©2023 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

A Unified Resource Allocation Framework for Virtual Reality Streaming over Wireless Networks

Nguyen Quang Hieu, Nam H. Chu, Dinh Thai Hoang, Diep N. Nguyen, and Eryk Dutkiewicz
School of Electrical and Data Engineering, University of Technology Sydney, Australia

***Abstract*—Although Rate Splitting Multiple Access (RSMA) is a promising scheme to effectively manage interference and enhance data rate and spectral utilization, its applications for Virtual Reality (VR) streaming have not been well studied. In addition to the strict latency requirement as in conventional High-Definition streaming, VR streaming further requires more computing resources at the transmitter to promptly react to the dynamic of users' Field-of-View interests. Unfortunately, current conventional RSMA approaches could not effectively handle these problems since they are not intentionally developed to deal with the special features of VR streaming. To address these challenges, we first propose a novel hierarchical multicast technique to effectively integrate the RSMA and VR streaming by exploiting the Field-of-Views of VR users. Then, the VR streaming problem established based on RSMA is formulated as a joint computation and communication optimization problem which can not only guarantee VR streaming latency requirement but also effectively manage interferences among users. Finally, due to the dynamic and uncertainty of wireless channels and users' demands, we develop a deep reinforcement learning approach to find the optimal policy for the system. This learning solution allows us to find the optimal parameters for the system via trail-and-error learning process, and thus it is effective in dealing with the uncertainty and unknown information from surrounding environment. Simulation results demonstrate that our proposed solution can satisfy the VR requirement of millisecond latency that is much lower than those of the baselines.**

***Index Terms*—VR streaming, RSMA, deep reinforcement learning, Proximal Policy Optimization, Metaverse.**

I. Introduction

Recent development of user-centric applications such as augmented, mixed, and virtual reality (VR) streaming/gaming have tailored the next generation network which is expected to provide high-rate and adaptive services for users [1]. Unlike conventional High-Definition (HD) video streaming applications over wireless networks, e.g., Netflix or Youtube, VR applications require not only tremendous data rate traffic but also the personalized experience for VR users with adaptive VR streaming facilities. For example, a 360^0 video can be divided into different tiles and transmitted to the users. This unique feature enables a transmission mechanism in which only desired tiles will be transmitted to users to fit their Field-of-Views (FoVs). As a result, tile transmission can reduce the wireless bandwidth usage of such 360^0 videos streaming. Nevertheless, VR streaming also brings new challenges compared with conventional HD video streaming.

The first challenge due to the diversity of VR devices, e.g., Head Mounted Display (HMD), and thus managing interferences among users is a very challenging task, especially in an area with high number of users equipped with various types of VR devices. Most existing works leverage the conventional multiple access to mitigate interference between users of VR streaming applications [3], [4]. However, their approaches are not effective in practice due to the assumption of low interference level at the users [3] or relying on a simple time slot-based multiple access scheme (without carefully considering QoE for different users) [4]. Recently, Rate Splitting Multiple Access (RSMA) has been emerging as an efficient technique to manage interference by employing the Successive Interference Cancellation (SIC) technique at the receivers and rate splitting scheme at the transmitter. As a result, RSMA can improve the data rate, Quality-of-Service (QoS), spectral and energy efficiency, and reliability compared to those of conventional multiple access schemes, e.g., Non-Orthogonal Multiple Access (NOMA) or Spatial Division Multiple Access (SDMA) [5], [6]. In [7], the authors propose a Hierarchical Rate Splitting Multiple Access (HRSMA) to further boost the users' achievable rate and enhance the robustness under the imperfect channel state information (CSI) at the BS. The idea of HRSMA is to divide users into different groups, and then users' messages are split into super common, group common, and private streams. By doing so, the inter-group interference due to channel overlaps and the intra-group interference due to imperfect CSI are alleviated.

Recently, there are several research works, e.g., [8] and [9], further exploring the performance of multi-group RSMA in different applications, e.g., satellite communications and multi-cell networks. However, current user clustering approaches in the above works are based on users' location or wireless channel, and thus they are not effective for the VR streaming applications [10]. Instead, clustering methods based on users' FoVs can benefit the spectral efficiency and latency by only sending a set of tiles from the 360^0 video frames [4], [11]. Moreover, it can be observed that above works (i.e., [7]–[9]) only focus on maximizing the data transmission rate by precoder design and power allocation strategies, which cannot guarantee a low latency requirement of VR application since the latency also depends on the computation processes, e.g., pre-rendering FoVs. Thus, the potential convergence of VR and RSMA remains unexplored.

The second main challenge is that the VR transmitter, e.g., a base station (BS), needs to quickly adapt to the changes of users' FoVs so that it can not only guarantee Quality-of-Experience (QoE) for the users but also better utilize the radio resources. A promising solution is to pre-render 360^0

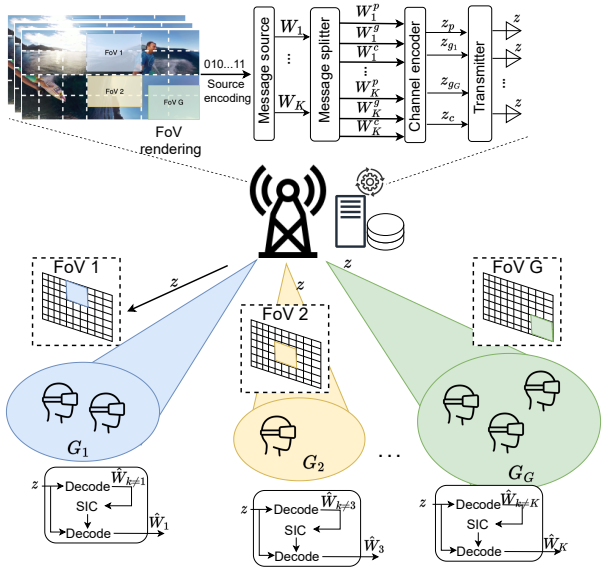

Fig. 1: Our proposed system model.

videos at the BS so that the system can quickly respond to the change of users' FoVs as well as achieve spectral efficiency via the reduction of video data transmitted over networks. Nevertheless, pre-rendering FoV is an expensive computing task at the BS. Thus, allocating resources of the BS to pre-render users' FoVs plays a key role in the system. In practice, this is a very challenging task due to the dynamic of the processes running at the BS and the dynamic nature of VR environment [2]. Moreover, under HRSMA scheme, the BS also needs to allocate computing resources for processing users' data streams and radio resources for each user's stream. However, in the literature, the computing resource allocation approach capturing the dynamic of VR streaming under RSMA has yet not been well studied.

To address the above challenges, we propose a novel framework that can effectively manage interference to facilitate VR streaming under RSMA communications. In particular, we first introduce a novel user clustering method based on users' FoVs for RSMA. By doing so, our framework can take advantages of both HRSMA and FoV pre-rendering to effectively manage not only interference but also computing resource for RSMA-based VR streaming applications. To enhance users' QoE, we formulate the HRSMA-based VR streaming problem as a joint communication and computation optimization problem with the objective of minimizing system latency. We then develop a deep reinforcement learning algorithm based on Proximal Policy Optimization (PPO) to address the high complexity of the optimization problem as well as the uncertainty and dynamic of wireless VR streaming, e.g., channel state information and users' demands. The simulation results show that our proposed approach can achieve a latency of 61 ms, which can satisfy the latency requirement of VR streaming.

II. System Model

We illustrate our proposed system model in Fig. 1. In particular, our system model consists of one M-antenna base station (BS) and K VR users. The users are divided into G different groups based on their FoVs. For example, the users in group G_1 are likely to be interested in the top center region of the VR frame, reflected by their FoVs. The BS is equipped with a computing server that can pre-render FoVs for different groups of users. We consider a downlink transmission setting in which the BS can transmit VR video frames to the users in a broadcast manner. We adopt HRSMA as an interference management scheme for our framework. With HRSMA, the VR video frames are transformed into a sequence of bits, i.e., messages, after the source encoding process. Let W_k denote the intended message for user k. With HRSMA, W_k is first split into three parts that are (i) a common message W_k^c, (ii) a group message W_k^g, and (iii) a private message W_k^p. The private message of user k is sent over a private stream z_k. Whereas the group messages and common messages are combined into G group streams, i.e., $\{z_g\}_{g\in\mathcal{G}}$, and one common stream z_c, respectively. The M-antenna BS then broadcasts the signal $\mathbf{x}$ defined as:

$$\mathbf{x} = \sqrt{P_0}\Big(\sqrt{\alpha_c}z_c + \sum_{g\in\mathcal{G}}\sqrt{\alpha_g}z_g + \sum_{k\in\mathcal{K}}\sqrt{\alpha_k}z_k\Big), \quad (1)$$

where P_0 is the transmit power of the BS, α_c, α_g and α_k are the power allocation coefficients. The received signal at user k is:

$$\mathbf{y}_k = \mathbf{h}_k^H\mathbf{x} + w_k, \quad (2)$$

where $\mathbf{h}_k \in \mathbb{C}^{M\times 1}$ is the channel between the user k and the BS, and $w_k \in \mathcal{CN}(0,\sigma_k^2)$ is the Additive White Gaussian Noise (AWGN). The decoding process at user k is as follows. The common stream z_c is first decoded by treating interference from other groups and private streams as noise. The SINR of decoding the common stream z_c at user k is:

$$\gamma_k^c = \frac{|\mathbf{h}_k|^2 P_0\alpha_c}{|\mathbf{h}_k|^2\sum_{g\in\mathcal{G}}P_0\alpha_g + |\mathbf{h}_k|^2\sum_{k\in\mathcal{K}}P_0\alpha_k + \sigma_k^2}. \quad (3)$$

Once the common stream is decoded, it is subtracted from the received signal $\mathbf{y}_k$ by using Successive Interference Cancellation (SIC). Next, the user decodes the group common stream z_g by treating interference from other private streams as noise. The SINR of decoding the group common stream z_g is:

$$\gamma_k^g = \frac{|\mathbf{h}_k|^2 P_0\alpha_g}{|\mathbf{h}_k|^2\sum_{g'\neq g}P_0\alpha_{g'} + |\mathbf{h}_k|^2\sum_{k\in\mathcal{K}}P_0\alpha_k + \sigma_k^2}. \quad (4)$$

Finally, the user k decodes its private stream z_k, whose corresponding SINR is:

$$\gamma_k^p = \frac{|\mathbf{h}_k|^2 P_0\alpha_k}{|\mathbf{h}_k|^2\sum_{g'\neq g}P_0\alpha_{g'} + |\mathbf{h}_k|^2\sum_{k'\neq k}P_0\alpha_{k'} + \sigma_k^2}. \quad (5)$$

Given the above SINR values, the achievable data rate of the user k in group g can be calculated by:

$$R_k^g = R_k^p + C_k^g + C_k^c, \quad (6)$$

where R_k^p is the achievable rate of decoding the private stream z_k, C_k^g is the achievable rate of decoding the group stream z_g, and C_k^c is the achievable rate of decoding the common stream z_c. Since the stream z_g and z_c are the combinations of multiple streams of the users, constraints (7) and (8) are applied to the

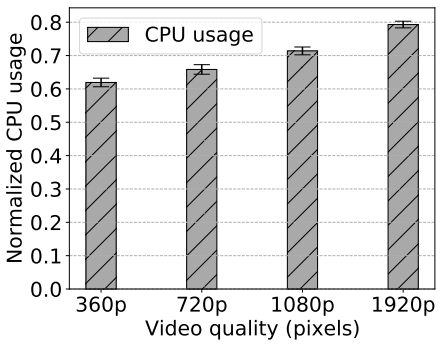

Fig. 2: An illustration of the CPU usage values of our local server under different FoV pre-rendering scenarios.

corresponding achievable rates C_k^g and C_k^c.

$$\sum_{k\in\mathcal{K}} C_k^c = \min_{k\in\mathcal{K}} \Big\{ \log_2(1+\gamma_k^c) \Big\}. \tag{7}$$

$$\sum_{k\in\mathcal{K}} C_k^g = \min_{k\in\mathcal{K}} \Big\{ \log_2(1+\gamma_k^g) \Big\}. \tag{8}$$

The constraints (7) and (8) guarantee the decodable of such streams at the receiver [5]. Unlike the group stream and common stream, each private stream is dedicated to a user individually so that R_k^p is given as follows:

$$R_k^p = \log_2(1+\gamma_k^p). \tag{9}$$

In 360^0 video streaming services, latency plays a critical role in users' QoE. In VR streaming applications, the system latency mainly consists of two factors, i.e., (i) the computing latency due to the FoV pre-rendering processed at the BS and (ii) the communication latency due to the transmitting video. In practice, since the FoV pre-rendering task is a intensive computing task at the BS, latencies for other computing tasks (e.g., processing for RMSA) can be considered to be negligible [11]. In this case, let N_x (bits) denote the data size, the latency of transmitting video of user k is defined by:

$$L_{v,k} = \frac{N_x}{BR_k^g}, \tag{10}$$

where B (Hz) is the broadcast channel's bandwidth.

To determine the latency due to the FoV pre-rendering at the BS, we perform a real experiment. First, we select the dataset containing 360^0 videos and corresponding users' behaviour data from [12]. Second, we use Vue-VR, an open-source software, to create a server that renders the FoV for each group $g \in \mathcal{G}$ [13]. Finally, CPU utilization of the rendering process is measured. Fig. 2 displays our experiment results for different video qualities, i.e., 360p, 720p, 1080p, and 1920p. This figure shows that the server uses more CPU capacity as the video quality increases. Based on these results, we can determine the latency due to FoV pre-rendering by

$$L_{r,k} = \frac{\mathbf{1}_k(\pi_G)}{f_g F_g \eta}, \tag{11}$$

where, $f_g \leq 1$ is the percentage of CPU cycles allocated to render FoV for group g, $F_d \leq 100\%$ is the current CPU usage for this rendering task derived from our above real experiment, and $\mathbf{1}_k(\pi_G)$ is the indicator function corresponding to a clustering policy π_G. Note that $\mathbf{1}_k(\pi_G) = 1$ if user k belongs to group g, and $\mathbf{1}_k(\pi_G) = 0$ otherwise. We now can derive the total latency of user k as $L_k = L_{r,k} + L_{v,k}$.

To maintain the fairness in terms of QoE among all the users, the BS will try to minimize the latency of the user with highest latency, called max-latency. Given the set of possible clustering policy Ψ_G, we can formulate the delay minimization problem as follows:

$$(\Phi): \min_{\mathbf{C},\ \boldsymbol{\alpha},\ \mathbf{f}} \ \max_{k\in\mathcal{K}} L_k \tag{12}$$

$$\text{s.t.} \quad \text{constrains (7) and (8)},$$

$$\alpha_c + \sum_{g\in\mathcal{G}} \alpha_g + \sum_{k\in\mathcal{K}} \alpha_k \leq 1, \tag{13}$$

$$\pi_G \in \Psi_G, \tag{14}$$

$$f_g \leq 1, \tag{15}$$

where vector $\mathbf{C} = [\{C_k^c\}_{k\in\mathcal{K}}, \{C_k^g\}_{g\in\mathcal{G},\ k\in\mathcal{K}}]$ represents the rate allocation, vector $\boldsymbol{\alpha} = [\alpha_c, \{\alpha_g\}_{g\in\mathcal{G}}, \{\alpha_k\}_{k\in\mathcal{K}}]$ represents the power allocation, and vector $\mathbf{f} = [f_1, \ldots, f_G]$ represents the computing resource allocation. Recall that the constraints (7) and (8) guarantee the decodable of received streams at the receiver. Constraint (13) guarantees that the total allocated power for communication is not higher than the power budget of the BS. Constraint (14) ensures the feasibility of clustering policy. Finally, constraint (15) guarantees that the allocated CPU cycles do not exceed the CPU capacity.

Note that if we leave out the rendering latency $L_{r,k}$ in L_k, the above optimization problem (Φ) is a kind of the multi-group multi-cast power control problem, which is an NP-hard problem, similar as NP-hard problem in [14]. This implies that even ignoring the FoV pre-rendering delay that is high uncertainty due to the dynamic of users' FoV as well as clustering policy, finding the optimal solution for (Φ) is still intractable. Moreover, the existing approaches based on conventional optimization for RMSA (e.g., [8], [9], and [16]) cannot be directly applied for (Φ) since they require complete information (e.g., the channel state distribution, users' demands, and CPU usage at the BS) in advance. Thus, it demands a solution that can address these dynamics and uncertainty of the environment to solve the considered problem. In the following, we introduce our proposed solution based on Deep Reinforcement Learning (DRL) to address these challenges.

III. DRL-enabled 360^0 Video Streaming with RMSA

This section discusses how to address the optimization problem (Φ) with DRL. In the DRL, based on the observed environment state s_t at time t, an agent takes an action a_t according to its current policy π. Then, at the end of time slot t, the agent observes a new state s_{t+1} and receives an immediate reward r_t indicating how well it performs. The optimal policy of the agent can be determined as the policy that maximizes the long-term average reward function as follows:

$$\pi^* = \underset{\pi}{\text{argmax}}\ \frac{1}{T}\sum_{t=0}^{T} r_t, \tag{16}$$

where T is the time horizon. Thus, finding the optimal policy π^* is actually a policy search problem, which can be solved by DRL algorithms. Note that a policy can be defined as a probability distribution over available actions at a state, i.e., $\pi = \Pr\{a_t \mid s_t\}$. Given the above, to address the optimization problem in (Φ), we first transform it to a reinforcement learning process. We then propose an effective learning algorithm based on Proximal Policy Optimization (PPO) to quickly find the optimal policy for the system.

A. DRL-based Optimization Formulation

We consider that a DRL agent is implemented at the BS to make decisions for the system. At time t, the agent observes its surrounding environment and determines the system's parameters (i.e., $\mathbf{C}$, $\boldsymbol{\alpha}$, $\mathbf{f}$) according to its observation, i.e., state. Thus, a system state consists of (i) the root mean square of all channel gains between the BS and users, i.e., $\{\hat{h}_k\}_{k\in\mathcal{K}}$, (ii) the maximum portion of CPU cycles allocated to G groups for FoV rendering, i.e., $\{F_g\}_{g\in\mathcal{G}}$, and (iii) the current clustering policy π_G. In this way, the system state space can be defined as follows:

$$\mathcal{S} = \Big\{ < \{\hat{h}_k\}_{k\in\mathcal{K}}, \{F_g\}_{g\in\mathcal{G}}, \pi_G >; \\ \hat{h}_k \in \mathbb{R}, F_g \in [0,1], \text{ and } \pi_G \in \Psi_G \Big\}, \tag{17}$$

Note that a clustering policy π_G is represented by a $G \times K$ binary matrix in which a row and a column correspond to a group and a user, respectively.

Recall that to improve the VR users' QoE, the agent aims to minimize the max-latency by optimizing the system parameters, i.e., power allocation coefficient vector $\boldsymbol{\alpha}$, rate allocation vector $\mathbf{C}$, and computing resource allocation vector $\mathbf{f}$, as in the objective function of problem Φ in (12). Thus, the action of the agent is defined as $\mathcal{A} = \{\boldsymbol{\alpha}, \mathbf{C}, \mathbf{f}\}$. Note that the BS aims to minimize the max-latency while in the DRL setting the agent's optimal policy is to maximize the long-term average reward, as shown in (16). Therefore, the reward of performing an action a_t at state s_t can be defined as the inverse of the max-latency as follows:

$$r_t(s_t, a_t) = \frac{1}{\max_{k\in\mathcal{K}} L_k}. \tag{18}$$

Given the above definitions of state space, action space, and reward function, we can re-formulate the problem (Φ) into DRL setting (i.e., finding the optimal policy π) as follows:

$$(\Phi^-): \max_{\pi} \quad \mathbb{E}_{s_t,a_t}\left[\frac{1}{T}\sum_{t=0}^{T} r_t\right], \tag{19}$$

$$\text{s.t.} \quad s_t \in \mathcal{S}, \tag{20}$$

$$a_t \in \mathcal{A}, \tag{21}$$

$$a_t \sim \pi(s_t), \tag{22}$$

$$s_t \sim \mathcal{P}(s_{t-1} \mid s_{t-1}, a_{t-1}), \tag{23}$$

where $\mathcal{P}(s_{t-1} \mid s_{t-1}, a_{t-1})$ is the state transition probability (i.e., the probability that the system moves to a next state s_{t+1} given the current state s_t and the selected action a_t) indicating the environment's dynamic, i.e., wireless channel quality and FoV pre-rendering demands. Note that the BS does not know the state transition probability in advance. In practice, it is very challenging to obtain this information due to the high dynamic and uncertainty of BS's surrounding environment. In the next subsection, we thus discuss our proposed learning approach to find the optimal policy for the BS without requiring complete information about the surrounding environment.

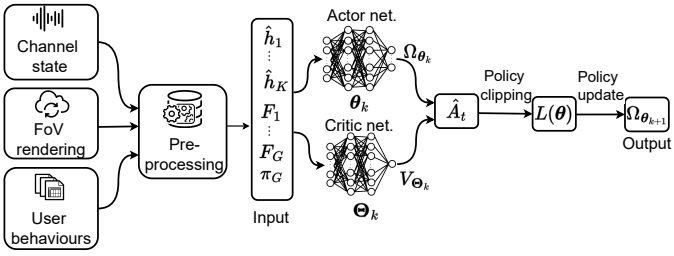

Fig. 3: The training process of our proposed DRL model.

B. DRL-powered RSMA for 360^0 Video Streaming

Algorithm 1 Proximal Policy Optimization

1: **Input:**
2: Initialize the actor network π_{θ_0} with parameters θ_0.
3: Initialize the critic network $\mathcal{V}_{\Theta_0}$ with parameters Θ_0.
4: **for** $k = 0, 1, 2, \ldots$ **do**
5: Using current policy derived from π_{θ_k} to collect set of trajectories $\mathcal{B}_k = \{(s_t, a_t, r_t) \mid t \in \{0, 1, \ldots, T\}\}$.
6: Determine advantage function $\hat{A}_t$ as in (24).
7: Update the actor and critic networks by Eqs. (27) and (28), respectively.
8: **end for**
9: **Output:** $\pi^* = \Pr\{a_t \mid s_t; \theta\}$

We illustrate our proposed training process in Fig. 3. In particular, our proposed solution is established based on two main elements, i.e., the data processing and learning algorithm. First, since the state space of the BS consists of data from different sources (i.e., channel gains from users, CPU usage for FoV rendering at the BS, and the clustering policy), it is necessary to perform the data processing phase to remove abnormal data points, i.e., data with invalid values. For instance, the CPU usage is larger than 100%. Second, to help the BS quickly find the optimal policy, this paper adopts the Proximal Policy Optimization (PPO) [15] instead of conventional action value-based DRL approaches due to several reasons. First, conventional action value-based methods (e.g., Q-learning, Deep Q-learning, and Deep Dueling Q-learning) can only address problems with discrete action space. However, in the considered problem, the action space is continuous, and thus these approaches could not be applied directly. In contrast, the PPO is a policy-based approach that can effectively address the continuous and high dimensional action space [15]. Second, thanks to the clipping technique applied on gradient step update, the PPO's learning process is stable in the high dimensional state and action spaces.

To learn the optimal policy for the BS, PPO leverages two deep neural networks, i.e., the actor and critic networks.

In particular, the BS's policy is parameterized by the actor network π_θ with parameters θ. In other words, when a state is feed to the actor network π_θ, it outputs an action for the system. Whereas the critic network $\mathcal{V}_\Theta$ with parameters Θ estimates the state-value function $V_t(s_t)$ that indicates how good is it for an agent to be at a state. The aim of PPO is to train the actor and critic networks by using data collected via interactions between the agent and its surrounding environment, e.g., observed states, actions, and rewards. The details of PPO's learning process are described in Algorithm 1 with main steps are as follows. For each learning iteration k, the BS first operates for T time steps under its current policy derived from the π_{θ_k} to collect a set of trajectories $\mathcal{B}_k = \{b_t = (s_t, a_t, r_t)\}_{t=0}^{T}$. Based on these trajectories, the BS then calculates the advantage function by using the generalized advantage estimator (GAE) [15] as follows:

$$\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \ldots + (\gamma\lambda)^{T-t+1}\delta_{T-1}, \quad (24)$$
$$\text{with } \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t),$$

where $\gamma \in [0, 1]$ is the discount factor that indicates the importance of future rewards, and $\lambda \in [0, 1]$ is the parameter controlling the bias and variance of GEA estimator [15]. Note that the advantage function indicates how good it is to perform an action compared with other actions at a given state. Given the advantage function $\hat{A}_t$, the training objective of the BS is calculated by:

$$J(\theta) = \min\left(\frac{\pi_{\theta_k}(a_t \mid s_t)}{\pi_{\theta_{k-1}}(a_t \mid s_t)}\hat{A}_t,\ clip\left(\epsilon\right)\hat{A}_t\right), \quad (25)$$

$$\text{where } clip(\epsilon) = \begin{cases} (1+\epsilon), & \text{if } \hat{A}_t \geq 0\ , \\ (1-\epsilon), & \text{if } \hat{A}_t < 0\ . \end{cases} \quad (26)$$

Recall that the policy is defined as a probability distribution over actions. As such, the probability ratio between new and old policies, i.e., $\frac{\pi_{\theta_k}(a_t|s_t)}{\pi_{\theta_{k-1}}(a_t|s_t)}$, is clipped between $[1-\epsilon, 1+\epsilon]$. In other words, the clip function, i.e., $clip(\epsilon)$, keeps the new policy from being updated too far from the old policy at each learning iteration, thereby stabilizing the learning process [15]. Finally, the parameters of the actor and critic are updated by Eqs. (27) and (28), respectively.

$$\theta_{k+1} = \underset{\theta}{\text{argmax}} \frac{1}{|\mathcal{B}_k|T} \sum_{b_t \in \mathcal{B}_k} \sum_{t=0}^{T} J(\theta), \quad (27)$$

$$\Theta_{k+1} = \underset{\Theta}{\text{argmin}} \frac{1}{|\mathcal{B}_k|T} \sum_{b_t \in \mathcal{B}_k} \sum_{t=0}^{T} \left(V_t(s_t, \Theta) - \sum_{t=0}^{T} r_t\right)^2. \quad (28)$$

IV. Performance Evaluation

This section presents the evaluation of our proposed approach. The simulation settings are set as follows. For the wireless environment, we use the similar channel model in [16], e.g., the number of antennas is six, and the constant channel gain is given as $\bar{\boldsymbol{h}}_k = g_k \times [1, e^{j\phi_k}, e^{j2\phi_k}, \ldots, e^{j5\phi_k}]$, where $\phi_k \in [0, 2\pi]$ and $g_k \in \mathbb{R}$ are control variables. The BS's power budget is 20 dB, and the noise at user k is drawn from complex Gaussian distribution, i.e., $\mathcal{CN}(0, g_k P^{-\beta_k})$, where

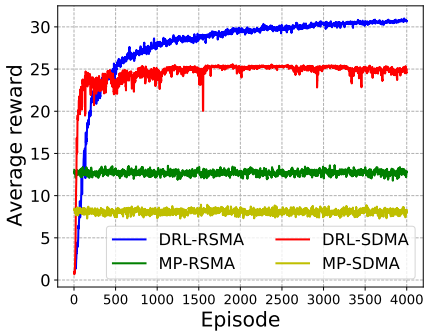

Fig. 4: Average reward values of our proposed approach, i.e., DRL-RSMA, compared to other approaches, in the training process.

$\beta_k = 0.6$ is the degree of freedom variable. The number of users and the number of groups are 6 and 3, respectively. For 360^0 video streaming, the selected dataset consisting of 88 videos with corresponding users' behaviours described in [12]. This dataset serves two purposes, i.e., FoV rendering and classifying users into multiple groups according to their interests. Since our objective is to provide a DRL-enabled 360^0 video streaming with RSMA, we select the SDMA as a baseline for the multiple access aspect. In particular, SDMA separately encodes messages into different streams for different users. Then, each user treats messages from others as noise. In addition, to evaluate our proposed learning algorithm, we choose the Myopic Policy (MP) as a baseline algorithm. The MP selects an action to maximize the immediate reward without considering the long-term average reward.

We first evaluate the convergence rate of our proposed learning approach under RSMA and SDMA schemes, as shown in Fig. 4. It can be observed that under the RSMA scheme, our proposed DRL-RMSA quickly outperforms MP-RSMA only after 50 episodes. In addition, after $3,500$ episodes. DRL-RMSA's average reward is stable at about 30, which is 136% greater than that of the MP-RSMA, respectively. Similarly, under the SDMA scheme, our proposed DRL learning algorithm clearly outperforms MP-SDMA. Moreover, Fig. 4 also demonstrates the benefit of the RSMA over the SDMA. Specifically, DRL and MP under the RSMA achieve 20% and 35% greater average rewards compared with those of the DRL and MP under the SDMA, respectively.

Next, we study the performance of our proposed DRL-RSMA under different system settings. First, to study the impacts of communication to the system, the BS's transmit power budget is varied from 0 to 20 dB, as shown in Fig. 5. Fig. 5(a) shows that the system latency obtained by our proposed DRL-RSMA decreases as the power budget increases from 0 to 20 dB. In addition, DRL-RSMA always achieves the lowest system latency compared to those of other approaches. In particular, when the transmit power is equal or larger than 10 dB, the DRL-RSMA's system latency achieves an order of magnitude 10^{-3}, i.e., equal or less than 61 milliseconds that can guarantee the seamless user experiences [11]. In contrast,

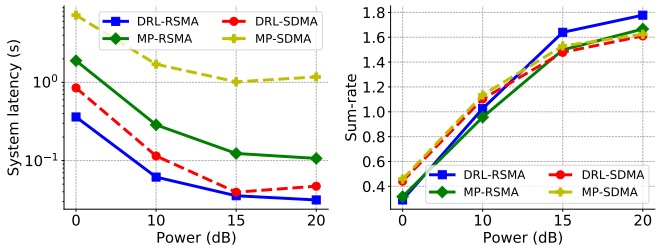

Fig. 5: (a) System latency and (b) sum-rate values when the power budget at the BS is varied.

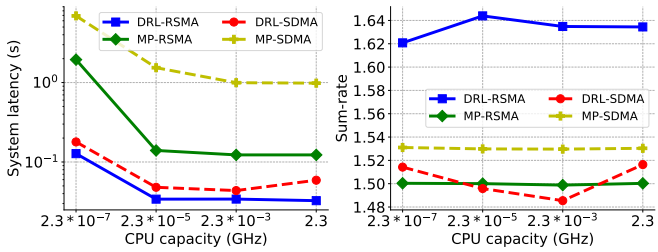

Fig. 6: (a) System latency and (b) sum-rate values when the CPU capacity of the BS is varied.

the system latencies of MP-SDMA and MP-RSMA are always larger than 100 ms, possibly causing motion sickness for users [11]. In terms of the sum-rate metric, as shown in Fig. 5(b), when the power budget is equal to or larger than 15 dB, our proposed DRL-RSMA achieves the highest sum-rate values in comparison with those of other approaches. It is stemmed from the fact that the RSMA works more effectively in the high-power region. In addition, our proposed approach aims to guarantee user fairness instead of the sum-rate.

Second, the CPU capacity of the BS is varied from 2.3×10^{-7}GHz to 2.3 GHz to investigate the impacts of computational capacity on the system performance, as shown in Fig. 6. Fig. 6(a) shows that as the CPU capacity increases, the latencies of all approaches decrease since the computation time for FoV pre-rendering decreases. It is observed that our proposed DRL-RSMA always satisfies the millisecond latency requirement of VR streaming, except when CPU capacity is 2.3×10^{-7} that is relatively low computing capacity for pre-rendering users' FoVs. In contrast, the latencies of MP-based approaches are always higher than 100 ms, which does not meet the latency requirement of VR streaming. In terms of sum-rate, the results obtained by all approaches are nearly unchanged as the CPU capacity increases, as shown in Fig. 6(b). The reason is that the sum-rate mainly depends on the power budget of the BS.

V. Conclusion

This paper has proposed a novel framework that can effectively manage interference in the wireless environment and jointly optimize communication and computing resources to facilitate VR streaming under RSMA. In particular, we have proposed a highly-effective user clustering method based on users' interests, i.e., FoVs, for RSMA-based VR streaming. By doing so, our framework can leverage the advantages of RSMA in interference management as well as FoV pre-rendering in reducing the amount of transmitting VR data. Then, we have developed an intelligence algorithm based on DRL to address the uncertainty and heterogeneity of the VR streaming environment. Simulation results showed that our proposed solution outperforms baseline methods and can satisfy the VR streaming requirement of millisecond latency.

References

[1] N. Gilbert, "74 virtual reality statistics you must know in 2021/2022: Adoption, usage & market share," https://financesonline.com/virtual-reality-statistics/, (accessed: Jun. 18, 2022).

[2] X. Jiang, F. R. Yu, T. Song, and V. C. Leung, "A survey on multi-access edge computing applied to video streaming: some research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 871–903, 2021.

[3] X. Wei, C. Yang, and S. Han, "Prediction, communication, and computing duration optimization for vr video streaming," *IEEE Transactions on Communications*, vol. 69, no. 3, pp. 1947–1959, Mar. 2020.

[4] C. Perfecto, M. S. Elbamby, J. Del Ser, and M. Bennis, "Taming the latency in multi-user vr 360°: A qoe-aware deep learning-aided multicast framework," *IEEE Transactions on Communications*, vol. 68, no. 4, pp. 2491–2508, Apr. 2020.

[5] Y. Mao, O. Dizdar, B. Clerckx, R. Schober, P. Popovski, and H. V. Poor, "Rate-splitting multiple access: Fundamentals, survey, and future research trends," *IEEE Communications Surveys & Tutorials*, Jul. 2022.

[6] Y. Mao, B. Clerckx, and V. O. Li, "Rate-splitting for multi-antenna non-orthogonal unicast and multicast transmission: Spectral and energy efficiency analysis," *IEEE Transactions on Communications*, vol. 67, no. 12, pp. 8754–8770, Dec. 2019.

[7] M. Dai, B. Clerckx, D. Gesbert, and G. Caire, "A rate splitting strategy for massive mimo with imperfect csit," *IEEE Transactions on Wireless Communications*, vol. 15, no. 7, pp. 4611–4624, Mar. 2016.

[8] L. Yin and B. Clerckx, "Rate-splitting multiple access for multigroup multicast and multibeam satellite systems," *IEEE Transactions on Communications*, vol. 69, no. 2, pp. 976–990, Feb. 2020.

[9] O. Tervo, L.-N. Trant, S. Chatzinotas, B. Ottersten, and M. Juntti, "Multigroup multicast beamforming and antenna selection with rate-splitting in multicell systems," in *Proc. IEEE Int. Workshop Signal Process. Adv. Wireless Commun.*, 2018, pp. 1–5.

[10] "How to avoid VR interference in the modern ARCADE," https://laigames.com/avoid-vr-interference/, (accessed: Jun. 18, 2022).

[11] S. Mangiante, G. Klas, A. Navon, Z. GuanHua, J. Ran, and M. D. Silva, "VR is on the edge: How to deliver 360 videos in mobile networks," in *Proceedings of the Workshop on Virtual Reality and Augmented Reality Network*, 2017, pp. 30–35.

[12] A. Dharmasiri, C. Kattadige, V. Zhang, and K. Thilakarathna, "Viewport-aware dynamic 360° video segment categorization," in *Proceedings of the 31st ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2021, pp. 114–121.

[13] M. Ibrahim, "Vue vr: A wrapper of panolens for building vr applications with vue based on threejs," https://github.com/mudin/vue-vr, (accessed: Jun. 18, 2022).

[14] E. Karipidis, N. D. Sidiropoulos, and Z.-Q. Luo, "Quality of service and max-min fair transmit beamforming to multiple cochannel multicast groups," *IEEE Transactions on Signal Processing*, vol. 56, no. 3, pp. 1268–1279, Feb. 2008.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[16] Y. Mao, B. Clerckx, and V. O. Li, "Rate-splitting multiple access for downlink communication systems: bridging, generalizing, and outperforming sdma and noma," *EURASIP journal on wireless communications and networking*, vol. 2018, no. 1, pp. 1–54, May 2018.