FRAS: Federated Reinforcement Learning Empowered Adaptive Point Cloud Video Streaming

Yu Gao, Pengyuan Zhou, Member, IEEE, Zhi Liu, Senior Member, IEEE, Bo Han, Senior Member, IEEE, Pan Hui, Fellow, IEEE

Abstract—Point cloud video transmission is challenging due to high encoding/decoding complexity, high video bitrate, and low latency requirement. Consequently, conventional adaptive streaming methodologies often find themselves unsatisfactory to meet the requirements in threefold: 1) current algorithms reuse existing quality of experience (QoE) definitions while overlooking the unique features of point cloud video thus failing to provide optimal user experience, 2) most deep learning approaches require long-span data collections to learn sufficiently varied network conditions and result in long training periods and capacity occupation, 3) cloud training approaches pose privacy risks caused by leakage of user reported service usage and networking conditions.

To overcome the limitations, we present FRAS, the first federated reinforcement learning framework, to the best of our knowledge, for adaptive point cloud video streaming. We define a new QoE model which takes the unique features of point cloud video into account. Each client uses reinforcement learning (RL) to train video quality selection with the objective of optimizing the user’s QoE under multiple constraints. Then, a federated learning framework is integrated with the RL algorithm to enhance training performance with privacy preservation. Extensive simulations using real point cloud videos and network traces reveal the superiority of the proposed scheme over baseline schemes. We also implement a prototype that demonstrates the performance of FRAS via real-world tests.

I. INTRODUCTION

Volumetric video has become popular in recent years thanks to its immersive user experience with six degrees of freedom (6DoF), including the position (X, Y, Z) and the orientation (yaw, pitch, roll) of the viewer. Users can freely select any preferred viewing angle of the 3D scene to get an experience beyond 360-degree video restricted to 3DoF. In other words, unlike 360-degree video systems in which users can only switch orientation, users can freely move body position and head direction to subscribe to a field of view (FoV) at any location within the scene when watching a volumetric video.

Point cloud is arguably the most promising volumetric video format and has drawn significant attention from both academia and industry [2], [24], [30]. Delivering point cloud video requires network-friendly encoding, bandwidth-aware quality level adaption, efficient quality assessment metrics, and accurate 6DoF motion prediction [6], [28]. For instance, the required bandwidth for point cloud video streaming at 30 frames per second can be as high as 6 Gbps due to the large size of raw point cloud frames, demanding efficient transmission methods.

Encoding and compression have thus been extensively researched to ensure transmission efficiency and user experience [4], [30]. Tiling is popularly adopted as well, since a user can only watch only a portion of the scene at a time [17], [27]. For example, Park et al. [27] proposed partitioning the point cloud video into tiles spatially. Each tile is encoded with a different quality level according to its relation to the user’s view frustum and distance to the user. However, decoding point cloud video requires more computation than traditional video [17], [20], [31], posing a distinguished challenge for point cloud video systems. Several works consider using uncompressed tiles to alleviate the burden of en(de)coding. For example, Li et al. [17] considered the uncompressed tiles and optimized the user’s quality of experience (QoE) by selecting the proper quality levels under the communication and computation constraints. Liu et al. [19] proposed a fuzzy logic solution to select the quality level for each tile based on the future bandwidth, user FoV, and the available computation capability. However, these schemes require predefined models or rules and thus cannot adapt to the dynamic network conditions. Although there are adaptive streaming approaches for traditional video streaming, such a proposal for point cloud streaming is yet to be explored.

To this end, in this paper we present FRAS, a federated reinforcement learning (FRL) empowered point cloud video streaming system. The point cloud video is properly partitioned into tiles with different quality levels and different computation requirements for decoding. A novel quality evaluation metric for point cloud videos is proposed thereafter, which takes into account the features of a point cloud video system such as decoding complexity and viewer position. Then, based on the predicted network bandwidth and viewing direction and the decoding complexity, RL-empowered quality level selection helps maximize the perceived user experience under the constraints of communication resources, computation resources, and user quality requirements. To augment the training performance restricted by local datasets, FRAS applies federated learning to aggregate the learning experience of distributed clients while protecting their privacy. Extensive simulations based on real point cloud video sequences and network traces are conducted and the results reveal the superiority of the proposed scheme over baseline schemes including ViVo [17], QUETRA [35], Pensieve [21], robustMPC [37], and Buffer-Based (BB) [13].
Figure 1: An example to demonstrate the federal-reinforcement-learning-based adaptive point cloud streaming framework, i.e., FRAS: (a) Local training. Users choose the optimal video quality chosen by the RL model and update their local model. (b) Global training. Multiple users upload local gradients to a cloud server for federated model aggregation and retrieve the updated global model.

To the best of our knowledge, this is the first paper that investigates point cloud video streaming using FRL. Our contributions are summarized as follows:

- We propose FRAS, a federated reinforcement learning (FRL)-empowered adaptive point cloud streaming framework, as depicted in Figure 1. FRAS leverages distributed clients’ datasets to improve the learning performance with privacy preservation. FRAS is the first FRL-based adaptive streaming (not only for point cloud video but in general) to the best of our knowledge.
- We propose a novel QoE definition that takes unique features of point cloud video into account. For example, we consider the decoding complexity of point cloud video, which is much higher than that of regular video, in the QoE model to make up for the rebuffering caused by decoding. Furthermore, we conducted a user study to validate our model and derive the weight parameters.
- Extensive simulations based on real point cloud video sequences and network traces are conducted, and the results reveal the superiority of the proposed scheme over baseline schemes. FRAS outperforms all SOTA methods in all five concerned metrics, specifically by up to 86%, 28%, 6%, 11%, and 67% on the performances of average QoE, average quality level, average PSNR, average bandwidth, and rebuffering, respectively. We implement a prototype of FRAS and validate its performance via real-world tests.

II. RELATED WORK

Point cloud encoding and processing. A point cloud is composed of points represented in 3D space. Each point is associated with multiple attributes such as coordinates and color. There are two major classes of point cloud encoding methods according to point cloud data distribution [4], [30]. A point cloud with uniform distribution can be projected into 2D frames using well-known 2D video technologies, while sparsely distributed point cloud data can be decomposed into hierarchical cubes with each point encoded as an index of its corresponding cube. Note that the point cloud encoding has higher computation complexity than the traditional video [17], [28].

Point cloud quality assessment defines the optimization objective and is an essential component of the point cloud video transmission system. Peak signal-to-noise ratio (PSNR), which figures out the difference between the ground truth frame and the relieved frame pixel by pixel, is a common way for traditional video transmission to measure quality. For point cloud video quality assessment, researchers have proposed several metrics with similar logic. For example, [12] uses the PSNR of point-to-point distortions to measure the objective quality with the MPEG PCC [23] quality metric software. [27] introduces utility measures based on the underlying quality of the representation, the level of detail to the user’s viewpoint and device resolution. However, these schemes basically inherit the traditional video metrics and do not fully take the features of point cloud video into account.

Point cloud video streaming has recently gained popularity [6], [17]. The majority of these works use similar methods to VR video streaming methods [10], [38] that divide the videos into smaller tiles and only transmit the tiles within the user’s FoV with the goal of optimizing the defined objective function [27], [31]. [17] considers the high computation complexity of point cloud video encoding during transmission optimization. These schemes are model-based and not adaptive to the dynamic network conditions.

Adaptive bitrate streaming. RL has become popular in traditional video streaming in recent few years [21], [54]. M. Claeys, et al. [5] designed a RL-based HTTP adaptive streaming client interacting with the environment to optimize the QoE. The challenges in this direction include the proper definition of the reward and the training algorithm. Moreover, conventional local training provokes a significant limitation on the scale of the training dataset, specifically constrained to individual user devices. Since networking conditions change
from session to session, each device needs a long time to collect data and learn about different networking conditions. As pointed out by Pensieve [21], retraining frequency depends on the generalizability of the model and the frequency of new network behaviors. The authors in Oboe [1] further demonstrate the necessity of retraining for Pensieve in the face of different network conditions.

**Privacy concern.** User privacy protection is largely overlooked in previous works. The traffic metadata required for training, such as transmission rates, can seriously expose user privacy. For example, an attacker can infer user activity from traffic rate changes [3]. Furthermore, a large amount of private information such as gender, age, cognitive state or even mental disorders can be revealed from user gaze data [8], [15], [29]. Therefore, the conventional central cloud training methods that collect distributed user datasets put user privacy at risk.

To address the aforementioned shortcomings of related approaches as depicted in Figure 2, we propose FRAS, a cross-device federated reinforcement learning adaptive streaming system to augment local model training via aggregating the learning experience from distributed clients. FRAS takes the decoding complexity and features of point cloud into the account of learning policy. Moreover, unlike current schemes relying on local user data [21], [36], FRAS leverages uniquely diverse datasets from distributed users via federated learning to explore the generalizability of models extracted from global learning experience. Without requiring user data uploading, FRAS guarantees privacy and non-identifiability for the resulting models [16].

### III. FRAS System

This section introduces the tiling and quality assessment metric of point cloud videos.

#### A. System overview

FRAS is a video-on-demand system that adaptively delivers point cloud streaming, thus providing consumers with a greater quality of experience. As illustrated in Figure 3, the point cloud streaming is delivered to the server through a wired network and tiled. The partitioned tiles are sampled to different qualities and compressed. Then, the server uses all its tiles (both compressed and uncompressed) to generate an MPD file for each video and sends it to the client. Each client uses the RL-based rate adaptation to select tiles ID and quality level based on environment information such as user’s 6DoF pose, network condition, buffer state, and tile information obtained from the MPD file. Thereafter, the clients get the demanded tiles through the network interface. Meanwhile, multiple clients aggregate gradients via the aggregation server through the FRL algorithm during the training phase.

Because the bandwidth between video source and server is usually sufficient, we focus on the transmission between server and client which has limited and dynamic bandwidth $B_{i,t}$ for client $i$ at time point $t$. Note that $t$ is the basic time unit during which the frames are encoded together as a group, similarly to the group of pictures (GOP) in traditional video streaming. $B_{i,t}$ is predicted by employed bandwidth prediction methodology as explained in Section III-C. Consider client $i$ has a FoV $F_{i,t}$ at time point $t$ demanding video quality not lower than $Q_i$. User $i$ has $C_{i,t}$ available computation capacity to decode the compressed tiles. The client makes the decisions using RL based on the 6DoF pose, bandwidth, playback buffer, and the decoding process complexity according to the predefined reward policy to optimize QoE.

#### B. Tiling and Downsampling

To guarantee the smooth viewport content switching and high video quality within the client’s FoV, the point cloud video is partitioned into homogeneous tiles. For computation and communication efficiency, we transfer the tiles within FoV with the encoding rate selected by our FRL algorithm while encoding and transmitting the tiles outside FoV with the lowest quality. First, we recognize the best-fit cuboid surrounding the interested point cloud space and identify its pose and 3D dimension. Then, we perform $N \times M$ partitioning on the plane perpendicular to the height and divide them into $H$ layers in the height direction, and finally, $N \times M \times H$ tiles are obtained as shown in Figure 4. The partitioned tiles at $t$ are further uniformly downsampled into $L$ quality levels. In this work, we downsample the tiles into 5 levels by 20%, 40%, 60%, 80%, and 100%. A compressed tile $\{n,m,h,l,t\}$ has data size of $s_{n,m,h,l,t}$, point-to-point PSNR $q_{n,m,h,l,t}$, and required computation resource $d_{n,m,h,l,t}$ for decoding. A higher quality level has a larger data size and requires more computation resources for encoding and decoding.

We consider that the server also reserves uncompressed tiles at all quality levels. An uncompressed tile $\{n,m,h,l,t\}$ is with data size $s'_{n,m,h,l,t}$, point-to-point PSNR $q'_{n,m,h,l,t}$, and does not require computational capacity for decoding. We consider lossless compression in this work to avoid lossy compression's impact on depth data [32] and thus the uncompressed tiles have the same PSNR values as the compressed ones encoded at the same quality levels. The uncompressed tiles are at a larger size, i.e., $s'_{n,m,h,l,t} > s_{n,m,h,l,t}$. Transmitting the uncompressed tiles allows for lowering decoding workload on the client side at the cost of more bandwidth.

#### C. FoV and Network Prediction

While the 360° video has only three degrees of freedom in terms of orientation (yaw, pitch, and roll), the point cloud
Figure 3: FRAS system architecture

Figure 4: An example of tiling and two sampling qualities.

Network bandwidth prediction is also significant in point cloud video streaming. In this work, we compare two methods for bandwidth prediction: gated recurrent unit (GRU) [9] and exponentially weighted moving average (EWMA) [14]. GRU is a popular artificial recurrent neural network architecture widely used for time-series forecasting. It is similar to long short-term memory (LSTM) but with a forget gate, and can achieve slightly better performance with much lower complexity in comparison with LSTM. EWMA is a simple and powerful statistical tool for time series modeling.

D. Constraints

As mentioned, FRAS fetches all tiles at selective compression rates except those outside FoV which are transmitted without compression. We first define two notations to facilitate the description of the quality level selection. \( I_{n,m,h,l,t} \) denotes the compressed tile \((n,m,h)\) at quality level \(l\) during time period \(t\), where \( I \in \{0,1\} \) represents the decision of transmission, 1 transmitted and 0 otherwise. Similarly, \( I'_{n,m,h,l,t} \) denotes the uncompressed tile \((n,m,h)\) at quality level \(l\) during \(t\). We briefly list the constraints as follows.

First, each transmitted tile is either compressed or uncompressed:

\[
\sum_{l=1}^{L} I_{n,m,h,l,t} + I'_{n,m,h,l,t} = 1, \forall n,m,h,t
\]  

Second, at any time point \(t\), the overall data volume of the transmitted tiles cannot exceed the bandwidth:

\[
\sum_{n=1}^{N} \sum_{m=1}^{M} \sum_{h=1}^{H} I_{n,m,h,l,t} \times s_{n,m,h,l,t} + I'_{n,m,h,l,t} \times s'_{n,m,h,l,t} \leq B_{i,t}
\]  

Third, the required decoding resources for all transmitted tiles cannot exceed the capacity budget:

\[
\sum_{n=1}^{N} \sum_{m=1}^{M} \sum_{h=1}^{H} I_{n,m,h,l,t} \times d_{n,m,h,l,t} \leq C_{i,t}
\]  

IV. FRL-EMPOWERED TRANSMISSION

This section introduces the adopted federated reinforcement learning (FRL) for adaptively selecting the proper tile quality level to maximize QoE under the communication and computation constraints.

A. Challenges

As the first look into this problem, we summarize the challenges as follows.

1) The decoding complexity is a unique feature of point cloud video compared to conventional video. For instance, decoding in point cloud video has longer latency than that in conventional video, and thus can affect the watching experience. We need to convert it to be comparable with other metrics to integrate it into the QoE calculation.

2) Another unique feature of point cloud video, 6DoF, enables clients to move positions. As a client would expect different frame qualities from different distances, the distance to the scene should also be considered.
3) Each round of federated learning consists of a number of local training epochs followed by the aggregation of clients’ model updates. Therefore, a proper number of local training epochs is critical as too many would result in slow global model convergence while too few would lead to slow local model convergence and high communication cost.

4) Lastly, though network prediction is not the focus of this work, it is an interesting additional functionality to improve ABR in general. Hence, it is important to ensure its processing delay does not impact the streaming service.

B. QoE Modeling Through User Study

We mainly consider the point-to-point PSNR, encoding quality level, and distance from the scene for the quality metric. When a client moves closer to the scene, users will feel more sensitive to each tile’s PSNR and its quality level. So the weights of quality components should vary with distance. Thus, for each client $i$ with FoV $F_{i,t}$ at time $t$, the quality of its received video is defined as follows:

$$Q_{dis} = \alpha \sum_{n,m,h,l,t} q_{n,m,h,l,t} + \beta \ast l$$

(4)

which indicates the weighted sum of the encoding quality level and the sum of the point-to-point PSNR values of the tiles in the FoV, taking the user’s distance into account. Next, the optimization goal also considers the rebuffing time $T_r$ and quality smoothness $\Delta l$ as in related works. Moreover, as mentioned before, we take the decoding complexity of point cloud video into account. Since QoE already contains quality metrics and time metrics, adding a third-dimension metric would further increase its complexity and the training difficulty. Hence, instead of directly using the required decoding capacity, we include a decoding time penalty $T_d$ in the QoE, which refers to the actual decoding time subtracting the chunk duration time. We conducted numerous decoding tests across different point cloud videos of varied qualities and used the average values in the algorithm. Notably, the decoding time of point cloud video tile is much larger than that of conventional video which normally is considered an instantaneously part within $T_r$, as proved by our decoding tests. As such, the QoE of point cloud video streaming can be formulated as follows.

$$QoE_{dis} = f(Q, T_r, \Delta l, T_d)$$

$$= \alpha \ast \sum_{n,m,h,l,t} q_{n,m,h,l,t} + \beta \ast l - \gamma \ast T_r - \delta \ast \Delta l - \varepsilon \ast T_d$$

(5)

where $\alpha$, $\beta$, $\gamma$, $\delta$, and $\varepsilon$ are weight parameters. Note that Our QoE model varies when the user’s distance to the FoV changes. Therefore, we derive groups of weight values according to different distances, as explained later, according to our user study results. The problem is to maximize the average QoE. As such, we solve challenge 1 and challenge 2.

Next, we conduct a user study to validate our defined QoE model and derive the weight parameters. Our approach to studying the impact of all factors in the QoE model mentioned in Equation 5 is to have each participant compare the videos impaired by various combinations of factors with a high-quality “perfect” version ($Q = 5$, $T_r = \Delta l = 0$) in each distance and provide a subjective rating. We choose the Longdress from the 8i Labs point cloud datasets for this study, which shows a dancing woman and lasts for 10 seconds. Table I shows the factors and their possible values for our QoE model training. There are 108 combinations of video pairs, with the impaired version on the left and the perfect version on the right. Then we ask each participant to randomly select 36 video pairs from the 108 combinations for a subjective experience comparison. After watching each pair, the participant rates which QoE the impaired video provides compared to the perfect video through 4 choices: similar to, slightly worse, worse, much worse. Then the impaired video’s QoE is labeled with a score of $(3,2,1,0)$ respectively.

We collected 1080 ratings from 30 participants through a survey and then used the ratings to calculate the QoE parameters in Equation 5 for each distance (1 m, 2 m, 3 m) separately through linear regression. The results are shown in Table II. Then we used 10-fold cross-validation to validate our QoE model and the mean absolute error (MAE) of the prediction results is shown in Figure 5. The average MAE at 1m, 2m, 3m is 0.71, 0.59, and 0.68, respectively.

$$\text{Distance} \quad \alpha \quad \beta \quad \gamma \quad \delta \quad \varepsilon$$

| Distance | $\alpha$ | $\beta$ | $\gamma$ | $\delta$ | $\varepsilon$ |
|----------|---------|---------|---------|---------|---------|
| 1 m      | 0.11    | 0.61    | 12.58   | -0.13   | 12.58   |
| 2 m      | 0.05    | 0.12    | 12.68   | -0.01   | 12.68   |
| 3 m      | 0.04    | 0.10    | 13.29   | -0.05   | 13.29   |

Table II: QoE model weight values used in FRAS.

C. FRL Algorithms

We train local models using RL algorithm to optimize each individual client’s QoE following the objective function defined in subsection IV-B. As mentioned, each client encounters limited network conditions in short periods and thus needs long-span data collection and training to get a generalized and robust
model, which impacts the QoE of the client and the battery life of the device. Therefore, we integrate FL and RL to aggregate the learning experiences of users for data augmentation and training acceleration under privacy preservation. The major details are listed as follows.

**State:** The RL agent takes \( s_t = (l_{t-1}, b_t, d_t, n_t, r_t) \) as the input, where \( l_{t-1} \) is the encoding buffer level of the last point cloud video chunk (uncompressed tiles have different level indices than compressed tiles to differentiate); \( b_t \) is the current buffer level; \( d_t \) is the predicted network bandwidth; \( n_t \) is the download time of the current chunk; \( r_t \) is a vector of \( L \) possible sizes for the next video chunk; \( r_t \) is the number of remained chunks in the video.

**Action:** The RL agent takes an action by selecting the encoding quality level for the next point cloud video chunk upon receiving \( s_t \). The actions are selected based on a policy \( \pi(\cdot) \rightarrow [0, 1] \)

**Local training:** We employ actor-critic algorithm to train the local policy. The reward is defined according to QoE (sub-section IV-B) as shown in Equation (6). The gradient of the cumulative discounted reward with respect to the policy is computed as Equation (7) [25].

\[
\begin{align*}
\nabla^\theta \mathbb{E}_{\pi_\theta} \left[ \sum_{t=0}^{\infty} \gamma^t r_t \right] &= \mathbb{E}_{\pi_\theta} \left[ \nabla^\theta \log \pi_\theta(s, a) A^\pi_\theta(s, a) \right] \\
&= \mathbb{E}_{\pi_\theta} \left[ \nabla^\theta \mathbb{E}_{\pi_\theta} \left[ \sum_{t=0}^{\infty} \gamma^t r_t \right] \right]
\end{align*}
\]

**Global aggregation:** We apply FedAvg [22] in the global aggregation stage. After \( E \) local epochs, \( m \) of \( K \) clients return their local gradients to the aggregation server, where \( m \) equals to \( \max(\mu \cdot K, 1) \) and \( \mu \) is a pre-defined participation ratio. Then the global gradients are calculated according to Equation (8) and used by all the \( K \) clients to update local models.

\[
\theta_i^G = \sum_{k=1}^{\mu\cdot K} \frac{w_k \cdot \theta_i^k}{\mu\cdot K}
\]

**Algorithm 1:** FRAS

1 **Procedure FL:**
   **Input:** Server actor network weight \( \theta_0^G \) and critic network weight \( \phi_0^G \), users \( \{c_1, c_2, \ldots, c_K\} \), local update step \( E \)
   **for** \( T = 1 \) to \( T_{\text{max}} \) **do**
   2 Select a number of users \( C = \{C_1^T, \ldots, C_T\} \)
   3 \( u_t^C, v_t^C \leftarrow \text{RL}(\theta_t^G, \phi_t^G) \)
   4 \( u_t^G \leftarrow \text{FedAvg}(u_t^C) \)
   5 \( u_{t+1}^G \leftarrow \text{FedAvg}(u_{t+1}^C) \)
   6 Update global model \( \theta_{t+1}^G \leftarrow \theta_t^G + u_{t+1}^G \)
   **end for**

2 **Procedure RL:**
   **Input:** \( C = \{C_1^1, \ldots, C_K^T\} \), global model \( \theta_t^G, \phi_t^G \)
   **for** each user \( c \in C \) **do**
   9 Reset \( d\theta \) and \( d\phi \) to 0, \( \theta = \theta^G, \phi = \phi^G \)
   10 for \( t = t_{\text{start}} \) to \( t_{\text{start}} + E - 1 \) **do**
   11 Feed \( V(c) \) into RL model to predict \( a_t^* \) and \( s_t^* \)
   12 for all clients \( c_i \) **do**
   13 Feed \( s_t^* \) into RL model to predict \( a_t^* \) and \( s_t^* \)
   14 \( r_t = f(V(c_i)) \)
   15 \( s_t = (l_{t-1}, b_t, d_t, n_t, r_t) \)
   16 Get policy \( \pi(a_t|s_t; \theta^G) \)
   17 Update reward \( r_t \) and new state \( s_{t+1} \)
   **end for**
   **end for**
   **end for**

3 **Procedure Prediction:**
   \( \text{FoV} \leftarrow \text{FoV predict}(c) \)
   \( B \leftarrow \text{BW predict}(c) \)
   **Return** \( c_{\text{FoV}}, c_B \)

Note that the integration of FL and RL is not straightforward and requires fine-tuning efforts. As stated in **challenge 3** the \( E \) local epochs of RL training affect the trade-off between transmission cost and convergence rate of local and global models. We found the optimal \( E \) for different data distributions via empirical tests. For **challenge 4** to enable real-time network prediction, we customize the input/output period lengths to decrease the processing delay to a level that can be transparent to the user while still facilitating RL training and inference. Please refer to Section V for the details. Another ad-hoc configuration is the number of workers ran by each client’s RL algorithm. Depending on the available capacity, each RL algorithm can run a varied number of workers in parallel using an asynchronous actor-critic algorithm (A3C [25]), or, just one single worker using synchronous update (A2C [33]). The performance of both methods is comparable, as found by the empirical study [35]. A few newer RL algorithms can also be applied in this context, which is not our focus and thus left out of this work.
V. Evaluation

A. Experimental Setup

Video source. We use 36 point cloud video sequences from Panoptic Studi[1] for training. We use another 6 point cloud video sequences from vsenseVDB2 database [7] i.e., AxeGuy, LubnaFriends, Rafa2, Matis, Loot and Longdress, for testing. Each point cloud frame is partitioned into 3 × 3 × 4 tiles and then sampled at (20%, 40%, 60%, 80%, 100%), corresponding to quality level from 1 to 5. We use the KD-tree based Google Draco codec [10] for encoding and decoding. We use the Draco rather than standardized MPEG’s V-PCC encoder [3] because it has better performance and can allow real-time decoding on contemporary hardware.

Network conditions. Since the point cloud video has high bitrates even after compression, we must use high-bandwidth traces to emulate realistic throughput fluctuations in the experiments. Thus, we replay the bandwidth traces from a commercial mmWave 5G throughput dataset provided by Lumos5G [24] to simulate the network. The dataset contains 118 traces under two different mobility modes: walking and driving, covering over 18 hours of active monitoring. During the test, the algorithm uses predicted bandwidth as described in subsection III-C to select the encoding quality for the next video chunk. Then the algorithm uses the selected quality level and predicted bandwidth together with several other state metrics to calculate the reward.

FoV. In this work, we use the viewport trajectory datasets provided by ViVo [11], which provide the viewport trajectory of 32 users (including smartphone users and headset users) in four volumetric videos. The datasets have position dimensions (x, y, z) and orientation dimensions (yaw, pitch, roll) in each item and have over 10,000 items overall. As mentioned in subsection III-C, we use three approaches to separately predict each dimension: 1) last FoV (LAST): use the FoV of the previous moment as the current FoV; 2) LR, MLP: use LR or MLP to predict the current FoV from a history window consisting of last 8 viewports. The lightweight neural network we use in MLP has two hidden layers with 8 and 2 neurons, respectively. The activation function of MLP is rectified linear unit (ReLU), and the solver for weight optimization is Limited-memory BFGS (L-BFGS) [18]. Additionally, LR and MLP are both implemented by the scikit-learn library [5].

Models and Metrics. FRAS passes 12 past bandwidth measurements to a 1D convolution layer with 128 filters, each of size 4 with stride 1. Possible next chunk sizes are input to another 1D-CNN with the same shape. Results from these layers are then aggregated in a 128-neuron hidden layer to apply the softmax function. The critic network uses the same layers except that a linear neuron is used as the final output. We apply the softmax function. The critic network uses the same layers except that a linear neuron is used as the final output. We set the decay from 0.1 over 3 × 10^5 iterations. We implemented this architecture using TensorFlow 2.3.0. Please refer to Table II for the QoE model weight values we used.

Baselines. We implement 5 baselines for performance comparison. (1) ViVo [11] performs a first comprehensive study of mobile volumetric video streaming, which considers the 6DoF feature of volumetric video and optimize the streaming from three visibility-aware approaches. (2) QUETRA [35] is a DASH rate adaptation algorithm which calculates the expected buffer occupancy using selected bitrate, network throughput, and buffer capacity based on queuing model. (3) An RL method adapted from Pensieve [21] which trains a single RL model. (4) robustMPC [37] improves MPC by accounting for errors of predicted throughput via normalization. (5) Buffer-Based (BB) [13] selects the encoding quality level based on the current buffer occupancy and estimated occupancy during the startup phase with the goal of keeping the occupancy above 0.1 second, and automatically chooses the highest level if the occupancy exceeds 1 second.

Implementation. We build a point cloud visualizer using PCL 1.9.1 and QT 5.12 to support each of the aforementioned baselines. The visualizer was configured to fetch quality selection decisions from an ABR client that implemented the corresponding algorithm. The player was configured to have a playback buffer capacity of 5000 ms. Each sequence has 300 frames, or 10 seconds of video, and will be played in a loop to present a more realistic playback scenario. Furthermore, we set the GOP to 10 in experiments, i.e., each chunk has approximately 330 ms duration in order to avoid sending a large number of requests in a very short time. The video player and the ABR server run on the same server as the client, which is equipped with an Intel Xeon Gold 6246R CPU, NVIDIA GeForce GTX 3090 GPU and 480G SSD. And an HTTP server was deployed with Python HTTP module, which stored tiled point cloud videos and corresponding MPD files.

B. Result and Analysis

FOV prediction. Firstly, note that we only show the prediction results for the (X, Z) dimensions of the position and the (Yaw, Pitch) dimensions of the orientation, because users rarely move upward or downward (Y dimension), or rotate about the longitudinal axis (roll dimension), in the datasets we use as mentioned by ViVo [11]. Figure 6 shows the position MAE (6a) and orientation MAE (6b) of the three prediction methods described in subsection III-C. As shown, in every dimension, LR performs slightly better than MLP, and they both perform much better than LAST. Since LR has the best prediction accuracy while requires less computation than MLP, we choose to use LR in FRAS for FoV prediction.

Network prediction. As mentioned in subsection III-C, we tested two bandwidth prediction algorithms: GRU and EWMA. Figure 8 shows their performances in two networking conditions, namely, while walking and driving. As shown, the EWMA-based method has a better performance. Besides, EWMA has a lower computational complexity than GRU. Therefore, we chose an EWMA-based method to predict the network bandwidth for better network prediction performance.
in FRAS. Specifically, we let the algorithm learn the bandwidth of the last 30 seconds to predict the next second.

**Training.** We first validate the core algorithm FRL by comparing its training performance with the RL algorithm adapted from Pensieve. Figure 7a and Figure 7b show the reward and loss of both algorithms during training. The curves of FRL represent the corresponding values of the global FL model while the curves of RL represent the corresponding values of an individual client model. The converged FRL outperforms Pensieve on reward by 9.77% and loss by 49.81%.

As mentioned in Section IV, the number of local RL epochs, \( E \), before each FL aggregation affects the performance tradeoff. We varied the data distribution heterogeneity across the clients and observed a pattern between the heterogeneity and optimal \( E \) value. As shown in Figure 9, as the heterogeneity increases, the algorithm requires a larger \( E \) for good performance, which is reasonable since heterogeneous data distribution demands more local training to favor each individual dataset’s pattern.

**Test.** We validate the performance of FRL in comparison with the baselines using bandwidth traces collected in different networking conditions. As shown in Figure 10, FRL significantly outperforms the baselines in all metrics under most scenarios. Specifically, FRL increases the average QoE (reward) by 86%, 76%, 68%, 25% and 22% compared to QUETRA, robustMPC, BB, Pensieve, and ViVo, respectively. Moreover, FRL increases the average video quality level by 22%, 16%, 28%, 5% and 14%, average PSNR by 5%, 3%, 6%, 1% and 4%, bandwidth usage by 8%, 4%, 11%, 1% and 4%, and rebuffer time by 67%, 56%, 33% and 11% compared to QUETRA, robustMPC, BB and ViVo respectively.
Note that we have overall 5 encoding quality levels (each level has a compressed and uncompressed version). Therefore, the improvements over baselines can be approximately seen as improving the average quality level, which aligns with the improvement results of PSNR. For the same reason, FRL presents higher bandwidth usage than the baselines. However, FRL presents the minimal rebuffering of all tested algorithms, which indicates FRL causes the lowest frequency of bandwidth saturation. Hence, it proves that FRL can best utilize the available bandwidth to achieve the best performance trade-off and QoE.

**Demo.** We ran real-world tests for all the algorithms on the prototype system with point cloud videos on real wireless networks. Figure 12 shows a sequence of point cloud video frames sampled approximately every two seconds. FRAS shows the best frame quality out of all the streaming algorithms, which validates the superiority of FRL and the practicability of FRAS.

**VI. Conclusion**

Point cloud streaming plays a key role in multimedia, especially for volumetric videos. Its unique features, such as 6DoF and decoding complexity, demand innovative QoE
definition and ABR algorithms. In this work, we propose FRAS, the first federated reinforcement learning framework, to the best of our knowledge, for adaptive point cloud video streaming. FRAS is an end-to-end framework that takes the unique features of the point cloud into account and augments learning performance by aggregating distributed modes of users but with privacy preservation. Empirical evaluations and prototype demo have shown the superior performance of FRAS in numerous perspectives.

REFERENCES

[1] Z. Akhtar, Y. S. Nam, R. Govindan, S. Rao, J. Chen, E. Katz-Bassett, B. Ribeiro, J. Zhan, and H. Zhang. Oboe: auto-tuning video abr algorithms to network conditions. In SIGCOMM, 2018.
[2] E. Alexiou, E. Upenik, and T. Ebrahimi. Towards subjective quality assessment of point cloud imaging in augmented reality. In 2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSpP), pp. 1–6. IEEE, 2017.
[3] N. Apostolopoulou, D. Reisman, S. Sundareshan, A. Narayanan, and N. Feamster. Spying on the smart home: Privacy attacks and defenses on encrypted IoT traffic. arXiv preprint arXiv:1708.05944, 2017.
[4] C. Cao, M. Preda, and T. Zaharia. 3d point cloud compression: A survey. In The 24th International Conference on 3D Web Technology, pp. 1–9. ACM, 2019.
[5] M. Cleasys, S. Latre, J. Famaey, and F. De Turck. Design and evaluation of a self-learning http adaptive video streaming client. IEEE communications letters, 18(4):716–719, 2014.
[6] A. Clemm, M. T. Vega, H. K. Ravuri, T. Wauters, and F. De Turck. Toward truly immersive holographic-type communication: Challenges and solutions. IEEE Communications Magazine, 58(1):93–99, 2020.
[7] E. d’Eon, B. Harrison, T. Myers, and P. Chou. Xi voxelize full bodies-a voxelized point cloud dataset. ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document WG11M40059/WG1M74006, Geneva, 2017.
[8] M. Faber, R. Bixler, and S. K. D’Mello. An automated behavioral measure of mind wandering during computerized reading. Behavior Research Methods, 50(1):134–150, 2018.
[9] R. Fu, Z. Zhang, and L. Li. Using istm and gru neural network methods for traffic flow prediction. In 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), pp. 324–328. IEEE, 2016.
[10] C. Guo, Y. Cui, and Z. Liu. Optimal multicast of tiled 360 vr video in odma systems. IEEE Communications Letters, 22(12):2563–2566, 2018.
[11] B. Han, Y. Liu, and F. Qian. Vivo: Visibility-aware mobile volumetric video streaming. In Proceedings of the 26th annual international conference on mobile computing and networking, pp. 1–13, 2020.
[12] M. Hosseini and C. Timmerer. Dynamic adaptive point cloud streaming. In Proceedings of the 23rd Packet Video Workshop, pp. 25–30. ACM, 2018.
[13] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson. A buffer-based approach to rate adaptation: Evidence from a large video streaming service. In Proceedings of the 2014 ACM conference on SIGCOMM, pp. 187–198, 2014.
[14] J. S. Hunter. The exponentially weighted moving average. Journal of quality technology, 18(4):203–210, 1986.
[15] J. T. Hutton, J. Nagel, and R. B. Loewenson. Eye tracking dysfunction in alzheimer-type dementia. Neurology, 34(1):99–99, 1984.
[16] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al. Advances and open problems in federated learning. arXiv preprint arXiv:1912.04977, 2019.
[17] J. Li, C. Zhang, Z. Liu, W. Sun, and Q. Li. Joint communication and computational resource allocation for qoe-driven point cloud video streaming. IEEE International Conference on Communications (ICC), 2020.
[18] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. Mathematical programming, 45(1):503–528, 1989.
[19] Z. Liu, J. Li, X. Chen, C. Wu, S. Ishihara, and Y. Ji. Fuzzy logic-based adaptive point cloud video streaming. IEEE Open Journal of the Computer Society, 1:121–130, 2020.
[20] Z. Liu, Q. Li, X. Chen, C. Wu, S. Ishihara, J. Li, and Y. Ji. Point cloud video streaming: Challenges and solutions. IEEE Network, 35(5):202–209, 2021.
[21] H. Mao, R. Netravali, and M. Alizadeh. Neural adaptive video streaming with pensieve. In Proceedings of the Conference of the ACM Special Interest Group on Data Communication, pp. 197–210, 2017.
[22] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In Artificial intelligence and statistics, pp. 1273–1282. PMLR, 2017.
[23] R. Mekuria and L. Bivolarovsky. Overview of the mpeg activity on point cloud compression. In 2016 Data Compression Conference (DCC), pp. 620–620. IEEE, 2016.
[24] R. Mekuria, K. Blom, and P. Cesar. Design, implementation, and evaluation of a point cloud codec for tele-immersive video. IEEE Transactions on Circuits and Systems for Video Technology, 27(4):828–842, 2016.
[25] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In International conference on machine learning, pp. 1928–1937. PMLR, 2016.
[26] A. Narayanan, E. Ramadan, R. Mehta, X. Hu, Q. Liu, R. A. K. Fezeu, U. K. Dayalan, S. Verma, P. Ji, T. Li, F. Qian, and Z.-L. Zhang. Lamos5g: Mapping and predicting commercial mmwave 5g throughput. In Proceedings of the ACM Internet Measurement Conference, IMC ’20, p. 176–193. Association for Computing Machinery, New York, NY, USA, 2020. doi: 10.1145/3419394.3423629
[27] J. Park, P. A. Chou, and J.-N. Hwang. Rate-utility optimized streaming of volumetric media for augmented reality. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 9(1):149–162, 2019.
[28] F. Qian, B. Han, J. Pair, and V. Gopalakrishnan. Toward practical volumetric video streaming on commodity smartphones. In Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications, pp. 135–140. ACM, 2019.
[29] N. Sammaknejad, H. Pouretemad, C. Eslahchi, A. Salahiard, and A. Alinejad. Gender classification based on eye movements: A processing effect during passive face viewing. Advances in cognitive psychology, 13(3):232, 2017.
[30] S. Schwarz, M. Preda, V. Baroncini, M. Budagavi, P. Cesar, P. A. Chou, R. A. Cohen, M. Krivouka, S. Lasserre, Z. Li, et al. Emerging mpeg standards for point cloud compression. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 9(1):133–148, 2018.
[31] 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, pp. 2405–2413, 2019.
[32] A. D. Wilson. Fast lossless depth image compression. In Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces, pp. 100–105, 2017.
[33] Y. Wu, E. Mansimov, R. B. Grosse, S. Liao, and J. Ba. Scalable trust-region method for deep reinforcement learning using kroncker-factorized approximation. Advances in neural information processing systems, 30, 2017.
[34] W. Xue, J. Fan, V. G. Lopez, J. Li, Y. Jiang, T. Chai, and F. L. Lewis. New methods for optimal operational control of industrial processes using reinforcement learning on two time-scales. IEEE Transactions on Industrial Informatics, 2019.
[35] P. K. Yadav, A. Shafiei, and W. T. Ooi. Quetra: A queuing theory approximated point cloud video streaming service. In Proceedings of the 25th ACM international conference on Multimedia, pp. 1130–1138, 2017.
[36] H. Yeo, Y. Jung, J. Kim, J. Shin, and D. Han. Neural adaptive content-aware internet video delivery. In OSR, 2018.
[37] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli. A control-theoretic approach for dynamic adaptive video streaming over http. In Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, pp. 325–338, 2015.
[38] M. Zink, R. Sitaraman, and K. Nahrstedt. Scalable 360° video stream delivery: Challenges, solutions, and opportunities. Proceedings of the IEEE, 2019.

C. Guo, Y. Cui, and Z. Liu. Optimal multicast of tiled 360 vr video in odma systems. IEEE Communications Letters, 22(12):2563–2566, 2018.