128 Mbps (c) High Bandwidth - 512 Mbps

**Fig. 2** – Impact of different quality representations on total latency for 1024 spatial resolution

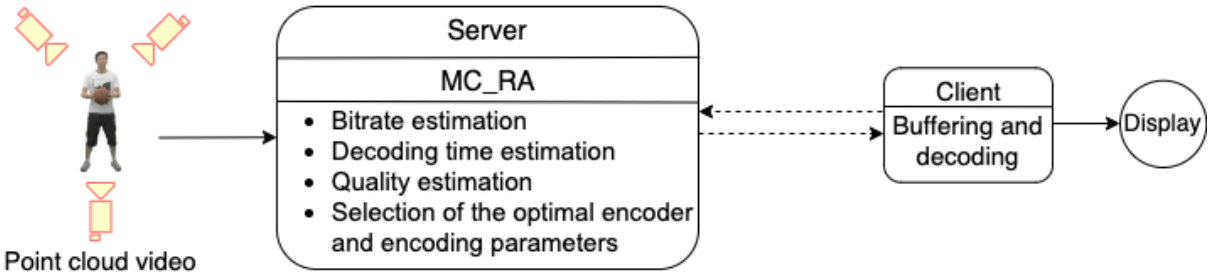



**Fig. 3** – System architecture of MC_RA

Finally, the total bit rate for G-PCC is estimated by the sum of the geometry bit rate and the attribute bit rate, which is given by:

$$R_{G-PCC} = g_{e1} \cdot PQS^{\,g_{e2}} + g_{e3} \cdot \frac{PQS^{\,g_{e4}}}{QP^{\,g_{e5}}} \qquad (5)$$

where $g_{e1}, g_{e2}, g_{e3}, g_{e4}, g_{e5}$ are content-dependent parameters. As an example, the estimation of the G-PCC bit rate for the Loot dataset is given in Fig. 4(b).

**Draco bit rate estimation:** The Draco documentation also specified geometry and texture quantization parameters, in which $QP$ controls the precision of quantization used for position data and $QT$ controls the texture quantization [5]. Based on our experimental results, $QP$ is the dominant factor contributing to bit rate. When the $QP$ value is fixed, we observed that the value of $QT$ has no impact on the total bitstream size. As shown in Fig. 4(c), the Draco bit rate for the Loot sequence is linearly correlated with $QP$, and similar linear relationships were observed in the other point cloud videos in our experiment. Therefore, Draco's bit rate can be estimated using $QP$ as follows:

$$R_{Draco} = d_{e1} + d_{e2} \cdot QP \qquad (6)$$

where $d_{e1}$ and $d_{e2}$ are content-dependent parameters.

## 3.2 Decoding time estimation

To estimate the total latency for streaming point cloud videos, we need to estimate the decoding time at the destination device. Given a certain level of computing capability, the decoding time for a codec is related to the compression level and the number of points in the point cloud [24]. We have developed decoding time estimation models using the configuration of the client device in our testbed, and the same models could be applied to other computing devices using customized parameters.

**V-PCC decoding time estimation:** Based on our experimental results, we have identified that the decoding time for V-PCC is linearly related to $GQP$ and $AQP$. Fig. 5(a) shows such linear relationship for the Loot sequence. The decoding time can be estimated by:

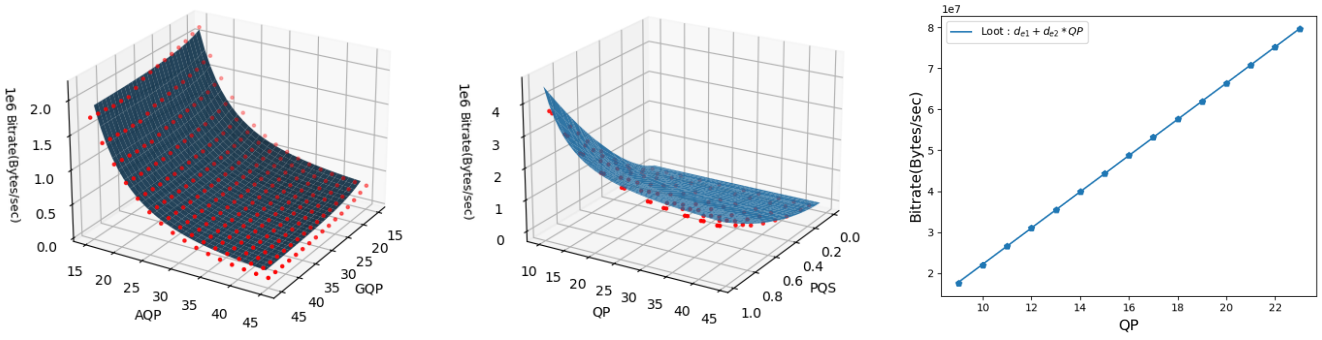
$$DT_{V-PCC} = v_{d1} + v_{d2} \cdot GQP + v_{d3} \cdot AQP \qquad (7)$$

where $v_{d1}, v_{d2}, v_{d3}$ are parameters that are dependent on both the point cloud content and the computing platform.

**G-PCC decoding time estimation:** Based on our experimental results, the parameter PositionQuantizationScale ($PQS$) is the dominant factor in decoding time and $QP$ has a slightly smaller impact on the total decoding time. An example for the Loot sequence is shown in Fig. 5(b). The relationship between $PQS$ and the decoding time can be depicted as a power model, while a linear model can be considered for $QP$ and the decoding time. The total decoding time for G-PCC is estimated by:

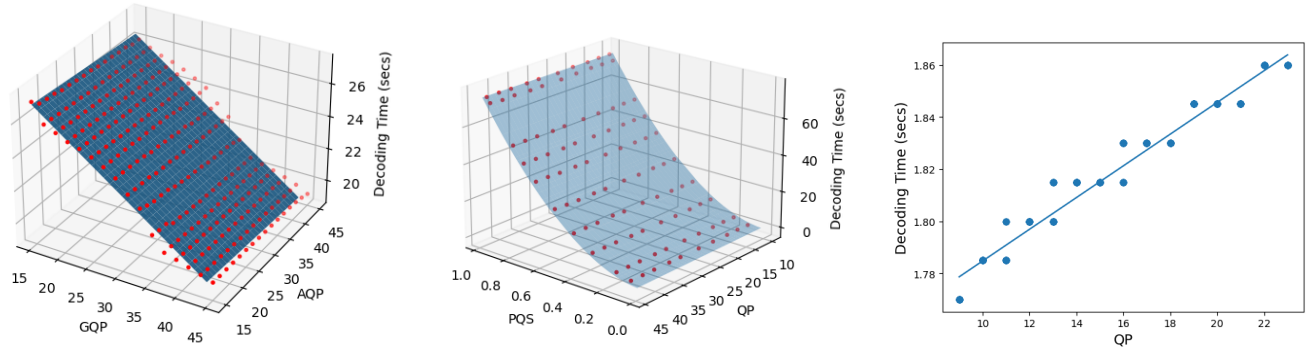
$$DT_{G-PCC} = g_{d1} \cdot PQS^{\,g_{d2}} + g_{d3} \cdot QP \qquad (8)$$

where $g_{d1}$, $g_{d2}$, and $g_{d3}$ are parameters that are dependent on both the point cloud content and the computing platform.




(a) V-PCC (b) G-PCC (c) Draco

**Fig. 4** – Illustration of the bit rate estimation models for Loot




(a) V-PCC (b) G-PCC (c) Draco

**Fig. 5** – Illustration of the decoding time estimation models for the Loot dataset

**Draco decoding time estimation model:** As the time for decoding in Draco is really small, it is hard to capture the decoding time pattern. However, our experimental results imply a linear relationship between $QP$ and decoding time as seen in Fig. 5(c). So, the total decoding time for Draco is given by:

$$DT_{Draco} = d_{d1} + d_{d2} \cdot QP \tag{9}$$

where $d_{d1}$ and $d_{d2}$ are parameters that are dependent on both the point cloud content and the computing platform.

## 3.3 Quality estimation model

Quality estimation models are also needed to satisfy users' requirements. We propose estimating the commonly used PSNR D1 metric [24] [25], which is derived from the point-to-point mean square error (MSE_P2P).

**V-PCC quality estimation:** The quality of the V-PCC point cloud's geometry is dependent on the parameter geometryQP ($GQP$), which regulates the compression of geometry and, thus, its quality. We chose to utilize the exponential model for V-PCC quality estimation in [26], which is given by:

$$Q_{V-PCC} = v_{q1} \cdot v_{q2}^{GQP} \tag{10}$$

where $v_{q1}$ and $v_{q2}$ are content-dependent parameters. This model produced superior curve fitting results on our dataset, and an example for the Loot sequence is shown in Fig. 6(a).

**G-PCC quality estimation:** As shown in Fig. 6(b), the quality of a G-PCC point cloud is correlated with PositionQuantizationScale($PQS$). Based on our experimental results, we propose a power model to establish the relation between $PQS$ and MSE_P2P.

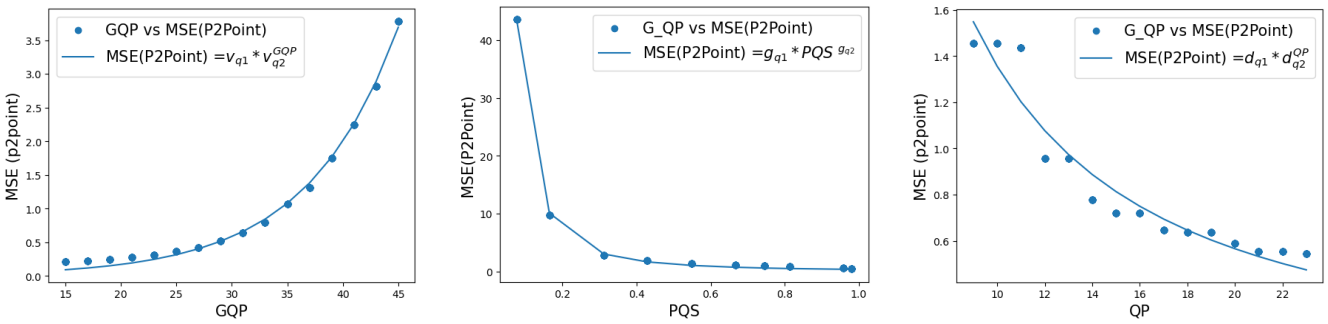
$$Q_{G-PCC} = g_{q1} \cdot PQS^{\,g_{q2}} \tag{11}$$

where $g_{q1}$ and $g_{q2}$ are content-dependent parameters.

**Draco quality estimation:** Similar as the case for for V-PCC, the quality of a reconstructed Draco point cloud exhibits an exponential relationship with the key parameter $QP$. Based on our experimental results, we have constructed the following quality estimation model for Draco:

$$Q_{Draco} = d_{q1} \cdot d_{q2}^{QP} \tag{12}$$

where $d_{q1}$ and $d_{q2}$ are content-dependent parameters.




(a) V-PCC (b) G-PCC (c) Draco

**Fig. 6** – Illustration of the quality estimation models for the Loot dataset

**Table 5** – Average accuracy of the bit rate, decoding time and quality estimation models

| Codec | bit rate | | | | Decoding time | | | Quality | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Model parameters | $R^2$ | RMSE | NRMSE | Model parameters | $R^2$ | RMSE | Model parameters | $R^2$ | RMSE |
| V-PCC | $v_{e1}, v_{e2}, v_{e3}, v_{e4}$ | 0.96 | 8969.6 | 0.042 | $v_{d1}, v_{d2}, v_{d3}$ | 0.92 | 0.072 | $v_{q1}, v_{q2}$ | 0.99 | 0.088 |
| G-PCC | $g_{e1}, g_{e2}, g_{e3}, g_{e4}, g_{e5}$ | 0.98 | 10579.6 | 0.027 | $g_{d1}, g_{d2}, g_{d3}$ | 0.99 | 0.036 | $g_{q1}, g_{q2}$ | 0.99 | 0.274 |
| Draco | $d_{e1}, d_{e2}$ | 0.99 | 25218.5 | 0.001 | $d_{d1}, d_{d2}$ | 0.91 | 0.001 | $d_{q1}, d_{q2}$ | 0.93 | 0.144 |

The curve fitting for this model is verified in Fig. 6(c).

## 3.4 Estimation of model parameters

In this section, we explain how to estimate the parameters in the bit rate, decoding time, and quality models. The values of all the parameters depend on the point cloud content. In addition, the parameters for decoding time are also dependent on the computing capability at the client/destination. Consequently, the parameter values implicitly capture the spatial resolution of the point cloud content. To obtain the model parameters, we encode three pairs of quantization parameters and also test the decoding time at the client machine in our testbed, and then we can find the solutions to the corresponding system of linear equations. For instance, by utilizing quantization pairs $(PQS_1, QP_1), (PQS_2, QP_2), (PQS_3, QP_3)$, we can derive the model parameters associated with G-PCC's bit rate, decoding time, and quality estimation models using equations *(13)*, *(14)*, *(15)*, respectively, as follows:

$$\begin{cases} R_{G\_PCC,geometry,1} = g_{e1} \cdot PQS_1^{g_{e2}} \\ R_{G\_PCC,geometry,2} = g_{e1} \cdot PQS_2^{g_{e2}} \\ \\ R_{G\_PCC,attribute,1} = g_{e3} \cdot \frac{PQS_1^{g_{e4}}}{QP_1^{g_{e5}}} \\ R_{G\_PCC,attribute,2} = g_{e3} \cdot \frac{PQS_2^{g_{e4}}}{QP_2^{g_{e5}}} \\ R_{G\_PCC,attribute,3} = g_{e3} \cdot \frac{PQS_3^{g_{e4}}}{QP_3^{g_{e5}}} \end{cases} \quad (13)$$

$$\begin{cases} DT_{G\_PCC,1} = g_{d1} \cdot PQS_1^{g_{d2}} + g_{d3} \cdot QP_1 \\ DT_{G\_PCC,2} = g_{d1} \cdot PQS_2^{g_{d2}} + g_{d3} \cdot QP_2 \\ DT_{G\_PCC,3} = g_{d1} \cdot PQS_3^{g_{d2}} + g_{d3} \cdot QP_3 \end{cases} \quad (14)$$

$$\begin{cases} Q_{G\_PCC,1} = g_{q1} \cdot PQS_1^{g_{q2}} \\ Q_{G\_PCC,2} = g_{q1} \cdot PQS_2^{g_{q2}} \end{cases} \quad (15)$$

where $R_{G\_PCC,geometry,1}$, $R_{G\_PCC,geometry,2}$, $R_{G\_PCC,color,1}$, $R_{G\_PCC,color,2}$, $R_{G\_PCC,color,3}$ are corresponding geometry and color bitstream sizes. $DT_{G\_PCC,1}$ , $DT_{G\_PCC,2}$ and $DT_{G\_PCC,3}$ are corresponding decoding times and $Q_{G\_PCC,1}$, $Q_{G\_PCC,2}$ are the corresponding reconstructed point cloud qualities.

## 3.5 Multi-codec rate adaptive streaming

With the aforementioned estimation models, we can then complete the MC_RA streaming solution that selects the best codec with encoding parameters at the server side to minimize the total latency while satisfying the quality requirement. The steps for the proposed solution are summarized in Algorithm 1. More specifically, after a point cloud video clip is captured, and given the user-required quality level in PSNR D1, we begin with the selection on the set of codecs, namely V-PCC, G-PCC, and Draco. We use three different pairs of quantization parameters for each of the codec to obtain the model parameters related to bit rate (for communication time), decoding time (for computation time) and quality estimation models (*line 3 in Algorithm 1*). The server could estimate the parameters

**Table 6** – Configuration table

| Level | Bandwidth(Mbps) | Computation | Quality (PSNR_D1) | | |
|---|---|---|---|---|---|
| | | | 512 spatial resolution | 1024 spatial resolution | 2048 spatial resolution |
| Low | 4 | 0.5 | 55 | 56 | 60 |
| Medium | 64 | 1 | 58 | 62 | 65 |
| High | 256 | 2 | 61 | 68 | 70 |

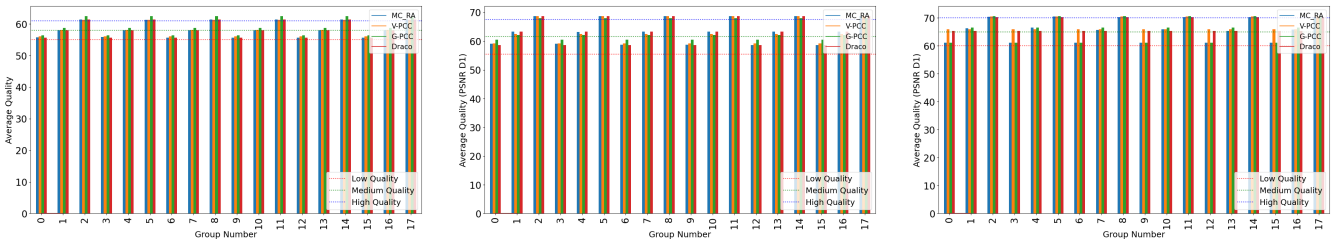
**Table 7** – Subset of streaming scenarios

| Group No. | Bandwidth(Mbps) Range | Computation Range | Quality (PSNR_D1) |
|---|---|---|---|
| 0 | Low - Medium | Low - Medium | Low |
| 1 | Low - Medium | Low - Medium | Medium |
| 2 | Low - Medium | Low - Medium | High |
| 3 | Low - Medium | Medium - High | Low |
| 4 | Low - Medium | Medium - High | Medium |
| 5 | Low - Medium | Medium - High | High |
| 6 | Low - High | Low - Medium | Low |
| 7 | Low - High | Low - Medium | Medium |
| 8 | Low - High | Low - Medium | High |
| 9 | Low - High | Medium - High | Low |
| 10 | Low - High | Medium - High | Medium |
| 11 | Low - High | Medium - High | High |
| 12 | Medium - High | Low - Medium | Low |
| 13 | Medium - High | Low - Medium | Medium |
| 14 | Medium - High | Low - Medium | High |
| 15 | Medium - High | Medium - High | Low |
| 16 | Medium - High | Medium - High | Medium |
| 17 | Medium - High | Medium - High | High |

**Algorithm 1** Multi-Codec Rate Adaptive (MC_RA) streaming method

1: Set $QR$ ▷ Set the Quality Requirement
2: Set $CODECS \leftarrow VPCC, GPCC, Draco$
3: Use three different pair of QPs for each of the codec to learn model parameters for bit rate, decoding time and quality estimation models.
4: **for each** $g \in GoF$ **do** ▷ $g$ is basic unit for playback, $GoF$ is Group of Frames
5: Set $CS \leftarrow \{\}$ ▷ $CS$ is Candidate Set
6: **for each** $codec \in CODECS$ **do**
7: Adjust QPs of $codec$ using quality estimation model such that $QR$ requirement is met, use eq. *(10)* for V-PCC, *(11)* for G-PCC, *(12)* for Draco
8: Calculate $codec$ communication time using eq. *(2)* for V-PCC, *(5)* for G-PCC and *(6)* for Draco
9: Calculate $codec$ computation time using eq. *(7)* for V-PCC, *(8)* for G-PCC and *(9)* for Draco
10: Calculate $codec$ total time (latency) using eq. *(1)*
11: Add $codec$ setting to $CS$
12: **end for**
13: GoF $g \leftarrow min\_time(CS)$ ▷ Select $g$ from $CS$ with minimum total time
14: Stream encoded GoF $g$ to the client (with specific codec and rate)
15: **end for**

for bit rate and quality based on the specific point cloud content, whereas the server would require the client to update its computation capability to estimate the decoding time model parameters. After the model parameters are obtained, we can meet the quality requirement by adjusting the quantization parameter using the developed quality model for each of the codec. For adjusting the QPs, we employ a modified binary search to fine-tune QPs and identify the QP pair that satisfies the desired quality standard and then we can calculate the total latency for each of the codec using the bit rate and decoding time estimation models (*lines 6-12 in Algorithm 1*). We then select the codec with the minimum total time for streaming (*lines 13-14 in Algorithm 1*). Our adapted binary search method operates within a predefined range of possible QP values for a codec. This approach aims to determine and return the QP value that most closely aligns with the specified quality requirement. However, if the user's quality requirement (PSNR D1) falls outside the bounds of the preset QP range for the codec, our modified binary search will automatically select either the lowest or highest available QP value, depending on the requirement. For instance, if the user specifies an exceptionally low quality requirement that cannot be met by the codec, we will opt for the best feasible QP value with the lowest possible quality. Conversely, if an extremely high quality requirement




(a) Andrew9 (b) Loot (c) Dancer

**Fig. 7** – Comparison of average quality

is given, which the codec cannot achieve, we will choose the feasible QP value within the codec's capabilities which achieves the highest possible quality. In all other scenarios, our implementation strives to identify the optimal QP value that best aligns with the specified quality requirement. The processing time for each group of frames is minimal due to the binary search's efficient time complexity. Moreover, our search is confined to a range of QP values for each codec. When compared to the encoding time, this processing time is relatively short, allowing it to be excluded when calculating the total latency.

## 4. PERFORMANCE EVALUATION

In order to validate the viability of our proposed scheme, we developed a simulation platform and carried out extensive experiments on the same testbed as in our preliminary study. First, we tested the accuracy of the proposed estimation models, and then we evaluated the networking performance of the entire MC_RA solution.

### 4.1 Evaluation of estimation models

In our experiments, three different pairs of QPs were sufficient for obtaining the model parameters for a specific point cloud sequence. For V-PCC the ($GQP$, $AQP$) pairs we selected were (15, 15), (30, 30) and (45, 45), while for G-PCC the ($PQS$, $QP$) pairs were (0.759, 11), (0.548, 25) and (0.958, 40). For Draco, only the $GQP$ was the contributing factor for the proposed models, and the selected $GQP$ values were 10, 15 and 20. Our estimation models are constructed and evaluated based on four point cloud video sequences, each with different spatial resolutions: andrew9 (512), loot (1024), soldier (1024), and dancer (2048). In addition, the client machine in our testbed was used for characterizing the decoding time. Table 5 shows the average $R^2$ and $RMSE$ for bit rate, decoding time, and quality estimation models for each of the codec. As the bit rate could be large we show the Normalized Root Mean Square Error (NRMSE), which is calculated by dividing the RMSE by the maximum value of the bit rate. The high average $R^2$ for all the developed models and lower RMSE and NRMSE values demonstrate the accuracy of the developed models.

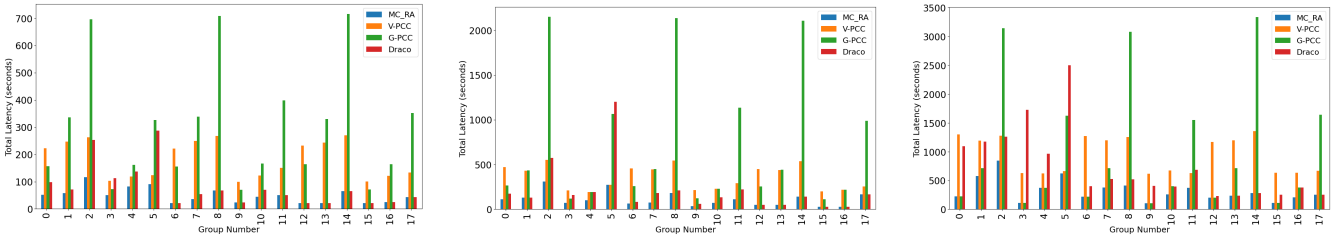
### 4.2 Evaluation of MC_RA streaming

In our simulation, we defined three levels of bandwidths (low, medium, and high), three levels of computation power (low, medium, and high), and three levels of quality (low, medium, and high). Additionally, the point cloud sequences were rendered for a duration of 20 seconds at a frame rate of 15 FPS and the unit for playback of point cloud video, referred to as the Group of Frames (GoF) denoted by *g*, is 15. This suggests that the point cloud videos are encoded and streamed in sets of 15 frames. Table 6 displays the values assigned to bandwidth, computation, and quality. The computational power of our client machine falls into the medium category, so the computation value on our system would be $computation = 1$ (level MEDIUM). A computation value of 0.5 would imply twice the decoding time on our system and a computation value of 2 would mean half the decoding time of value compared to our setup.

To evaluate the effectiveness of the proposed method, we created multiple tests that simulate different scenarios by combining the levels outlined in Table 6. In total, we generated 108 unique tests that cover all possible streaming scenarios, and a subset of the streaming scenarios we investigated are presented in Table 7. We introduced three baseline approaches: 1) encoding with only V-PCC, 2) encoding with only G-PCC, and 3) encoding with only Draco. All the three baseline approaches include the same QP selection strategy in Algorithm 1 to satisfy a certain quality requirement. We assess the performance of the proposed method using three point cloud sequences with varying spatial resolutions: andrew9 (512 resolution), loot (1024 resolution), and dancer (2048 resolution). For each point cloud, we conduct five simulations and present the average values for comparison.

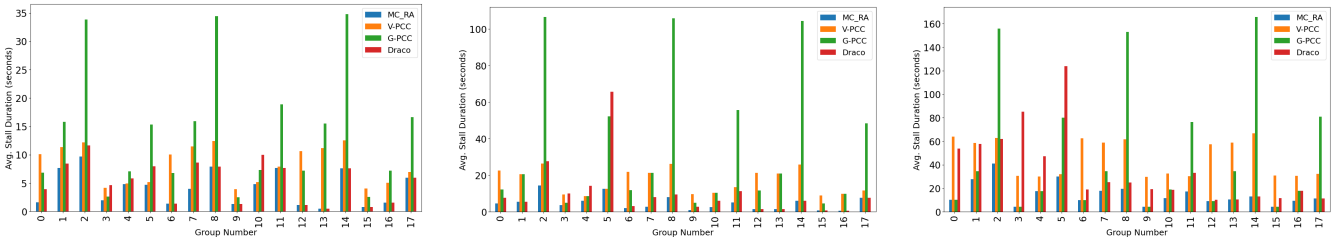
The average quality attained by each method is shown in Fig. 7. It is evident that all the methods have met the required quality levels of low, medium, and high. This is because all the methods have incorporated the QP selection scheme in Algorithm 1, which adjusts the QP to ensure that the quality requirement is met.

In addition to quality, we compare MC_RA against the three baseline approaches using the *total latency* and the *average stall duration*. The total latency, including the




(a) Andrew9 (b) Loot (c) Dancer

**Fig. 8** – Comparison of total latency




(a) Andrew9 (b) Loot (c) Dancer

**Fig. 9** – Comparison of average stall duration

transmission time and the decoding time, is a key factor for interactive and live streaming applications. A stall during a media playback refers to temporary pauses or stop, typically due to buffering issues, slow network connectivity, or insufficient system resources [28]. Stalling is a major factor that could degrade user experience.

Fig. 8 compares the total latency of MC_RA and the baseline approaches. In all scenarios, the total latency of MC_RA is at least equivalent to the baseline approach. The increased total latency observed in G-PCC can be attributed to the longer decoding time required at higher quality levels. On the other hand, V-PCC exhibits relatively consistent total latency across all subsets of scenarios. It is also evident that higher quality levels result in greater overall total latency. In group numbers 12-17, where higher bandwidth and computational resources are available, MC_RA's total latency is comparable to that of Draco for all the datasets. This is because MC_RA selects Draco due to its lower total latency compared to V-PCC and G-PCC. In the majority of scenarios within group numbers 0 to 11, MC_RA demonstrates significantly lower total latency compared to baseline approaches. Its total latency (for group numbers 0 to 11) was on average 36.5% lower than that of the best baseline approach, showcasing the adaptability of MC_RA to varying resource conditions. Furthermore, across all the subset of scenarios presented in Table 7, MC_RA consistently achieved an overall total latency that was 22.37% lower than the most optimal baseline method, with only a 1.85% standard deviation in results across five simulation tests. This highlights the consistent performance improvement offered by MC_RA in various streaming scenarios.

The average stall duration, as shown in Fig. 9, demonstrates MC_RA's smoother point cloud video playback with fewer stalls. MC_RA has a lower average stall duration compared to the baseline approaches in most scenarios, indicating minimal disruptions and shorter stall durations. However, G-PCC exhibits longer average stall durations for high-quality requirements due to its extended decoding time. Overall, for the streaming scenarios presented in Table 7, MC_RA consistently achieved an overall average stall duration that was 22.69% lower than the most optimal baseline method with a standard deviation of 1.88% in results across five simulation tests. These results confirm that MC_RA satisfies the quality standards efficiently and can adapt to various bandwidth and computation scenarios.

## 5. CONCLUSION

In this paper, we have investigated the streaming of point cloud videos for interactive and live HTC scenarios. We presented comparison of three point cloud codecs V-PCC, G-PCC, and Draco, in terms of compression performance and computation and communication costs. Since each codec has its best operating conditions, we have designed a Multi-Codec Rate Adaptive (MC_RA) streaming method that uses bit rate, decoding time and quality estimation models to select the best codec along with encoding parameters to achieve the minimum total latency while satisfying a user's quality requirement. Experiments were conducted for different point clouds with various system resource constraints. The proposed method has achieved much better performance in terms of latency and average stall duration compared to the baseline methods, boosting the QoE of real-time applications.

## REFERENCES


[1] I. F. Akyildiz and H. Guo. "Holographic-type Communication: A New Challenge for The Next Decade". In: *ITU Journal on Future and Evolving Technologies.* (2022).

[2] D. Graziosi, O. Nakagami, S. Kuma, A. Zaghetto, T. Suzuki, and A. Tabatabai. "An overview of ongoing point cloud compression standardization activities: Video-based (V-PCC) and geometry-based (G-PCC)". In: *APSIPA Transactions on Signal and Information Processing* 9 (2020), e13.

[3] Laura Singleton. *Imperial College Business School to offer live lectures via hologram*. https://www.imperial.ac.uk/news/188851/imperial-college-business-school-offer-live/. [Online; accessed 30-March-2023].

[4] C. Cao, M. Preda, and T. Zaharia. "3D point cloud compression: A survey". In: *The 24th International Conference on 3D Web Technology*. 2019, pp. 1–9.

[5] Google. *DRACO: 3D DATA COMPRESSION*. https://google.github.io/draco/. [Online; accessed 11-December-2022].

[6] A. F. R. Guarda, N. M. M. Rodrigues, and F. Pereira. "Adaptive Deep Learning-Based Point Cloud Geometry Coding". In: *IEEE Journal of Selected Topics in Signal Processing* 15.2 (2021), pp. 415–430. DOI: 10.1109/JSTSP.2020.3047520.

[7] J. Wang, D. Ding, Z. Li, and Z. Ma. "Multiscale point cloud geometry compression". In: *2021 Data Compression Conference (DCC)*. IEEE. 2021, pp. 73–82.

[8] T. Huang and Y. Liu. "3D point cloud geometry compression on deep learning". In: *Proceedings of the 27th ACM international conference on multimedia*. 2019, pp. 890–898.

[9] M. Quach, G. Valenzise, and F. Dufaux. "Improved deep point cloud geometry compression". In: *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*. IEEE. 2020, pp. 1–6.

[10] J. Wang, H. Zhu, H. Liu, and Z. Ma. "Lossy point cloud geometry compression via end-to-end learning". In: *IEEE Transactions on Circuits and Systems for Video Technology* 31.12 (2021), pp. 4909–4923.

[11] M. Bui, L. Chang, H. Liu, Q. Zhao, and G. Chen. "Comparative study of 3D point cloud compression methods". In: *2021 IEEE International Conference on Big Data (Big Data)*. IEEE. 2021, pp. 5859–5861.

[12] J. Park, P. A. Chou, and J. Hwang. "Rate-utility optimized streaming of volumetric media for augmented reality". In: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 9.1 (2019), pp. 149–162.

[13] J. Li, C. Zhang, Z. Liu, W. Sun, and Q. Li. "Joint communication and computational resource allocation for QoE-driven point cloud video streaming". In: *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE. 2020, pp. 1–6.

[14] L. Wang, C. Li, W. Dai, S. Li, J. Zou, and H. Xiong. "QoE-Driven Adaptive Streaming for Point Clouds". In: *IEEE Transactions on Multimedia* (2022).

[15] C. Loop, Q. Cai, S. Orts Escolano, and P.A. Chou. *Microsoft Voxelized Upper Bodies – A Voxelized Point Cloud Dataset*. ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document m38673/M72012.

[16] E. d'Eon, B. Harrison, T. Myers, and P. A. Chou. *"8i Voxelized Full Bodies – A Voxelized Point Cloud Dataset," ISO/IEC JTC1/SC29/WG1 input document M74006 and ISO/IEC JTC1/SC29/WG11 input document m40059, Geneva, January 2017*.

[17] Y. Xu, Y. Lu, and Z. Wen. *"Owlii Dynamic human mesh sequence dataset" ISO/IEC JTC1/SC29/WG11 m41658, 120th MPEG Meeting, Macau, October 2017*.

[18] Hao Liu, Hui Yuan, Qi Liu, Junhui Hou, and Ju Liu. "A comprehensive study and comparison of core technologies for MPEG 3-D point cloud compression". In: *IEEE Transactions on Broadcasting* 66.3 (2019), pp. 701–717.

[19] João Pedro Casanova Prazeres. *Static Point Clouds Compression efficiency of MPEG point clouds coding standards*. http://hdl.handle.net/10400.6/10880. [Online; accessed 11-December-2022]. 2020.

[20] Zhi Liu, Jie Li, Xianfu Chen, Celimuge Wu, Susumu Ishihara, and Yusheng Ji. "Fuzzy logic-based adaptive point cloud video streaming". In: *IEEE Open Journal of the Computer Society* 1 (2020), pp. 121–130.

[21] J. Van Der Hooft, T. Wauters, F. De Turck, C. Timmerer, and H. Hellwagner. "Towards 6dof http adaptive streaming through point cloud compression". In: *Proceedings of the 27th ACM International Conference on Multimedia*. 2019, pp. 2405–2413.

[22] Cheng-Hao Wu, Xiner Li, Rahul Rajesh, Wei Tsang Ooi, and Cheng-Hsin Hsu. "Dynamic 3D point cloud streaming: Distortion and concealment". In: *Proceedings of the 31st ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*. 2021, pp. 98–105.

[23] *Nginx*. https://nginx.org. Accessed: May 10, 2023.

[24] S. Perry. "JPEG Pleno point cloud coding common test conditions v3. 2". In: *87th JPEG Meeting, WG1N87037*. 2020.

[25] D. Tian, H. Ochimizu, C. Feng, R. Cohen, and A. Vetro. "Geometric distortion metrics for point cloud compression". In: *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2017, pp. 3460–3464.

[26] Q. Liu, H. Yuan, J. Hou, R. Hamzaoui, and H. Su. "Model-based joint bit allocation between geometry and color for video-based 3D point cloud compression". In: *IEEE Transactions on Multimedia* 23 (2020), pp. 3278–3291.

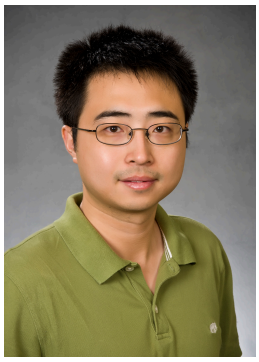
[27] L. Li, Z. Li, S. Liu, and H. Li. "Frame-level Rate Control for Geometry-based LiDAR Point Cloud Compression". In: *IEEE Transactions on Multimedia* (2022).

[28] M.-N Garcia, F. De Simone, S. Tavakoli, N. Staelens, S. Egger, K. Brunnström, and A. Raake. "Quality of experience and HTTP adaptive streaming: A review of subjective studies". In: *2014 Sixth International Workshop on Quality of Multimedia Experience (QoMEX)*. 2014, pp. 141–146. DOI: 10.1109/QoMEX.2014.6982310.

## AUTHORS

**Mahendra Suthar** earned his B.Tech. degree from the Department of Information Technology at Prasad V. Potluri Siddhartha Institute Of Technology, India, in 2020. Presently, he is enrolled in the Master of Science program in computer science at the University of Cincinnati. His primary research focus lies in the field of point cloud compression and streaming.

**Rui (April) Dai** is an associate professor in the Department of Electrical and Computer Engineering at University of Cincinnati, Ohio, USA. She received her BS in electronics and information engineering and MS in communications and information systems from Huazhong University of Science and Technology, Wuhan, China, in 2004 and 2007, respectively. She received her PhD degree in electrical and computer engineering from Georgia Institute of Technology, Atlanta, GA, USA in 2011. She was a postdoctoral fellow at the Center for Assistive Technology and Environmental Access of Georgia Tech from 2011 to 2012. She was an assistant professor at the Department of Computer Science at North Dakota State University from 2012 to 2014 and an assistant Professor in the Department of Electrical Engineering and Computer Science at University of Cincinnati from 2014 to 2020. Her current research interests include multimedia communications and networking, wireless sensor networks, and cyber-physical systems.



**Junjie Zhang** is an associate professor in the Department of Computer Science and Engineering at Wright State University, Ohio, USA. He received his Ph.D. degree in computer science from Georgia Institute of Technology in 2012, and M.S. in systems engineering and B.S. in computer science from Xi'an Jiaotong University, China, in 2006 and 2003, respectively. He was an assistant professor in the Department of Computer Science and Engineering at Wright State University from 2012 to 2018. His research focuses on building secure and trustworthy networked systems.

**Sasu Tarkoma** is Dean of the Faculty of Science at the University of Helsinki and Professor of Computer Science. He has authored 4 textbooks and published over 240 scientific articles and 11 granted US patents. His research interests are Internet technology, distributed systems, data analytics, and mobile and ubiquitous computing. His research has received several best paper awards and mentions, for example, at IEEE PerCom, ACM CCR, and ACM OSR. He is actively involved in the Helsinki Institute for Information Technology (HIIT), the Finnish Center for AI (FCAI) flagship, Helsinki Center for Data Science (HiDATA), and the Allied ICT Finland network.

**Ian F. Akyildiz** received BS, MS, and PhD degrees in electrical and computer engineering from the University of Erlangen–Nurnberg, Germany, in 1978, 1981, and 1984, respectively. Currently, he is an adjunct professor with the University of Helsinki, Finland. He is the founder and President of Truva Inc., a consulting company based in Georgia, USA, since 1989. He has also been a member of the Advisory Board at the Technology Innovation Institute

(TII) Abu Dhabi, United Arab Emirates, since June 2020. He is the founder and the Editor-in-Chief of the newly established of International Telecommunication Union Journal on Future and Evolving Technologies (ITU J-FET) since August 2020.

He served as the Ken Byers Chair Professor in Telecommunications, the Past Chair of the Telecom Group at the ECE, and the Director of the Broadband Wireless Networking Laboratory, Georgia Institute of Technology, from 1985 to 2020. He has had many international affiliations during his career and established research centers in Spain, South Africa, Finland, Saudi Arabia, Germany, Russia, India, and Cyprus. Dr. Akyildiz is an IEEE Life Fellow and an ACM Fellow. He has received numerous awards from IEEE, ACM, and other professional organizations, including the Humboldt Award from Germany. In August 2022, according to Google Scholar his H-index is 136 and the total number of citations to his articles is more than 140K+. His current research interests include 6G/7G wireless systems, terahertz communication, reconfigurable intelligent surfaces, nano-networks, metaverse and mulsemedia communication, Internet of Space Things/CUBESATs, Internet of Bio-Nano Things, molecular communication, and underwater communication.