Intelligent QoE Management for IoMT Streaming Services in Multi-User Downlink RSMA Networks

The-Vinh Nguyen, Duc-Thien Hua, Thien Ho Huong, Vinh Truong Hoang, Nhu-Ngoc Dao, Sungrae Cho

***Abstract*—The exponential growth of the Internet of Multimedia Things (IoMT) traffic has posed a threat of service quality degradation due to the limitation of current communication, networking, and computing advances in mobile networks. In this regard, managing the Quality-of-Experience (QoE) for IoMT services is a vital challenge to meet user satisfaction. To cope with this problem, we investigate the joint optimization of video quality variation and latency in multi-user downlink Rate-Splitting Multiple-Access (RSMA) networks, especially within imperfect network conditions and state information. To accomplish this, we first formulated the joint optimization problem into a Markov decision process framework, then exploited a deep reinforcement learning approach to adaptively calculate the optimal configuration of the RSMA against environment dynamics. As a result, the proposed *Deep Deterministic Policy Gradient on RSMA-based Video streaming System (DDPG-RMAVS)* provides QoE maintenance by minimizing video resolution reduction and latency. Extensive simulation results revealed that the proposed *DDPG-RMAVS* algorithm surpasses existing algorithms by achieving higher video quality, lower delay, larger buffer capacity, and limited stalling events, representing a significant breakthrough in IoMT streaming optimization.**

***Index Terms*—Internet of multimedia things, quality of experience, rate splitting multiple access, mobile network**

I. Introduction

THE popularity of mobile Internet of Multimedia Things (IoMT) streaming services has grown significantly due to their convenience and accessibility. According to Ericsson Mobility report data and forecasts [1], the total global mobile traffic has reached 118 EB per month at the end of the year 2022, and it is forecast to incline to 472 EB per month by the end of 2028, with 66% of the share belongs to 5G's mobile data traffic. In particular, IoMT traffic currently accounts for 71% of all mobile data traffic, and it is projected to increase to 80% by the year 2028, making it the most dominant data category on the Internet. As a result, the demand for IoMT streaming services is expected to increase as more users enter the market. The emergence of video service providers such as Youtube, Tiktok, and Twitch demonstrates the ubiquity of video services in our daily lives.

As the Internet and IoMT services boom, there is a need for an efficient multiple-access framework that can efficiently utilize wireless resources and provide massive connectivity. One such potential technology is the Rate-Splitting Multiple-Access (RSMA) [2], which allows multiple users to transmit data simultaneously over a shared spectrum. The working principle of RSMA involves separating the transmission data of each user into two parts: a common part that all users can receive and utilize, and a private part that is designated specifically for each user. Each user then reconstructs the original message from the common and private messages using the Successive Interference Cancellation (SIC) technique [3]. This approach has been proven to be able to improve significantly spectral efficiency and offer an alternative to traditional methods like Space-Division Multiple-Access (SDMA), Frequency-Division Multiple-Access (FDMA), and the recent Non-Orthogonal Multiple-Access (NOMA) of 5G networks [4].

A. Motivation

With the advantages of RSMA, the application of IoMT services would bring many benefits to users' experience. One of the most notable achievements of RSMA is the advanced resource allocation capability [5], [6]. RSMA enables better resource allocation among multiple users, which can lead to improved Quality-of-Service (QoS) and Quality-of-Experience (QoE); hence users will experience fewer interruptions or buffering when experiencing a video stream. Secondly, RSMA has a powerful interference management capability, which can increase the overall capacity of the streaming system while stabilizing the network between each user and the server [7], [8]. This means more users can access the IoMT service simultaneously without a downgrade in video transmission due to highly dense video data traffic. Moreover, RSMA ensures that each user is allocated a fair share of the wireless communication resources [9], [10]. No single user will monopolize the available resources, and all users will have an equal opportunity to access the IoMT service. Last but not least, RSMA can result in better energy efficiency by optimising wireless communication resources [11], [12]. Devices can operate more efficiently and use less battery power. Overall, RSMA is a valuable technique for improving the performance and efficiency of wireless communication systems and can provide significant benefits for IoMT services.

A simple sketch of end-to-end video transmission is depicted in Fig.1. With the demand dramatically increasing,

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2023-RS-2022-00156353) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation). *(Corresponding authors: Thien Ho Huong, Nhu-Ngoc Dao, Sungrae Cho)*

T.-V. Nguyen and N.-N. Dao are with the Department of Computer Science and Engineering, Sejong University, Seoul 05006, South Korea (e-mails: nguyenthevinh@sju.ac.kr, nndao@sejong.ac.kr).

D.-T. Hua and S. Cho are with the School of Computer Science, Chung-Ang University, Seoul 06974, South Korea (e-mails: thien@uclab.re.kr, srcho@cau.ac.kr).

T.H. Huong and V.T. Hoang are with the Faculty of Computer Science, Ho Chi Minh City Open University, Ho Chi Minh City 70000, Vietnam (e-mails: thien.hh@ou.edu.vn, vinh.th@ou.edu.vn).

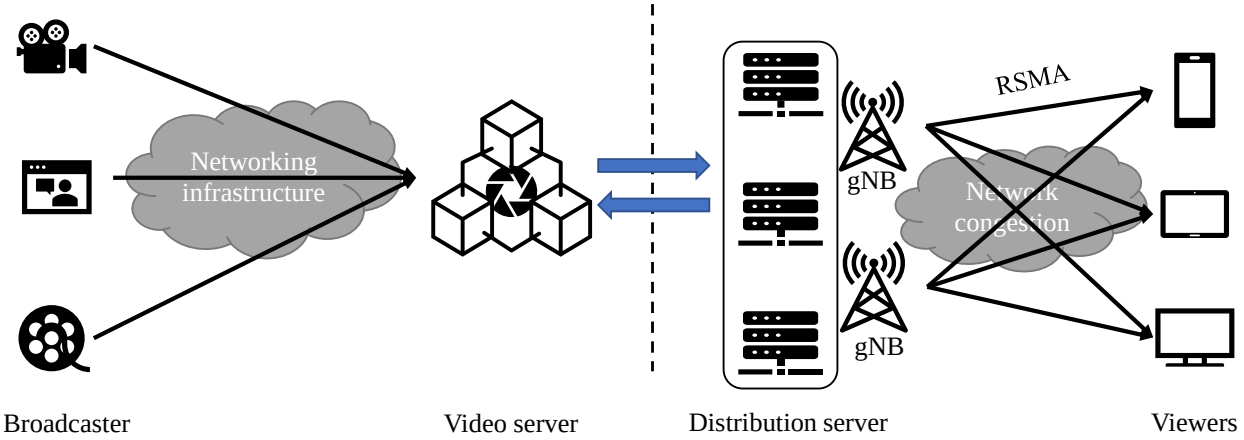


Fig. 1: End-to-end IoMT transmission.

network congestion appears during the downlink distribution. Despite the numerous advantages and benefits that RSMA could offer for IoMT services, the extent of discussion regarding their integration remains highly limited. While recent research acknowledges that IoMT streaming is among the services that can benefit from RSMA, these studies failed to provide substantial details. To the best of our knowledge, there has not been any official scientific research conducted on studying IoMT services over an RSMA network, highlighting the need for further investigation into this area. In this paper, we have modeled a downlink IoMT streaming scenario, in which RSMA was deployed at the Base Station (BS) and simultaneously sent video data from the server to multiple users. Additionally, we focus on resolving the two important QoE utilities of IoMT streaming: *Latency* and *Quality*. In order to optimize the two aforementioned utilities under the dynamic environment of an RSMA-based IoMT streaming system, we came up with a Deep Reinforcement Learning (DRL) algorithm, namely *Deep Deterministic Policy Gradient on RSMA-based Video streaming System (DDPG-RMAVS)*.

B. Contributions

In this work, we investigate how *DDPG-RMAVS* can resolve the joint optimization problem between maximizing video quality and minimizing the transmission latency on an IoMT streaming system over RSMA networks. The major contributions of this work are as follows:

- We considered a downlink IoMT streaming system in multi-user RSMA networks. We aimed to jointly optimize the two important QoE utilities of IoMT streaming service: *Video quality* and *Latency* by managing the video bitrate selection and signal transmit power at BS. We formulated the optimization problem under transmit power constraint, buffer constraint, and minimum rate constraint of users. We especially investigated the imperfect video transmission environment, in which an imperfect Channel State Information at Transmitter (CSIT) and an imperfect SIC were considered.
- The optimization problem is non-convex and challenging to solve with traditional optimization methods. Therefore, we proposed the *DDPG-RMAVS* algorithm, a DRL-based approach as the feasible way to achieve a sub-optimal solution. We have transformed the optimization problem into a Markov Decision Process (MDP) framework to facilitate the use of *DDPG-RMAVS*.
- We demonstrated through simulation results that the proposed algorithm achieves better performance than other state-of-the-art optimization models in terms of quality, latency, buffer capacity, and the number of stalling events. We conclude that the application of *DDPG-RMAVS* onto RSMA-based IoMT streaming system can enhance the QoE of users, particularly in terms of latency and video quality.

The structure of this paper is as follows. The system model and the optimization problem's formulation are explained in Section III. We present the MDP framework and suggest the application of *DDPG-RMAVS* in Section IV. Section V provides a detailed account of the simulation setting and presents numerical results alongside comparative analysis. We also discuss some of the possible drawbacks remaining on the paper, as well as providing some promising future research directions in VI. Finally, Section VII offers a summarized conclusion.

Notation: Upper and lower case letters written in bold represent the matrices and vectors, respectively. Transpose, Hermitian, Euclidean norm, and expectation operator are represented by $(.)^T$, $(.)^H$, $\|.\|$, and $\mathbb{E}\{.\}$, respectively. The circularly symmetric complex Gaussian (CSCG) distribution with zero mean and variance σ^2 is denoted as $\mathcal{CN}(0, \sigma^2)$.

II. Related Works

Given the superior of RSMA scheme and the potential of DRL models for IoMT streaming and RSMA, we have organized related works into three main categories: (*i*) comparison between RSMA and other traditional multiple access schemes, (*ii*) the utilization of DRL solutions in RSMA, and (*iii*) the application of DRL techniques to address utility problems in live IoMT streaming.

A. RSMA in Comparison with Traditional Schemes

The most notable achievement that distinguishes RSMA from other multiple access methods is sophisticated interference management. As previously stated, RSMA divides messages into common and private messages, allowing RSMA to partially interpret interference as noise and partially decode interference. This strategy, incorporating the interference management methods of NOMA and SDMA, enables RSMA to handle interference more efficiently, therefore achieving higher performance than NOMA/SDMA.

The work of [2] has comprehensively conducted a comparison between RSMA and SDMA/NOMA in different metrics. Firstly, RSMA gains more Degree-of-Freedom (DoF) than NOMA and SDMA in imperfect CSIT environment [13], [14], [15]. Similarly, Clerckx *et al.* [16] demonstrated that larger sum-DoF and symmetric DoF in RSMA enable the exploitation of multi-antenna strategy in Multiple-input multiple-output (MIMO). Secondly, the precoders in RSMA outperform other schemes in terms of achievable rate while reducing complexity and computational weight. Precoder design adds to the success of inter-user interference control by allowing users to use greater rates. Schroder *et al.* [17] compared the achievable rate maximization in multibeam satellite systems and highlighted the superiority of the RSMA system. According to the research of [18], [19], [20], RSMA outperforms SDMA and NOMA in both perfect and imperfect CSIT by attaining higher average rate region and weighted sum rate under various user deployments and network loads. Finally, as compared to NOMA/SDMA, RSMA achieves the greatest throughput in both Shannon Bound and Link-level simulations platforms[21]. Furthermore, the work of [22] indicates that RSMA is more robust to high mobility users in MIMO networks because it achieves higher throughput than SDMA, whereas NOMA is vulnerable to high user mobility due to the user clustering problem.

B. Deep Reinforcement Learning with RSMA

RSMA wireless communication networks are intricate, marked by dynamic channel conditions, user mobility, and interference variations. Tackling the optimization challenges in RSMA utilities, given the network's non-convex and unstable nature, is demanding. DRL has emerged as a promising solution for such complex environments, allowing the development of optimal policies through interaction with the network without prior knowledge of the system model.

Recent research has focused on power allocation in RSMA, aiming to maximize sum rates, minimize transmit power, and enhance user fairness [2]. RSMA splits data rates into multiple streams, demanding significant power and potentially impacting energy consumption and system performance. Various algorithms have been proposed to address RSMA challenges. In [23], a Proximal Policy Optimization (PPO) rate allocation. Giang *et al.* [24] employed Deep Q-Learning (DQN) for uplink RSMA's sum-rate maximization. Zhang *et al.* [25] integrated RSMA with intelligent surface-assisted wireless information and power transfer. PPO jointly determined optimal power allocation, common rate, beam-forming, and phase shifting.

Unmanned Aerial Vehicle (UAV)-based RSMA has gained attention, as it enhances spectral efficiency in bandwidth-limited UAV networks. In [26], Thien *et al.* used a Deep Deterministic Policy Gradient (DDPG) model to maximize sum rates in a downlink RSMA system deployed on UAVs. DDPG was also applied to optimize UAV trajectory for sum-rate maximization in [27]. Truong *et al.* proposed *HAMCE* a system optimizing RSMA aspects, including offloading decision, splitting ratio, transmit power, and decoding order, using DDPG [28]. The study investigated the impact of Ornstein-Uhlenbeck (OU) noise on DRL model exploration and exploitation. Ji *et al.* [29] explored uplink-downlink decoupled user association in RSMA within a multi-UAV scenario, considering various constraints to maximize user and multicast group sum rates.

C. Deep Reinforcement Learning with IoMT Streaming

DRL methods have gained prominence in optimizing IoMT streaming, particularly in adapting bitrate algorithms to varying network conditions and user preferences. Notable applications of DRL in video utilities encompass video resolution, latency, energy efficiency, and security [30]. For instance, Huang *et al.* used DRL in [31] to dynamically select video bitrates based on past frame quality, enhancing real-time IoMT streaming quality and stability. Hong *et al.* [32] employed DDPG for bitrate and latency control, outperforming DQN by 3.6% in QoE. In the context of 360-degree video streaming, Zhang *et al.* presented *DRL360* in [33], a DRL framework leveraging Deep Learning for bandwidth and view-ports prediction, and an Actor-Critic model for tile rate allocation. Pang *et al.* [34] proposed an actor-critic-based Asynchronous Advantage Actor Critic (A3C) algorithm to optimize latency in 360-degree videos, emphasizing their latency sensitivity.

An interesting study related to both IoMT streaming and RSMA, employing DRL, is the work of Hieu *et al.* [35]. They addressed transmission latency in RSMA-based virtual reality IoMT streaming by clustering users according to their priority for virtual video streams and utilizing a PPO-based model to mitigate latency under computational constraints. However, this research focused on a specific type of video content (virtual video streams), making it distinct from the broader scope of live IoMT streaming. In light of this research gap, our study provides a comprehensive examination of RSMA-based IoMT streaming, encompassing a practical and generalized IoMT streaming framework.

III. Problem Statement

This paper considers a live IoMT streaming system in multi-user downlink RSMA networks with imperfect CSIT and SIC, as shown in Fig.2. This model consists of one multi-antenna gNB and a set of $\mathcal{K} = \{1, \ldots, K\}$ single-antenna users. The number of transmit antennas equipped at the gNB is M. BS is assumed to employ the one-layer RSMA technology in this model to send video signals to every user simultaneously. Without loss of generality, the terms *video signals* and *messages* might be used interchangeably throughout this study. Adopting the 3GPP standard model for multimedia services in mobile networks [36], the 5G media streaming (5GMS) server

TABLE I: List of denotations

Denotation	Description
K	Total number of users
M	Total number of antenna on BS
w	Bandwidth of BS
$\mathbf{p}_c$	Transmit power for common stream
$\mathbf{p}_{p,k}$	Transmit power for private stream of user k
P	Total transmit power of BS
$\mathbf{x}$	Transmit signal from BS
$\mathbf{h}_k$	Perfect CSIT of user k
$\mathbf{e}_k$	CSIT error estimation of user k
$\widehat{\mathbf{h}}_k$	Imperfect CSIT of user k
n_k	AWGN noise of user k
σ^2	Noise variance
ξ	Imperfect SIC variance for common stream
$\delta_k[t]$	Playtime of video chunks at timeslot t of user k
Π	Size in bit of one video chunk
τ	Duration of one timeslot in seconds
$r_k[t]$	Bitrate of video chunks at timeslot t of user k
$b_k[t]$	Buffer in seconds at timeslot t of user k
$\rho_k[t]$	Video playtime at timeslot t

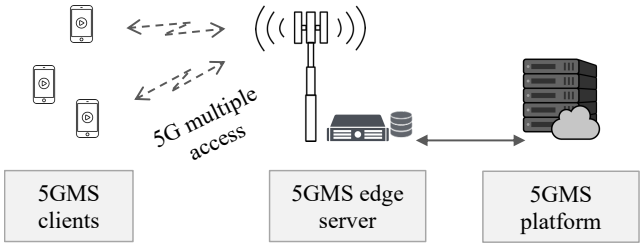


Fig. 2: RSMA downlink 5GMS live streaming system model.

located at the gNB calculates optimal solutions for content transcoding and delivery while the central application servers at the core networks are in charge of video distribution and long-term content storage.

A. Channel Model

Denote a message sent by gNB to user k, $\forall k \in \mathcal{K}$, as W_k. Following the one-layer RSMA technique, the message W_k is firstly divided into two parts: a common message $W_{c,k}$ and a private message $W_{p,k}$ [18]. The common messages of all users, i.e., $\{W_{c,1}, \ldots, W_{c,K}\}$, are concatenated into a single common message, indicated as W_c, and accordingly encoded into a common stream $s_c \in \mathbb{C}$. The private message of user k, $W_{p,k}$, on the other hand, is encoded independently into a private stream $s_{p,k} \in \mathbb{C}$. Hence, the total number of encoded streams $\mathbf{s} = [s_c, s_{p,1}, \ldots, s_{p,K}]^T \in \mathbb{C}^{K+1}$, consists of K private streams and one common stream. The transmit power allocated for common stream s_c and private stream $s_{p,k}$ of user k are denoted as $\mathbf{p}_c \in \mathbb{C}^M$ and $\mathbf{p}_{p,k} \in \mathbb{C}^M$, respectively. Therefore, the transmit power matrix is represented as $\mathbf{P} = [\mathbf{p}_c, \mathbf{p}_1, \ldots, \mathbf{p}_K]^T \in \mathbb{C}^{M\times(K+1)}$. The transmit signal $\mathbf{x} \in \mathbb{C}^M$ is represented as follows [37]

$$\mathbf{x} = \sqrt{\mathbf{p}_c}\mathbf{z}_c s_c + \sum_{k=1}^{K} \sqrt{\mathbf{p}_{p,k}}\mathbf{z}_{p,k} s_{p,k}, \tag{1}$$

where $\mathbf{z}_c \in \mathbb{C}^M$ and $\mathbf{z}_{p,k} \in \mathbb{C}^M$ are the precoding vectors of the common and private streams of user k, and the power constraint $\mathbf{p}_c + \sum_{k=1}^{K} \mathbf{p}_{p,k} \leq P$ holds, i.e, the sum of allocated power for all message streams must not exceed the total available power P at the BS. For practical implementation, imperfect CSIT is assumed with a possible error in channel estimation. Let $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_K] \in \mathbb{C}^{M\times K}$ be the perfect CSIT estimation of the downlink from the gNB. Each element of $\mathbf{h}_k$ represents the channel gain between user k and each transmits antenna. The imperfect CSIT is given by

$$\widehat{\mathbf{h}}_k = \mathbf{h}_k + \mathbf{e}_k, \tag{2}$$

where $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_K] \in \mathbb{C}^{M\times K}$ be the channel estimation error matrix. The CSIT error variance is derived as $\sigma_{e,k}^2 \triangleq \mathbb{E}_{\mathbf{e}_k}\{\| \mathbf{e}_k \|^2\}$ [19], identical for all users, i.e. $\sigma_{e,k}^2 = \sigma_e^2$, and scalable as $\sigma_{e,k}^2 = \mathcal{O}(P^{-\varrho})$. The scaling factor $\varrho \in [0, \infty)$ indicates the quality of CSI at the BS in high signal-to-noise ratio (SNR) region [38]. For the extreme case of $\varrho = 0$, the CSIT quality remains invariant regardless of the SNR. On the other hand, as $\varrho \to \infty$, the error variance $\sigma_{e,k}^2 \to 0$, hence perfect CSIT is achieved. Thus, with a finite $\varrho > 0$, the CSIT quality is improved due to the increasing SNR. In this paper, we set the range $\varrho \in [0, 1]$ because $\varrho = 1$ corresponds to perfect CSIT from a DoF perspective [39].

At the receiver side, the received signal $\mathbf{y}_k$ at user k is formulated as

$$\mathbf{y}_k = \widehat{\mathbf{h}}_k^H \mathbf{x} + n_k, \tag{3}$$

where $n_k \sim \mathcal{CN}(0, \sigma_k^2)$ is the corresponding mean-zero additive white Gaussian noise (AWGN) at user k, with the variance σ_k^2.

In a one-layer RSMA system, after user k has received signal $\mathbf{y}_k$, the common stream s_c is firstly decoded by treating all private streams from K users as noise. These noises are filtered using the SIC technique placed at each receiver. Therefore, the signal-to-interference-plus-noise ratio (SINR) $\gamma_{c,k}$ for transmitting common stream to user k is

$$\gamma_{c,k} = \frac{\mathbf{p}_c |\widehat{\mathbf{h}}_k^H \mathbf{z}_c|^2}{\sum_{k=1}^{K} \mathbf{p}_{p,k} |\widehat{\mathbf{h}}_k^H \mathbf{z}_{p,k}|^2 + \sigma_k^2}. \tag{4}$$

All users decode the common stream. To do this, the decoding rate for all users must not exceed the minimum decoding SINR of all users [40]. Hence, the transmission rate for the common stream for all users is

$$R_c = w. \log_2 \left(1 + \min\left\{\gamma_{c,k}\right\}_{k=1}^{K}\right), \tag{5}$$

where w is the channel bandwidth. Since each user has a distinct amount of messages to contribute to the common message stream, the transmission rate R_c is the sum of all the portions which each k user allocated to transmit the common message. Denote C_k is the common rate allocated by user k, we have

$$\sum_{k=1}^{K} C_k \leq R_c. \tag{6}$$

Ideally, after user k has successfully decoded the common stream using SIC and formed a common message $\widehat{W}_{c,k}$, this message is then removed from the total received message. However, there could be errors that occurred during this

process and the common message signal has not been detached completely, resulting in the imperfect SIC condition. Therefore, when user k decodes its own private stream by treating private streams of other users as noise, interference still exists from the common stream. Thus, the transmission rate $R_{p,k}$ for the private stream at k-th user is computed as

$$R_{p,k} = w.\log_2\left(1+\gamma_{p,k}\right) \tag{7}$$

in which the SINR $\gamma_{p,k}$ of the private stream can be calculated as

$$\gamma_{p,k} = \frac{\mathbf{p}_{p,k}|\widehat{\mathbf{h}}_k^H\mathbf{z}_{p,k}|^2}{\sum_{j=1,j\neq k}^{K}\mathbf{p}_{p,j}|\widehat{\mathbf{h}}_k^H\mathbf{z}_{p,j}|^2+\sigma_k^2+\xi.\mathbf{p}_c|\widehat{\mathbf{h}}_k^H\mathbf{z}_c|^2} \tag{8}$$

where $\xi \in [0,1]$ corresponds to the imperfect SIC after decoding the common message [41], [42]. In overall, the total achievable rate R_k for user k is given by

$$R_k = C_k + R_{p,k}. \tag{9}$$

Subsequently, the private message $\widehat{W}_{p,k}$ of user k is extracted after the decoding of private stream $s_{p,k}$. Henceforth, user k reconstructs the intended message $\widehat{W}_k$ by combining $\widehat{W}_{c,k}$ and $\widehat{W}_{p,k}$, which should have the form similar to the original message W_k.

B. Video Traffic Model

Given that a discrete-time stochastic live IoMT streaming session at user k incorporates into the considered system model. At the beginning of timeslot t, the 5GMS server located at the gNB is caching $\Omega_k[t]$ contiguous video chunks to provide for user k. Obviously, $\Omega_k[t] = 0, t = 0$. Without loss of generality, the size in bits of a video chunk and the duration of a timeslot are fixed as Π and τ, respectively. In addition, denote the number and play time of video chunks downloaded by user k during timeslot t are $\Delta_k[t]$ and $\delta_k[t]$. Given that video bitrate $r_k[t]$ has been selected for the video stream in timeslot t, $\delta_k[t]$ can be calculated as

$$\delta_k[t] = \frac{\Pi}{r_k[t]}. \tag{10}$$

On the other hand, the number of video chunks continuously appended to the cache at the 5GMS server during timeslot t is given by $\frac{\tau}{\delta_k[t]}$. Hence, $\Delta_k[t]$ is given by

$$\Delta_k[t] = \begin{cases} \left\lfloor \frac{\tau R_k[t]}{\Pi} \right\rfloor & \text{if } \tau R_k[t] < \Pi\left(\Omega_k[t]+\frac{\tau}{\delta_k[t]}\right) \\ \Omega_k[t] + \left\lfloor \frac{\tau}{\delta_k[t]} \right\rfloor & \text{otherwise,} \end{cases} \tag{11}$$

where $\lfloor\cdot\rfloor$ is a floor function. Accordingly, the cache at the 5GMS server remains at the beginning of timeslot $t+1$ as

$$\Omega_k[t+1] = \Omega_k[t] + \frac{\tau}{\delta_k[t]} - \Delta_k[t]. \tag{12}$$

In a downlink live IoMT streaming, the video session begins when the 5GMS streaming server receives user video requests and detailed information about the network conditions. The process includes the server sending encoded video data with the requested quality, the transmission from the gNB to users, decoding the data, and finally adding to the video buffer for playing back. Regarding these processes, we introduce the terms *Buffering* and *Stalling* based on the 3GPP technical report on media streaming [43]. Video buffering includes initial buffering and re-buffering. The former term indicates the duration starting when the user triggers the video request until the video is being playback on user devices. The latter term, re-buffering, refers to the pre-loading of video chunks. On the other hand, video stalling occurs when the layout stops due to re-buffering, a user action, the end of video content, or a permanent failure. To further investigate the correlation of these terms to IoMT streaming systems, let the time for user k to download a video chunk be formulated as

$$d_k^c[t] = \frac{r_k[t].\delta_k[t]}{R_k[t]}. \tag{13}$$

The video chunk is downloaded to the user's device to be decoded by the decoder on the user side. After decoding the video chunk, it would be added to the buffer and ready for playback. As the video is played, the data is retrieved from the buffer and played in real time. The total time for user k to download all the video chunks within time slot t is:

$$d_k^t[t] = d_k^c[t].\Delta_k[t]. \tag{14}$$

Denote the buffered video time in the playback buffer of user k, i.e., current buffer size, at time slot t as $b_k[t]$ seconds. Whenever a video chunk is downloaded and decoded completely, it is cached to the playback buffer and the buffered video time increases $\delta_k[t]$ seconds. Intuitively, during a time slot t, a volume of $\Delta_k[t].\delta_k[t]$ video time has been added to the playback buffer while a volume of $\rho_k[t]$ video time has been played by the user, where $\rho_k[t]$ must not be greater than the length of a time slot. At the beginning of time slot $t+1$, $b_k[t+1]$ can be calculated as

$$b_k[t+1] = \max\{b_k[t] + \Delta_k[t].\delta_k[t] - \rho_k[t], 0\}. \tag{15}$$

Accordingly, a stalling event occurs if $b_k[t] + \Delta_k[t].\delta_k[t] < \rho_k[t]$ and the stalling time is equal to $\rho_k[t] - b_k[t] + \Delta_k[t].\delta_k[t]$. In this case, $b_k[t+1] = 0$ leads to a re-buffering event, which helps to cache a sufficient number of video chunks before continuing video stream playback.

In Dynamic Adaptive Streaming over HTTP, the video quality is chosen by the video server, which was adaptively adjusted according to the network condition of the user. Obviously, the perceived quality of video chunks can be modeled as a concave function of video bitrate $r_k[t]$, i.e., the higher the bitrate, the better quality can be achieved [31], following the principle of Adaptive Bitrate Streaming. Hence, the quality of the video chunk for user k is denoted as a concave function of bitrate, $\mathbf{Q}\{r_k[t]\}$. An example of video quality selection based on bitrate is available on an open-source "Big Buck Bunny" dataset [44] or in the work of [45]. For reference, the minimum and maximum resolutions in [44] are 360p and HD 1080p, respectively. One of the most important factors which directly impacts the user video experience is the variation in quality between two sequences of video chunks. The quality between them must align with each other, allowing the user to smoothly change the quality at the chunk boundaries if

necessary [46]. We formulate the adjacent quality variations as follows

$$V_k[t] = \mathbf{Q}\left\{r_k[t]\right\} - \mathbf{Q}\left\{r_k[t-1]\right\}, \quad t = 1, 2, \dots \tag{16}$$

The resolution fluctuation among chunks may cause unpleasant experiences for users. In particular, video resolution increases (i.e., $V_k[t] > 0$) provide a better experience for users and vice versa. In other words, the interval of V_k and the quality of video chunks in the next timeslot should be maximized.

C. Optimization Problem

The above equations have defined a discrete dynamic IoMT streaming model by highlighting the two important metrics which directly affect the QoE of users in downlink IoMT streaming. Overall, the QoE performance of user k can be expressed as follow

$$\mathbf{QoE}_k[t] = \alpha V_k[t] - (1-\alpha) d_k^t[t], \tag{17}$$

where the first and second terms are the video quality variation and the latency, respectively, which associate with a balance factor α. Subsequently, the QoE optimization problem for all users is formulated as

$$\begin{aligned} \max_{r_k[t], \mathbf{P}[t]} \quad & \sum_{k=1}^{K} \mathbf{QoE}_k[t] \\ \text{s.t.} \quad & \text{C1: } \mathbf{p}_c + \sum_{k=1}^{K} \mathbf{p}_{p,k} \leq P, \forall k \in \mathcal{K} \\ & \text{C2: } \sum_{k=1}^{K} C_k \leq R_c \\ & \text{C3: } C_k > 0, \quad \forall k \in \mathcal{K} \\ & \text{C4: } b_k[t] \geq 0, \quad \forall k \in \mathcal{K} \\ & \text{C5: } b_k[t] + \Delta_k[t].\delta_k[t] \geq \rho_k[t], \end{aligned} \tag{18}$$

where (C1) specifies the transmit power constraint for all users must not surpass the total available power at the BS; constraint (C2) use to ensure the total portion of all users' common rate is subjected to the total rate allocated for decoding the common stream; constraint (C3) requires the common rate assigned for each user to be positive. The buffering restriction is demonstrated by the constraint (C4) and (C5), in which (C4) is set to keep the buffer size cannot be a negative value, and (C5) is the stalling event.

In theory, the optimization problem at (18) could be addressed using the dynamic programming method [47]. Nonetheless, because of the non-convex linked rate expressions and significant computational complexity, traditional optimization tools are inefficient and overly complex for solving the problem. Particularly, the channel gains $\mathbf{h}_k$ between user k and BS fluctuates over time, resulting in uncertainty and unstable dynamic of the channel. Furthermore, the consideration of imperfect CSIT and SIC in RSMA implies the problem cannot be solved directly due to the lack of prior knowledge regarding the channel state distribution and network conditions. To this end, we proposed a DRL-based algorithm for a downlink RSMA-based IoMT streaming system, capitalizing on the ability of DRL to handle non-convex issues.

IV. Proposed Solution

In this section, we provide a DRL technique called *DDPG-RMAVS*, which is based on DDPG to solve the optimization problem. First, we will discuss the fundamentals of DDPG and why DDPG is best suited for the solution. We then evaluate and define our problem as a MDP framework, and demonstrate how to apply the *DDPG-RMAVS* onto our model.

A. Preliminaries

The IoMT streaming environment comprises of a high-dimensional state space and an action space. Therefore, to effectively extract features from complex state space $\mathcal{S}$ and learn policies for action space $\mathcal{A}$, deep function approximators in DDPG were leveraged. Particularly, DDPG used a straightforward actor-critic architecture and parameterized the actor-network $\mu(s|\theta^\mu)$ with weight parameter sets θ^μ, similarly to the critic network $Q(s, a|\phi^Q)$ with ϕ^Q. The actor represents the policy μ that deterministically selects action a based on state s using parameters vector θ^μ, and the Q-value function in the critic determines the performance of the action chosen by the actor. The optimal accumulative Q-value of the critic can be determined using the Bellman equation:

$$Q^*\left(s, a|\phi^Q\right) = \max_{a \in \mathcal{A}, r, s' \sim E} \left[r(s,a) + \gamma \max_{a'} Q^*(s', a'|\phi^Q) \right] \tag{19}$$

in which r indicates the reward received from the environment after executing action a at state s; $\gamma \in [0, 1]$ is the discount factor to reduce the importance of future rewards; s' and a' are the next state and the next chosen action to be executed.

The DDPG algorithm interpolates between the optimal policy and the Q-Learning approach. The main objective of policy optimization is to identify a policy that deterministically maps states to a specific action that maximizes the expected return $J(s|\theta^\mu)$, i.e, the expected reward starting from s, which is done by the actor. As in Q-Learning, the critic is learnt using the Bellman equation in (19). The actor is updated with respect to the actor parameters following the policy gradient [48]:

$$\begin{aligned} \nabla_{\theta^\mu} J(\theta^\mu) &= \mathbb{E}\left[\nabla_{\theta^\mu} Q\left(s, a|\phi^Q\right)\right] \\ &= \mathbb{E}\left[\nabla_a Q\left(s, a|\phi^Q\right) \nabla_{\theta^\mu}\left(a\right)\right] \\ &= \mathbb{E}\left[\nabla_a Q\left(s, a|\phi^Q\right) \nabla_{\theta^\mu}\left(\mu'(s)\right)\right] \end{aligned} \tag{20}$$

Since DDPG is an off-policy algorithm, the on-policy exploration ability of DDPG can be limited and actions may not be able to find useful learning signals at the start of training. To overcome this, the original actor policy $\mu(s|\theta^\mu)$ is added with a noise process $\mathcal{N}$, i.e,

$$a = \mu(s|\theta^\mu) + \mathcal{N} \tag{21}$$

in which the noise $\mathcal{N}$ can be generated based on the OU process [49], or mean-zero AWGN [50]. Similar to DQN, a replay buffer $\mathcal{D}$ is also introduced for DDPG to ensure that it can learn from a variety of experiences and to decorrelate datasets provided to learn the probability distribution. The replay buffer is a finite-sized cache that consists of many tuples in the form of $\langle s_t, a_t, r_t, s_{t+1} \rangle$. These tuples contain the information of state s_t, the action a_t to be executed at time t,

the reward r_t it perceived after the action, and the state s_{t+1} after the action has been applied onto the environment. After each time slot t, a tuple is uniformly sampled and stored in D. The mini-batch $\mathcal{B}$ of the replay buffer is then used as input for training Deep Neural Networks (DNN) models.

Another term worth being mentioned is the target networks, which includes a target actor network $\mu_{\theta_{\text{targ}}}$ and a target critic network $Q_{\phi_{\text{targ}}}$. These target networks ensure the chance of convergence and stabilize the training for the original actor-critic algorithm. Subsequently, we can retrieve the tuples in batch $\mathcal{B}$ to calculate the mean-squared Bellman error (MSBE),

$$L(\phi^Q) = \mathbb{E}_{s_t,a_t,r_t,s_{t+1}\sim\mathcal{B}}\left[\left(Q(s_t,a_t|\phi^Q) - y_t\right)^2\right] \quad (22)$$

where $Q(s_t,a_t|\phi^Q)$ is the Q-value of action a_t chosen at state s_t by the critic based on the parameters ϕ^Q. The target y_t is the optimal accumulative $Q^*\left(s,a|\phi^Q\right)$ calculated based on Bellman equation (19) and can be formulated as

$$y_t = r(s_t,a_t) + \gamma Q_{\phi_{\text{targ}}}\left(s_{t+1}, \mu_{\theta_{\text{targ}}}(s_{t+1})\right) \quad (23)$$

The objective is to minimize the MSBE loss, i.e, minimize (22) and make the Q-function $Q(s_t,a_t|\theta_Q)$ to be as close as the target y_t as possible. For the sake of stabilization during MSBE minimization, the parameters of target actor θ_{targ} and the target critic ϕ_{targ} are updated once per main network update:

$$\theta_{\text{targ}} \leftarrow \rho\theta_{\text{targ}} + (1-\rho)\theta^\mu$$
$$\phi_{\text{targ}} \leftarrow \rho\phi_{\text{targ}} + (1-\rho)\phi^Q \quad (24)$$

where the hyperparameter $\rho \ll 1$, sometimes is referred to as the target network update rate.

B. Markov Decision Process framework

In this sub-section, we transform our system into an MDP framework for DRL agents. In a IoMT streaming system, the characteristics of video chunks are decided by the streaming server based on the observations it received from the transmission network and the feedback from users. Hence, we considered the streaming server as the DRL agent and the downlink IoMT streaming process as an environment. We reformulate the optimization problem (18) into MDP by defining a tuple $(\mathcal{S},\mathcal{A},r,\gamma)$ which includes a state space $\mathcal{S}$, an action space $\mathcal{A}$, a reward function ($r:\mathcal{S}\times\mathcal{A}\rightarrow\mathbb{R}$), and a discount factor γ, respectively.

State space: In practice, the IoMT streaming server consults the network conditions to adaptively select the optimal quality for the video chunk. For the RSMA transmission, the factor which can affect the selection decision is the downlink channel state $\widehat{\mathbf{H}} = \left[\widehat{\mathbf{h}}_1, \widehat{\mathbf{h}}_2, ..., \widehat{\mathbf{h}}_K\right]$. Furthermore, the buffer $\mathbf{b}$ also plays a crucial role in choosing the video quality, as a larger buffer size at t allows the agent to be more flexible in selecting higher video quality, even when stringent achievable rate occurs. Concretely, at each time step t, the agent observes the changes in these factors and feeds it into the state space, which can be described as a tuple:

$$s_t = \langle \widehat{\mathbf{H}}(t), \mathbf{b}(t)\rangle \quad (25)$$

in which $\widehat{H}(t) \in \mathbb{C}^{M\times K}$ is the imperfect CSIT channel state of K users and $\mathbf{b}(t) \in \mathbb{C}^K$ is the available buffer size at time t of all K users. The state s_t will have the dimension of $((M+1)\times K)$.

Action space: Given the state s_t, the DRL-based model determines an action based on policy μ. In particular, the agent seeks the quality for the video chunks in timeslot t $\mathbf{Q}\{\mathbf{r}(t)\} \in \mathbb{C}^K$ for all K users and the transmit power matrix $\mathbf{P}(t) \in \mathbb{C}^{M\times(K+1)}$ of K users in addition with one common message. Since the video quality is the concave function of video bitrate, the video quality part can be replaced by the video bitrate. The action space a_t is a tuple:

$$a_t = \langle \mathbf{r}(t), \mathbf{P}(t)\rangle \quad (26)$$

where $\mathbf{r}(t) \in \mathbb{C}^K$ is the bitrate vector of all K users. The action a_t has a dimension of $(K + M \times (K+1))$.

State Transition Probability: A state transition probability alters the state of an environment between consecutive time steps. Following the Markov property, the state transition probability can be expressed as

$$P(s_{t+1}|s_t,a_t) = P(\widehat{\mathbf{H}}_{t+1}, \mathbf{b}_{t+1}|\widehat{\mathbf{H}}_t, \mathbf{b}_t, \mathbf{r}_t, \mathbf{P}_t) \quad (27)$$

where the state s_{t+1} of the next time step $(t+1)$ is only depend on the current state s_t and the chosen action a_t at time step t.

Reward: As mentioned in the previous subsection, in each time slot, the agent tries to find an optimal policy $\mu*$ which maximizes the expected reward, for which the Q-value function can be formulated as in [51]

$$Q(s_t,a_t|\phi^Q) = \mathbb{E}\left[\sum_{t=1}^{T}\gamma^{t-1}(r_t|s_t,a_t)\right] \quad (28)$$

where the optimal policy is mathematically expressed as

$$\mu* = \arg\max_{\mu} Q(s_t,a_t|\phi^Q), \quad \forall s_t \in \mathcal{S}$$

In our proposed system, the optimization objective is to maximize the QoE of all users throughout the entire IoMT streaming session. Therefore, the reward r_t can be calculated as in equation (18)

$$r_t(s_t,a_t) = \sum_{k=1}^{K}(\mathbf{QoE}_k[t] - p_k[t]) \quad (29)$$

We introduce the term $p_k[t]$ as the penalty received for action a_t that does not satisfy the constraint (C5) in (18). This constraint is set up to avoid an empty buffer and a stalling event occurs. In particular, the penalty $p_k[t]$ for user k at time step t is defined as follows:

$$p_k[t] = \Phi\left(b_k[t] + \Delta_k[t].\delta_k[t], \rho_k[t]\right) \quad (30)$$

where the function Φ is equal to 10 if the calculation of the buffer at the next time step $b_k[t] + \Delta_k[t].\delta_k[t]$ is less than the video playtime $\rho_k[t]$, which leads to an empty buffer at next time step $(t+1)$ and re-buffering event occurs. Conversely, Φ is set to be 0 if the buffer size is larger than $\rho_k[t]$, ensuring the streaming session can continue without any disruption. With this penalty setup, the agent would have to consider the trade-off between the video quality $\mathbf{Q}\{r_t\}$ with the buffer size $\mathbf{b}_t$ in low total achievable rate R_k scenario.

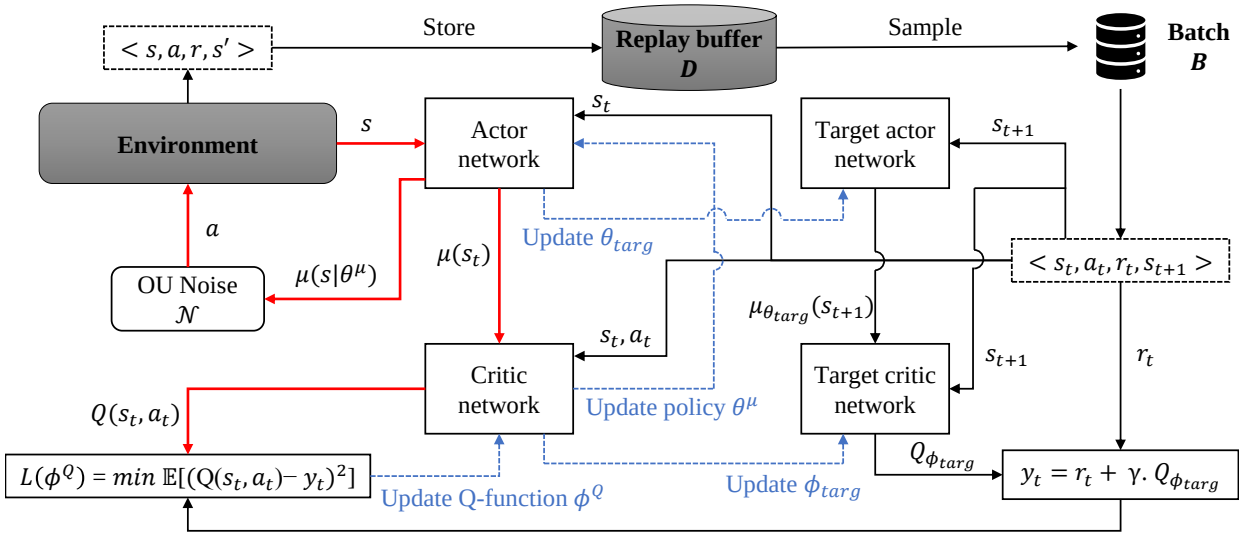


Fig. 3: *DDPG-RMAVS* algorithm.

C. Algorithm Methodology

At the beginning of each timestep t, the agent located in the gNB generates an observation s_t as in (25), and then determine the action a_t using policy μ. Subsequently, the action is performed onto the environment, which in turns feedback the next state s_{t+1} and the reward r_t. The reward is defined in (29), which consist for the QoE and the penalty term. As previously mentioned in IV-A, the policy μ is structured to select which action results in highest accumulative reward r_t. Hence, our objective is to maximize the QoE by increase video quality and minimize latency while avoiding the penalty.

The policy parameters θ^μ in actor network determine action a_t based on (21), henceforth create a tuple of $(\widehat{\mathbf{H}}[t], \mathbf{b}[t], \mathbf{r}[t], \mathbf{P}[t])$. From $\widehat{\mathbf{H}}[t]$ and $\mathbf{P}[t]$, the total achievable rate $R[t]$ can be calculated using (4)$\rightarrow$(9). Furthermore, as the video bitrate $r[t]$ is selected within the action a_t, the play time $\delta_k[t]$ is determined in (10), as well as the number of downloaded chunks $\Delta_k[t]$ in (11) and cached chunk $\Omega_k[t+1]$ in (12). These variances are used to calculate the latency $d^t[t]$ and the buffer capacity $b_k[t+1]$. Moreover, the video quality $\mathbf{Qr}[t]$ is the concave function from video bitrate vector $\mathbf{r}[t]$, hence we can also derive the adjacent quality variation $V[t]$. From $V[t]$ and $d^t[t]$, QoE can be measured in (17), therefore the reward r_t is calculated using the equation (29).

From the aforementioned analysis, the action a_t are directly impact the reward r_t, as high bitrate action $\mathbf{r}[t]$ would resolve on maximizing the quality variation term $V[t]$ and reverse. Additionally, the power allocation matrix $\mathbf{P}[t]$ greatly improves the achievable rate R, therefore allowing RSMA to download higher bitrate video. The action also needs to consider the penalty term, for keep generating $\mu(s|\theta^\mu)$ with high $r[t]$ can inevitably causing the buffer to drain, hence stalling events occur.

D. Complexity Analysis

This section investigates the complexity of the proposed *DDPG-RMAVS* algorithm on optimizing video latency and quality variation. Our implementation relies on fully connected DNN for actor and critic networks, hence we derive the computational complexity based on the DNN workload. In IV-B, we have defined the dimension of state space $\mathcal{S} = ((M+1)\times K)$ and the action space $\mathcal{A} = (K+M\times(K+1))$. As can be seen from Fig.4, the DNN structure of actor network and critic network are identical, each contains of one input layer, two hidden layers, and one output layer.

Denote n_1^a and n_2^a as the number of neuron nodes in the first and second hidden layer of actor network, respectively. In the training process, the agent performs backpropagation and feed-forward propagation to determine action a_t from state s_t. Hence, the complexity of the actor network in training for each step is formulated as in [52]

$$\mathcal{O}_{train}^a = \mathcal{O}\left((\mathcal{S})^2 + (n_1^a)^2 + (n_2^a)^2 + (\mathcal{A})^2\right) \tag{31}$$

Correspondingly,we express the computational complexity for the critic network as

$$\mathcal{O}_{train}^c = \mathcal{O}\left((\mathcal{S}+\mathcal{A})^2 + (n_1^c)^2 + (n_2^c)^2\right) \tag{32}$$

with n_1^c and n_2^c are the nodes available at the two hidden layers in critic network. Furthermore, because the DNN structure of the target actor network and target critic network are similar to the main networks, in addition to the utilization of both the main and target networks during training process, the computational complexity of *DDPG-RMAVS* during training is computed as

$$\mathcal{O}_{train} = 2 \times E \times S \times \mathcal{B} \times (\mathcal{O}_{train}^a + \mathcal{O}_{train}^c) \tag{33}$$

where E to be the number of training episodes and S be the number of steps in an episode.

For the decision-making process, the backpropagation in the main actor network is omitted due to the joint action selection to interact with environment. Henceforth, the computational complexity during the decision-making is only determined

by the forward propagation of neural network, which can be computed as

$$\mathcal{O}_{dm} = E \times S \times \mathcal{B} \times (\mathcal{S}n_1^a + n_1^a n_2^a + n_2^a \mathcal{A}) \quad (34)$$

We observe that the complexity of the proposed algorithm is polynomial and practical in scenarios where the environment can be dynamically scaled up. Considering K-dimensional input, the computational complexity of such alternative optimization like semi-definite relaxation [53] can up to $\mathcal{O}(K^6+1)$, whereas for *DDPG-RMAVS*, the complexity rises only to K^2 during training process and reduces to K in decision-making, which greatly reduces the execution time and computational resources.

The pseudo-code of *DDPG-RMAVS* algorithm for optimizing QoE in RSMA-based IoMT streaming system is shown in Algorithm 1. Fig.3 demonstrates the general working principles of the proposed algorithm. Fig.4 illustrates the procedure of generating the action a from actor network and $Q(s_t, a_t)$ from critic network (the red line operation in Fig.3). The denotations of parameters used in the algorithm are clearly explained in Table II.

Algorithm 1 *DDPG-RMAVS*

Initialize: Initialize the buffer size $\mathbf{b}$ and number of video chunks cache at 5GMS server Ω of state $s(t_0)$ to be 0.
Initialize the weights θ^μ and ϕ^Q of DNN actor network $\mu(s)$ and critic network $Q(s)$, respectively.
Initialize target weight $\theta_{\text{targ}} \leftarrow \theta^\mu$ of the target actor μ' and the target weight $\phi_{\text{targ}} \leftarrow \phi^Q$ of the target critic Q'.
Initialize replay buffer size D and mini-batch $\mathcal{B}$.
for $episode \leftarrow 1$ to E **do**
    Initialize state $s[t_0]$ based on (25).
    while $t \leq S$ **do**
        Agent in BS observe state $s[t]$
        Actor generates action $a[t]$ according to (26)
        Compute reward $r[t]$ according to (29)
        Observe next state $s[t+1]$ based on (15)
        Store tuple $(s[t], a[t], r[t], s[t+1])$ in buffer D
        Sampling batch $\mathcal{B}$ tuples from D to train DNN
        Update ϕ^Q by minimizing loss in (22)
        Update θ^μ from policy gradient in (20)
        Update target networks based on (24)
        t++
    end while
end for
Return θ^{μ^*}

V. Performance Evaluation

A. Simulation Settings

The simulation is conducted on a GPU-based server with NVIDIA GeForce RTX 3090 Ti. The CPU is Intel(R) Core(TM) i9-12900K 3.20 GHz with 64G RAM. The software environment is set with Anaconda using Pytorch 2.0.0, CUDA 11.8, Python 3.9.16, and Gym 0.26.1. Using PyTorch and Python programming, we generate the IoMT streaming environment and train the agent.

TABLE II: Simulation parameters

Parameter	Value
Number of antennas on BS, M	14
Number of users, K	10
Base Station bandwidth, w	1 MHz
Power, P	23 dBm
Noise variance, σ^2	-170 dBm/Hz
SIC error, ξ	0.1
Imperfect CSIT, ϱ	0.8
Chunk size, Π	3 Mb
Timeslot duration, τ	3 seconds
Max/min bitrate of codec	0.1 Mbits/22 Mbits
Number of training episodes, E	2000
Number of steps per episode, S	700
Actor hidden layer nodes, n_1^a & n_2^a	1024 & 512 nodes
Critic hidden layer nodes, n_1^c & n_2^c	512 & 256 nodes

1) RSMA System Settings: The environment comprises one BS and K number of users in downlink IoMT streaming transmission using RSMA. The number of antennas on BS is set to be $M = 14$ simultaneously serving $K = 10$ single-antenna users. In RSMA, each user is allocated a portion of available bandwidth, which is divided among users in a fixed manner, hence the bandwidth is assumed to be the same for all users regardless of the number of users. We assigned a bandwidth of $w = 1$ MHz to evaluate the algorithm's efficacy in scenarios with restricted bandwidth and to eliminate any predisposition towards consistently high available rates, which would otherwise result in obtaining the maximum bitrate for every user. The noise variance σ^2 at the transmitter is selected to be -170 dBm/Hz and the total power P allocated by the BS for all users is 23 dBm.

The state of the environment is defined as in (25). Firstly, we attempted to randomly initialize the imperfect channel state matrix $\widehat{\mathbf{H}} = \mathbf{H} + \mathbf{E}$ by identifying the perfect channel state matrix $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_K]$ and the error matrix $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_K]$. Each error channel vector $e_k \in \mathbf{E}$ is identically distributed and drawn from complex Gaussian distribution, $\mathcal{CN}(0, \sigma_{e,k}^2)$ with $\sigma_{e,k}^2 = P^{-\varrho}$. For the simulation, we set $\varrho = 0.8$. Subsequently, we initialize the buffer size $\mathbf{b} \in \mathbb{C}^K$. At $t = 0$, the buffer is empty due to no video data having been downloaded, i.e., $\mathbf{b}[t_0] = 0$. Additionally, for the portion of C_k in (6), we assumed that each user has the same portion, i.e., $C_1 = C_2 = \ldots = C_K$.

2) IoMT Streaming System Settings: For IoMT streaming characteristics, we determined the chunk size $\Pi = 3.10^6$ bits and the time duration for each timeslot $\tau = 3$. We trained the agent for $S = 700$ for each episode. Additionally, we assumed the maximum video bitrate supported by the video codec is 22 Mbps and the minimum bitrate is 100 Kbps. The video bitrate is also sorted into different ranges that are identical for each distinct video quality based on Youtube live streaming bitrate selection guide [54]. After the actor-network of DDPG has output the bitrate for time slot t, the bitrate is assigned to be equal to the closest relevant bitrate minimum range of video quality. There are 10 different quality categories considered in this paper, which are 360p, 480p, 720p, 720p @60fps, 1080p, 1080 @60fps, 1440p @30fps, 1440p @60fps, 4K @30fps, and 4K @60fps. For example, based on the CSIT and the

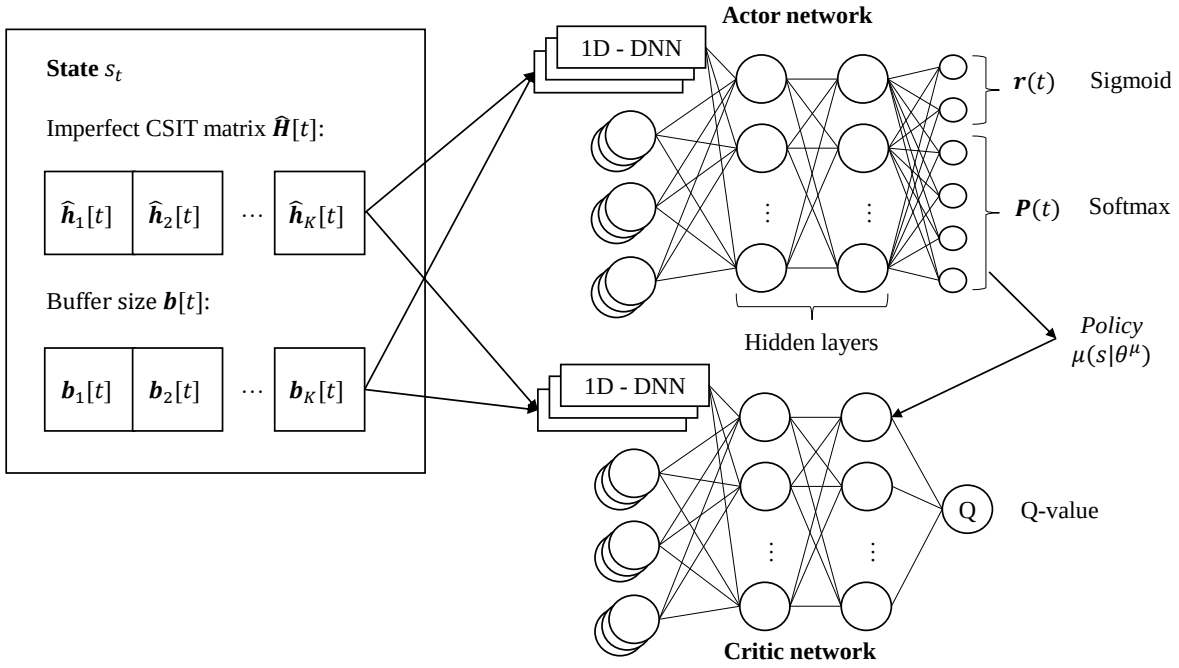


Fig. 4: Actor-Critic network interaction.

buffer size of user-k at timeslot t, the actor model predicts the appropriate bitrate $r_k[t]$ to be 2.67 Mbps. According to [54], this bitrate falls into the category of 720p @60fps and 1080p, of which the bitrates are 2.25 Mbps and 3 Mbps, respectively (we only consider the minimum range). We will assign the bitrate into 720 @60fps quality, i.e.,$r_k[t] = 2.25$ Mbps, which is the lower range to ensure the system does not violate the buffer constraint.

The system level and IoMT streaming level simulation parameters are summarized in Table II.

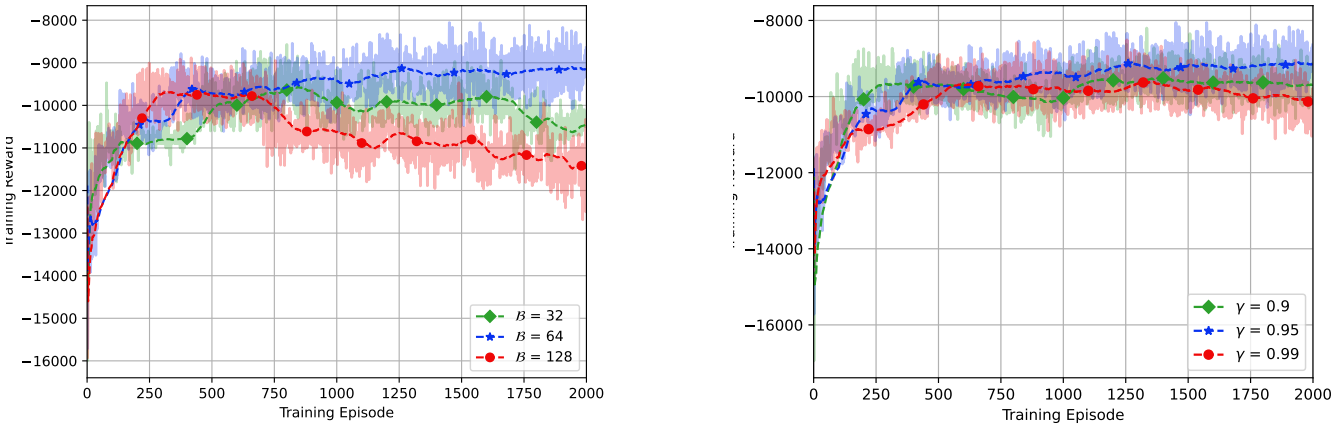
B. Convergence Analysis

To improve the certainty of the model study across different random CSIT changes, we performed a Monte Carlo simulation for over 2000 episodes. To begin, because the performance of the algorithm is subject to hyper-parameter tuning, we have investigated algorithm convergence by experimenting with different values of hyper-parameters. We examined the impact of hyper-parameter tuning on the proposed algorithm based on changes in training rewards.

Initially, we trained our suggested model using varying batch sizes $\mathcal{B}$. We experimented with three batch sizes: $\mathcal{B} = \{16; 32; 64\}$, and the outcomes can be seen in Fig.5a. The model experiences the slowest convergence with batch size $\mathcal{B} = 32$, converging around 800 episodes. On the other hand, the largest batch size, $\mathcal{B} = 128$, leads to the quickest convergence, occurring after roughly 300 episodes. However, as training continues, the training reward for $\mathcal{B} = 128$ gradually decreases compared to the other two due to diminished stochasticity and a lack of exploration caused by the excessively large batch size. In particular, the training rewards at episode 2000 with $\mathcal{B} = 128$ is approximately -11500, which is about 8.7% worsen than $\mathcal{B} = 32$ and 20.87% than $\mathcal{B} = 64$. The most stable batch size appears to be $\mathcal{B} = 64$, as its convergence is relatively similar to that of $\mathcal{B} = 128$, occurring around 400 episodes, but be able to achieve the highest final training reward among the three sizes.

We examined three different discount factors for our model, denoted as $\gamma = \{0.9; 0.95; 0.99\}$. As the discount factor approaches to the value of 1, the model places greater emphasis on accumulating future rewards instead of focusing on short-term gains. The results in Fig.5b reveal that the three discount factors produce relatively similar outcomes. However, the most stable and consistent convergence is achieved with $\gamma = 0.95$. With this discount factor, the model begins to converge after 400 episodes, and the subsequent episodes display less fluctuation compared to those with $\gamma = 0.99$ and $\gamma = 0.9$.

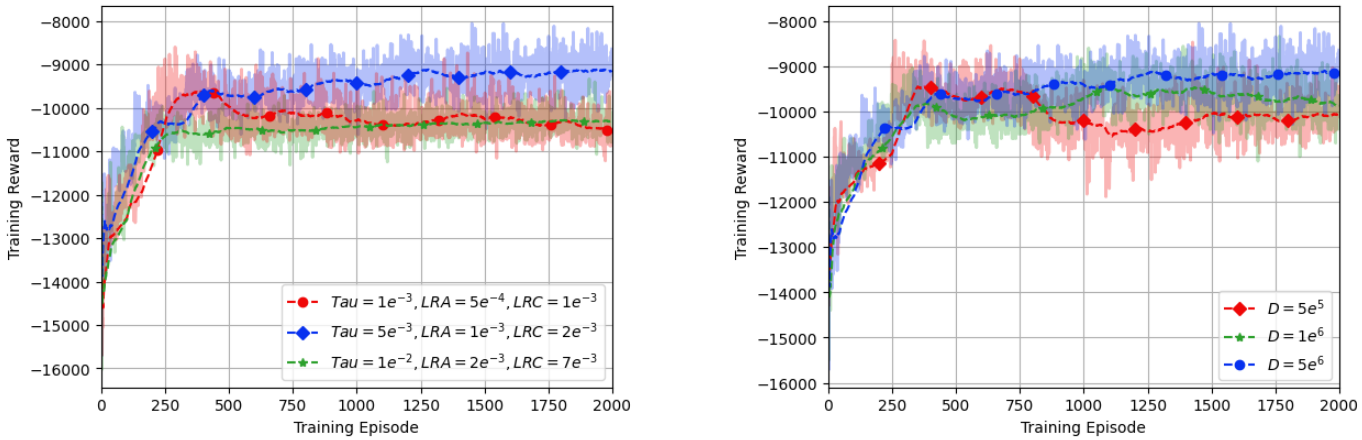
In Fig.6a, we demonstrate the impact of the actor-network learning rate (LRA), critic network learning rate (LRC), and soft target update rate (Tau) on the model. We selected three sets of rates: $(Tau, LRA, LRC) = (5e^{-3}, 1e^{-3}, 2e^{-3})$; $(1e^{-2}, 2e^{-3}, 7e^{-3})$; $(1e^{-3}, 5e^{-4}, 1e^{-3})$, respectively. Upon initial observation, it is apparent that higher learning rates enable the model to converge more rapidly. For $Tau = 1e^{-2}$, the model converges after just 250 episodes. However, the training rewards at the convergence point are the lowest compared to the other rates and barely increase afterward, suggesting the possibility of overfitting. Conversely, with $Tau = 1e^{-3}$, the model converges around 450 episodes and achieves the highest training rewards. Yet, as training continues, the rewards decrease and ultimately result in the lowest rewards. For $Tau = 5e^{-3}$, the simulation results indicate that convergence occurs after 350 episodes, and the learning process is adequate, as the training rewards exhibit acceptable fluctuations for exploration while the overall reward continues to increase and attains the highest reward rate among the three sets. Therefore, we chose the moderate rate set $(5e^{-3}, 1e^{-3}, 2e^{-3})$ as the



(a) Batch sizes

(b) Discount factor

Fig. 5: Training rewards on different batch sizes and discount factor.



(a) Learning rates

(b) Replay buffer sizes

Fig. 6: Training rewards on different learning rates and replay buffer sizes.

learning rates for our proposed algorithm.

In our final analysis, we assessed the influence of varying replay buffer D sizes on the overall training reward. Fig.6b displays the training rewards after training our model using three different replay buffer sizes: $D = \{5e^5; 1e^6; 5e^6\}$. Our simulation results indicate that with $D = 5e^5$, the model converges most rapidly, but the training rewards exhibit fluctuations and decrease as training continues. This may indicate that the model is not learning sufficiently from its past experiences due to the limited buffer size. For $D = 1e^6$ and $D = 5e^6$, the convergence is relatively similar, occurring at around 320 episodes. Nevertheless, the buffer size of $5e^6$ appears to yield larger rewards and exhibit greater stability than $D = 1e^6$.

From the above analysis, we have come to the conclusion of some optimal hyper-parameters which we would utilize for *DDPG-RMAVS* model in order to achieve convergence: Batch size $\mathcal{B} = 64$, discount factor $\gamma = 0.95$, learning rates set $Tau = 5e^-3$, $LRA = 1e^-3$, $LRC = 2e^-3$, and replay buffer size $D = 5e^6$.

C. Balancing Parameter Selection

We have introduced the balance factor α in equation (17) as a critical parameter with a significant impact on the performance of *DDPG-RMAVS*. The value of $\alpha = [0, 1]$ adjusts the priority of the reward function between video quality and latency term. Given that the objective is to maximize the QoE (17), the *DDPG-RMAVS* algorithm tends to prioritize maximizing the video quality term over the latency term when α is closer to 1, due to the higher rewards associated with quality, and reverse. The selection of an appropriate α value relies on user preferences. However, it is important to consider the trade-off between video quality and latency, as it can also impact other crucial metrics such as overall video quality, delay, and buffer capacity. The influence of different α values on these metrics is demonstrated in Fig.7.

In this analysis, we investigate the impact of different α values on different metrics. The analysis is presented using a double error graph in Fig.7a, where α values range from 0.0 to 1.0. The blue line represents video quality, ranging from the score of 0.5 (lowest quality) to 5.0 (highest quality),

whereas the red line represents video latency in seconds. The data is collected from the performance of the *DDPG-RMAVS* algorithm after 2000 episodes of training, and each data point represents the average K users. Examining the graph, it becomes evident that as α approaches 0.0, the video delay reaches its minimum value, averaging at 1.9 seconds. Conversely, the video quality also reaches its lowest point, approximately around 1.4 score, indicating an average quality close to 720p. As the α value increases, both video quality and video delay experience an upward trend. The video quality score rises from 1.4 at $\alpha = 0.0$ to 3.4 at $\alpha = 1.0$, indicating an improvement in quality. Simultaneously, the average delay increases from approximately 1.9 seconds at $\alpha = 0.0$ to 2.55 seconds at $\alpha = 1.0$. These findings demonstrate the trade-off between video quality and latency, with an increase in α leading to improved quality but also resulting in higher latency values.

We further investigate the influence of different α values on the buffer capacity. As previously discussed, a higher α value prioritizes the video quality term in the reward function. However, maximizing video quality under constrained network conditions can lead to increased buffer drain. In Fig.7b, we assess the average buffer capacity in seconds for 10 users after the streaming session. It is noteworthy that the buffer capacity gradually decreases as the α value increases. At $\alpha = 0.0$, the buffer capacity reaches a peak of approximately 2400 seconds, whereas it decreases to around 100 seconds at $\alpha = 1.0$. This indicates that when α is low, the *DDPG-RMAVS* model transmits the video signal with the lowest video quality, resulting in a smaller data size and minimal delay. Consequently, the video signal consistently feeds into the buffer.

Based on the aforementioned analysis, the trade-off between maximizing video quality and minimizing video delay is crucial for the model's performance. To ensure stability and overall satisfaction in QoE for all users, we determine an appropriate α value of $\alpha = 0.4$ for further implementation. With $\alpha = 0.4$, the video delay averages at approximately 2.2 seconds, while the video quality achieves a score of around 2.5, equivalent to a resolution of 1080p. Additionally, the buffer capacity after the streaming session is approximately 1550 seconds, providing a sufficient amount to ensure smooth streaming throughout the entire session with minimal interruptions.

D. Perfect vs. imperfect SIC-CSIT

We further discuss about the impact of imperfect SIC-CSIT to the convergence of *DDPG-RMAVS* algorithm. As previously mentioned in III-A, the imperfect SIC occurs during the decoding of private stream, hence the SIC noise were added in the calculation of private stream SINR $\gamma_{p,k}$ in (8). The common message noise term $\xi.\mathbf{p}_c|\widehat{\mathbf{h}}_k^H\mathbf{z}_c|^2$ were scaled by the factor $\xi \in [0,1]$, in which $\xi \to 0$ implies perfect SIC scenario and all the common message has been successfully decoded before decoding the private stream. Reversely, $\xi \to 1$ illustrates an extreme case of imperfect SIC in which user k cannot decode its common message. An increase of ξ would directly reduces $\gamma_{p,k}$, leads to a decrease in total achievable rate R_k.

On the other hand, the imperfect CSIT exists in the error channel matrix $\mathbf{E}$, as it represents the inconsistency of the channel estimation. In III-A, we showed that the error variance σ_e^2 were derived from $\mathbf{E}$ and scalable as $\sigma_e^2 = \mathcal{O}(P^{-\varrho})$, where $\varrho \in [0,1]$. For simulation, we have selected $\varrho = 0.8$. Since $P = 23$ dBm, the value of $\sigma_e^2 \approx 0.08$. As σ_e^2 is an uniform Gaussian distribution of perfect channel matrix $\mathbf{H}$, we can calculate the distribution of $\mathbf{E} = [-0.5, 0.5]$ of $\mathbf{H}$. Concretely, at each timestep, the channel matrix $\mathbf{H}$ is randomly increase or decrease an amount of $\mathbf{E} = \pm 50\%\mathbf{H}$.

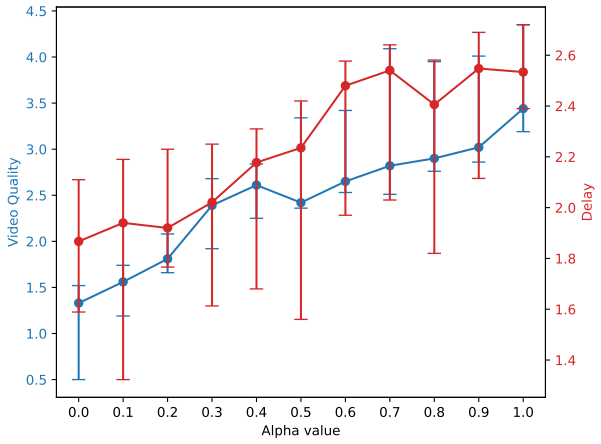
In Fig.8, we have compared the reward and the QoE after training *DDPG-RMAVS*. Recall that for $\xi = 0\%$ and $\varrho = 1$, perfect SIC-CSIT is achieved. As can be seen from Fig.8a, the proposed algorithm can achieve highest rewards when SIC and CSIT are perfect. As the SIC and CSIT are more imperfect, the rewards decreases rapidly. Similarly, Fig.8b reflects the impact of perfect SIC-CSIT on the average QoE for K users, where the highest QoE is achieved when the channel is perfect. An noticeable point is, the impact of imperfect CSIT affects more on the performance of *DDPG-RMAVS* than imperfect SIC, as despite the worst imperfect SIC, e.g. $\xi = 10\%$, if the CSIT is perfect, it can performs better than the case of perfect SIC but imperfect CSIT.

We also compare the video quality and video delay in different SIC-CSIT scenarios in Fig.9. It can be seen that with perfect SIC-CSIT, users can receive highest video quality while experience least video latency. The scenario of lowest video quality and longest latency is given to the case of $\xi = 10\%$, $\varrho = 0.8$, which is the worst imperfect SIC-CSIT.

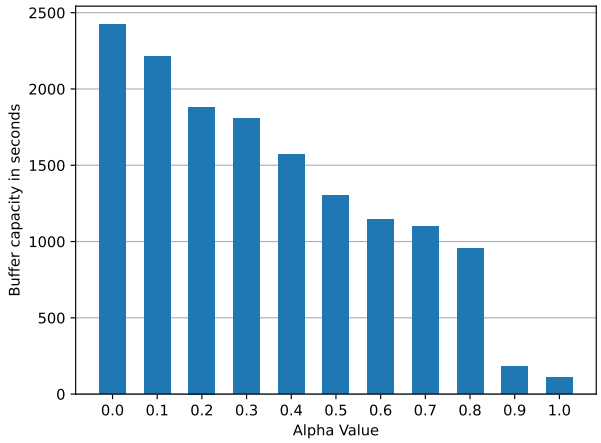
E. Performance of the Proposed Algorithm

In order to demonstrate the effectiveness of *DDPG-RMAVS*, we compare its performance with the following optimization algorithms:

- *RSMA-based Video streaming with Soft Actor-Critic (RVS)*: This algorithm is based on Soft Actor-Critic (SAC) algorithm. The working principle of SAC shares some similarities with DDPG since SAC is an off-policy algorithm, a policy gradient method that is using actor-critic architecture. However, several key advantages of SAC over DDPG are the ability to learn stochastic policies, incorporate entropy terms to encourage exploration, and be less prone to hyper-parameter tuning [55], [56]. Henceforth, theoretically, *RVS* can result in better performance compared to *DDPG-RMAVS* in learning continuous action space.
- *Advantage Actor-Critic RSMA-based Video streaming (A2C-RV)*: We employ the Advantage Actor-Critic (A2C) algorithm as documented in [57], [58]. Similar to DDPG, it follows an actor-critic structure, but the A2C operates as an on-policy algorithm, thereby focusing on learning from fresh experiences rather than relying on past events (e.g., replay buffer). In parallel with *RVS*, *A2C-RV* opts for actions by leveraging adjustable probability distributions and sampling methods. Instead of resorting to noise

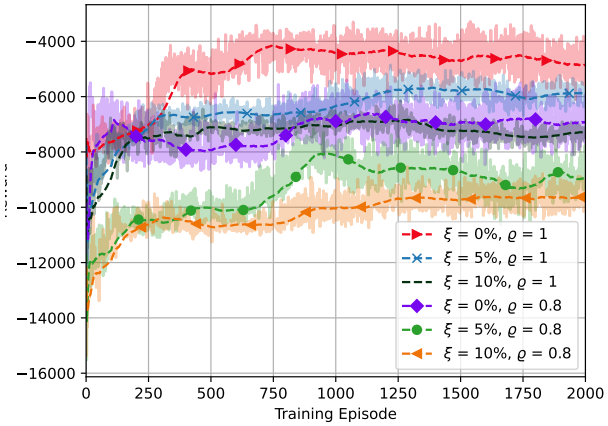


(a) Video quality and Delay

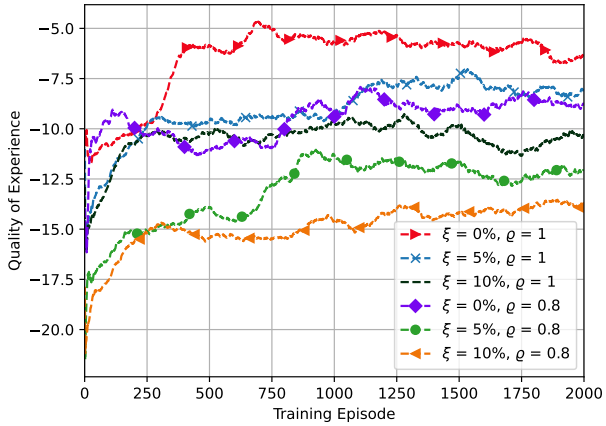


(b) Buffer capacity

Fig. 7: Affect of different alpha values on video quality, delay, and buffer size.



(a) Reward



(b) QoE

Fig. 8: Training rewards and QoE with different imperfect SIC-CSIT factors.

to promote exploration, it uses the inherent randomness from the action distribution.

- *Deep Q-Learning Video streaming RSMA-based System (DQN-RV)*: This approach is constructed based on DQN algorithm. As *DDPG-RMAVS* is similar to *DQN-RV* as both use the Q-learning technique, the core difference is the actor-critic architecture, as *DQN-RV* does not have it. For this reason, *DQN-RV* is more suitable for discrete action space tasks, rather than continuous tasks.
- *Greedy Video streaming RSMA-based System (GVRS)*: In *GVRS*, we utilize the Greedy method, a traditional optimization algorithm. Here, the continuous action space is transmuted into a discrete one, where the actions, referenced in (26), are selected from a discrete collection of available bitrates and power settings with the aim of optimizing (17). To tackle the potential issue of action value violation, we promptly normalize the power value to fall within $[0, 1]$, and adjust the bitrate to align with the set of available options stated in V-A2.

The parameters used in the four aforementioned algorithms are also similar to *DDPG-RMAVS*, which has been listed in Table II. Additionally, since *RVS* and *A2C-RV* are actor-critic architecture-based, the number of hidden layers and nodes on both actor network and critic network are identical to *DDPG-RMAVS*. For *DQN-RV*, the number of hidden layers and nodes are identical to the critic network of *DDPG-RMAVS*. We also performed hyperparameter tuning for both algorithms and selected the optimal values for each of them.

In Figure 10a, we present the cumulative training rewards over 2000 episodes for various algorithms. The *DDPG-RMAVS* algorithm demonstrates the most promising performance with an average training reward of -9000 per episode, surpassing its closest competitor, *RVS*, by 11.1%. In contrast, *GVRS* performs notably poorly, with the lowest average reward of approximately -17000 per episode, 88.9% lower than that of *DDPG-RMAVS*. This suboptimal performance in *GVRS* is at-

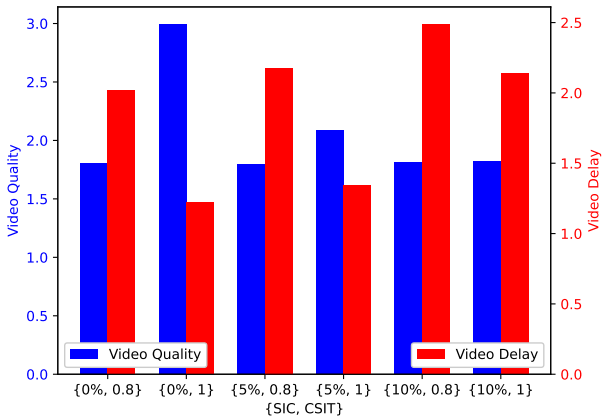


Fig. 9: Video delay and video quality with different imperfect SIC-CSIT factors.

TABLE III: Comparison on runtime complexity (in s) of *DDPG-RMAVS* vs. different algorithms

Algorithm	K=2	K=5	K=10	K=20	K=50
GVRS	1.72	−−	−−	−−	−−
DQN-RV	6.09	6.84	7.83	10.38	16.67
A2C-RV	6.79	7.4	8.23	10.41	17.32
RVS	9.81	10.58	11.17	13.89	20.47
DDPG-RMAVS	7.55	8.64	9.66	12.37	19.12

TABLE IV: Comparison on stalling events and buffer capacity of user-1 on different algorithms

Algorithm	Stalling event	Buffer capacity
GVRS	482	519.38
DQN-RV	89	1994.14
A2C-RV	45	2953.21
RVS	15	2672.61
DDPG-RMAVS	9	4218.53

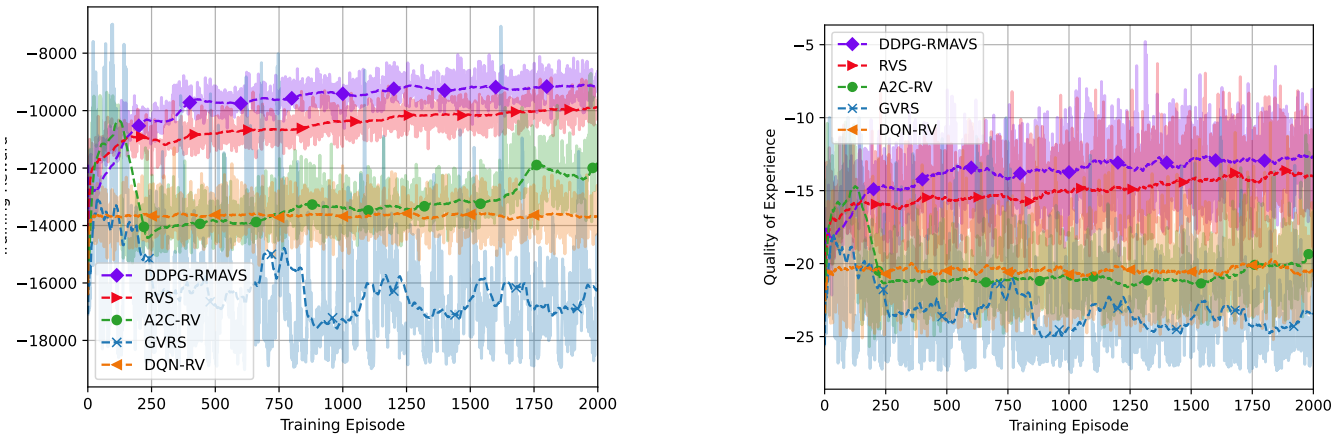
tributed to its focus on maximizing rewards without adequately considering penalties, resulting in negative rewards. The *DQN-RV* algorithm maintains a relatively stable training reward of approximately -14000 throughout the episodes, making it the second-worst performer. *A2C-RV* exhibits early reward fluctuations, likely due to probabilistic factors, but stabilizes and shows a 14.3% improvement in reward from -14000 to -12000 between the 250th and 2000th episodes.

In alignment with the findings on training rewards, we further delve into the average QoE across all users. Fig.10b illustrates the total QoE for all users at the final timestep (i.e., measured at the 700th timestep) across the 2000 episodes. The proposed *DDPG-RMAVS* algorithm yields the highest QoE score, averaging at about -13.5. Given that the total number of users is $K = 10$, the average QoE per user using *DDPG-RMAVS* comes to -1.35. Assuming the quality variation component in (17) is negligible (e.g., the quality of chunks is at its maximum and can no longer be increased), the download time for a 3-second timeslot would equate to 1.35 seconds. Comparatively, the average QoE scores for *RVS*, *A2C-RV*, *DQN-RV*, and *GVRS* are -15, -21, -20.5, and -23.5, respectively. This indicates that *DDPG-RMAVS* offers a 10% higher QoE than *RVS*, around 37% increase over *A2C-RV* and *DQN-RV*, and a 42.6% improvement compared to *GVRS*. Overall, it is evident that the QoE is reflective of the training reward across all algorithms, showcasing the relevance and efficacy of the training reward in predicting user experience outcomes.

In Figure 11a, we examine the Q-loss of various actor-critic algorithms, with exceptions for *GVRS* and *DQN-RV*. Notably, *GVRS* lacks a loss function, and *DQN-RV* exhibits significantly higher Q-loss values due to its unique calculation method. The *A2C-RV* algorithm initially maintains a Q-loss of approximately 1.5, a 60% reduction compared to *DDPG-RMAVS*, which initially fluctuates but later stabilizes at an average of about 4. However, after the 1750th episode, the Q-loss in *A2C-RV* sharply rises to nearly 6, coinciding with the period of rising rewards as seen in Figure 10a. In contrast, the *RVS* algorithm experiences a peak Q-loss of 6 around the 600th episode, which is 50% higher than the eventual stabilized Q-loss of *DDPG-RMAVS*. The peak Q-loss in *RVS* rapidly decreases over the next 400 episodes, ultimately stabilizing at around 4, matching the stable Q-loss of *DDPG-RMAVS*. In summary, due to its consistent maintenance of a stable Q-Loss, the *DDPG-RMAVS* algorithm emerges as the most preferable, with a 33.3% lower Q-loss compared to the peak values observed for *A2C-RV* and *RVS*.

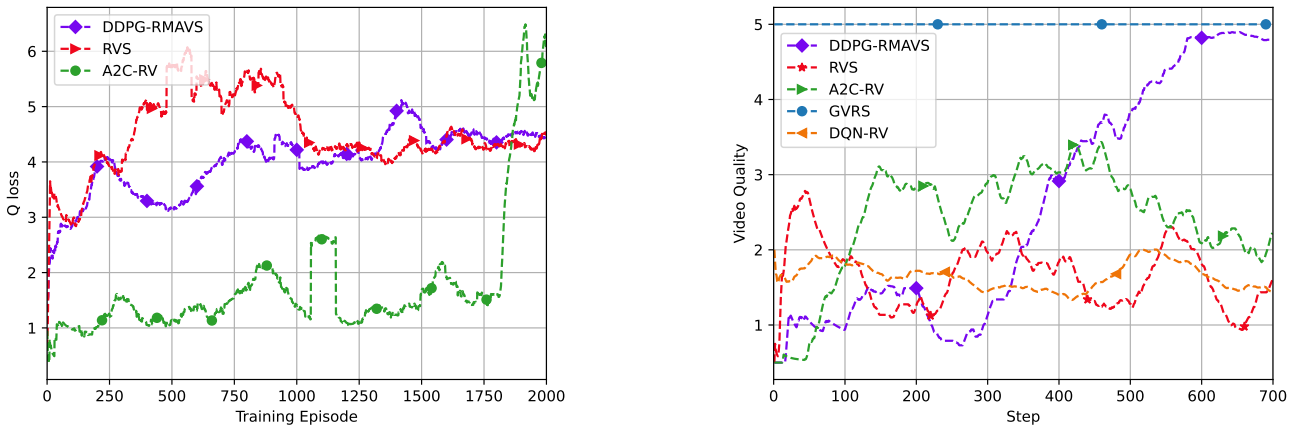
In Figure 11b, we illustrate changes in video bitrate during a streaming session. Each session starts at the lowest bitrate (360p) and aims to maximize it for improved user experience. *GVRS* consistently prioritizes the highest bitrate, risking buffering and penalties. On the other hand, the proposed *DDPG-RMAVS* algorithm starts at the lowest quality, fluctuating between 0.5 and 1.5 in the initial 250 timesteps but subsequently surges to nearly 5, an approximate 900% increase. *A2C-RV* achieves a peak score of 3.5 (1440p) around the 430th timestep but settles at 2 by the session's end. *RVS* exhibits significant bitrate fluctuations, resulting in the lowest quality score among the algorithms. Lastly, *DQN-RV* maintains stable video quality, with a score mostly around 1.8.

Table III demonstrates the runtime of *DDPG-RMAVS* compares with other algorithms on different number of users K. Each column contains the training time on 700 steps on one episode in seconds. The *GVRS* has the lowest runtime at 1.72 seconds, representing the lowest complexity algorithm. This is true because *GVRS* has a finite and discrete action to choose from. Thanks to this, despite the increasing number of users, *GVRS* runtime barely change. For the other four algorithms, as K increases, the runtime progress due to space complexity increment. Among them, algorithm *DQN-RV* has the lowest complexity due to one actor-critic network instead of main-target actor-critic, hence results in second-best lowest runtime. For *A2C-RV*, the runtime is relatively 1 second lower than *DDPG-RMAVS* and 2 seconds than *RVS*. Meanwhile, *RVS* although having similar complexity as with *DDPG-RMAVS*, the entropy term is the key factor for higher computational complexity.



(a) Rewards (b) Quality of Experience

Fig. 10: Different model comparison on average training rewards and QoE.



(a) Q Loss (b) Video Quality

Fig. 11: Different model comparison on Q Loss and perceived video quality on one user.

We conduct an in-depth analysis of how *DDPG-RMAVS* impacts user experience by examining buffer capacity and re-buffering events throughout the streaming session, as detailed in Table IV. Notably, *DDPG-RMAVS* stands out with the largest buffer size at 4218.53 seconds and the lowest incidence of stalling events, with just 9 instances, the fewest among the algorithms considered. This represents a 42.88% increase in buffer size compared to the next-best algorithm in buffer capacity, *A2C-RV*, and a 40% reduction in stalling events compared to *RVS*, which excels in mitigating stalling. *RVS*, *A2C-RV*, and *DQN-RV* end with final buffer sizes of 2672.61 seconds, 2953.21 seconds, and 1994.14 seconds, respectively. Despite having a smaller final buffer size, *RVS* experiences fewer stalling events than *A2C-RV*, enhancing its overall stability and appeal. Conversely, *GVRS* exhibits the smallest buffer size at 519.38 seconds and the highest incidence of stalling events, averaging 482 occurrences over 700 timesteps. This behavior can be attributed to *GVRS*'s aggressive bitrate selection approach, which prioritizes maximizing (17) without considering buffer penalties, resulting in frequent re-buffering events and reduced user experience.

VI. Limitations and Future Works

In this section, we delve into certain limitations not addressed in this research. These may arise from assumptions made during the simulation execution or inherent disadvantages within the model itself. By shedding light on these limitations, we aim to identify and suggest promising directions for future research.

A. Limitations

- **Ratio of common message decoding rate:** In section III-A, we have mentioned that the decoding rate for the common message of all users R_c is the sum of rates for decoding the common message of user k C_k. In practice, the ratio for the rate C_k of each user contributing in R_c

varies. This can depend on the interference, the distance of the user to BS, or other factors that affect the SINR of each user. In this research, we assumed that the portion of C_k for each user is equal for every user. Henceforth, the calculation of the total transmission rate R_k for each user can be affected by this.

- **Impact of linear precoding:** The impact of linear precoder $\mathbf{z}$ has not been clarified in this research. The most important usage of linear precoder is its beamforming capability, which helps users to accurately decode its intended message, increasing interference management and enhancing robustness [59], [60]. Nevertheless, the implementation of a linear precoder subsequently increases the complexity of the system model and, hence was not considered for the simulation in the scope of this study.

B. Future Works

We propose several potential avenues for future exploration in light of the aforementioned challenges. First, existing research, such as Yang *et al.* [61] and Hieu *et al.* [23], which explored common message rate allocation could serve as a reference for future enhancements. Further investigations could be conducted on the precoding problem, particularly its beamforming capacity, to understand its impact on video streaming parameters like latency and stability. Furthermore, analyzing the implementation for high mobility users could provide a clearer perspective on the effects of unpredictable, imperfect CSIT. Another worthy pursuit is the development of an AI prediction model to anticipate the magnitude of $\mathbf{e}_k$ and fluctuations in buffer capacity.

VII. Conclusion

This study delves into the QoE optimization challenge, focusing on video quality and latency within a multi-user downlink RSMA-based IoMT streaming system under imperfect CSIT and SIC conditions. Through an analysis of the communication channel and IoMT streaming model, we underscored factors directly influencing performance. Following this, we formulated an optimization problem aimed at maximizing user QoE through joint optimization of video bitrate selection and power allocation. We converted this problem into an MDP framework, leading to the proposal of a DRL strategy, the *DDPG-RMAVS*, which is grounded in the DDPG algorithm. Simulation outcomes displayed a significant performance superiority of our proposal over SAC-based, A2C-based, DQN-based, and Greedy-based counterparts, as well as a remarkable convergence. In particular, *DDPG-RMAVS* displayed an 11.1% enhancement in training rewards relative to *RVS*, a buffer size 42.88% greater than that of *A2C-RV*, a 33.3% reduction in Q-Loss when compared with peak values registered by *RVS*, 54.84% higher video quality compare with peak values of *A2C-RV*, and 85% less stalling events than *DQN-RV*. Despite these achievements, further research opportunities linger, including exploring aspects such as the ratio for common message decoding rate, the impact of linear precoding, and the fluctuations in CSIT changes. These areas are poised for future exploration.

References

[1] Ericsson, "Mobile data traffic outlook," https://www.ericsson.com/en/reports-and-papers/mobility-report/dataforecasts/mobile-traffic-forecast, 2022, (Accessed on June 28, 2023).

[2] 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*, 2022.

[3] B. Clerckx, Y. Mao, E. A. Jorswieck, J. Yuan, D. J. Love, E. Erkip, and D. Niyato, "A primer on rate-splitting multiple access: Tutorial, myths, and frequently asked questions," *IEEE Journal on Selected Areas in Communications*, 2023.

[4] Z. Yang, M. Chen, W. Saad, W. Xu, and M. Shikh-Bahaei, "Sum-rate maximization of uplink rate splitting multiple access (RSMA) communication," *IEEE Transactions on Mobile Computing*, vol. 21, no. 7, pp. 2596–2609, 2020.

[5] L. Li, K. Chai, J. Li, and X. Li, "Resource allocation for multicarrier rate-splitting multiple access system," *IEEE Access*, vol. 8, pp. 174 222–174 232, 2020.

[6] H. Fu, S. Feng, and D. W. K. Ng, "Resource allocation design for irs-aided downlink MU-MISO RSMA systems," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2021, pp. 1–6.

[7] M. Z. Hassan, G. Kaddoum, and O. Akhrif, "Interference management in cellular-connected internet of drones networks with drone-pairing and uplink rate-splitting multiple access," *IEEE Internet of Things Journal*, vol. 9, no. 17, pp. 16 060–16 079, 2022.

[8] S. Naser, L. Bariah, S. Muhaidat, M. Al-Qutayri, M. Uysal, and P. C. Sofotasios, "Interference management strategies for multiuser multicell MIMO VLC systems," *IEEE Transactions on Communications*, vol. 70, no. 9, pp. 6002–6019, 2022.

[9] Y. Xu, Y. Mao, O. Dizdar, and B. Clerckx, "Max-min fairness of rate-splitting multiple access with finite blocklength communications," *IEEE Transactions on Vehicular Technology*, 2022.

[10] B. Lee and W. Shin, "Max-min fairness precoder design for rate-splitting multiple access: Impact of imperfect channel knowledge," *IEEE Transactions on Vehicular Technology*, 2022.

[11] Z. Yang, J. Shi, Z. Li, M. Chen, W. Xu, and M. Shikh-Bahaei, "Energy efficient rate splitting multiple access (RSMA) with reconfigurable intelligent surface," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2020, pp. 1–6.

[12] Y. Mao, B. Clerckx, and V. O. Li, "Energy efficiency of rate-splitting multiple access, and performance benefits over SDMA and NOMA," in *2018 15th International Symposium on Wireless Communication Systems (ISWCS)*. IEEE, 2018, pp. 1–5.

[13] C. Hao, B. Rassouli, and B. Clerckx, "Achievable dof regions of mimo networks with imperfect csit," *IEEE Transactions on Information Theory*, vol. 63, no. 10, pp. 6587–6606, 2017.

[14] E. Piovano and B. Clerckx, "Optimal dof region of the k-user miso bc with partial csit," *IEEE Communications Letters*, vol. 21, no. 11, pp. 2368–2371, 2017.

[15] A. G. Davoodi and S. Jafar, "Degrees of freedom region of the (m, n1, n2) mimo broadcast channel with partial csit: An application of sum-set inequalities based on aligned image sets," *IEEE Transactions on Information Theory*, vol. 66, no. 10, pp. 6256–6279, 2020.

[16] B. Clerckx, Y. Mao, R. Schober, E. A. Jorswieck, D. J. Love, J. Yuan, L. Hanzo, G. Y. Li, E. G. Larsson, and G. Caire, "Is noma efficient in multi-antenna networks? a critical look at next generation multiple access techniques," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 1310–1343, 2021.

[17] A. Schröder, M. Röper, D. Wuebben, B. Matthiesen, P. Popovski, and A. Dekorsy, "A comparison between rsma, sdma, and oma in multi-beam leo satellite systems," in *WSA & SCC 2023; 26th International ITG Workshop on Smart Antennas and 13th Conference on Systems, Communications, and Coding*. VDE, 2023, pp. 1–6.

[18] 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, 2018.

[19] H. Joudeh and B. Clerckx, "Sum-rate maximization for linearly precoded downlink multiuser miso systems with partial csit: A rate-splitting approach," *IEEE Transactions on Communications*, vol. 64, no. 11, pp. 4847–4861, 2016.

[20] Y. Mao and B. Clerckx, "Beyond dirty paper coding for multi-antenna broadcast channel with partial csit: A rate-splitting approach," *IEEE Transactions on Communications*, vol. 68, no. 11, pp. 6775–6791, 2020.

[21] O. Dizdar, Y. Mao, W. Han, and B. Clerckx, "Rate-splitting multiple access for downlink multi-antenna communications: Physical layer design and link-level simulations," in *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, 2020, pp. 1–6.

[22] O. Dizdar, Y. Mao, and B. Clerckx, "Rate-splitting multiple access to mitigate the curse of mobility in (massive) mimo networks," *IEEE Transactions on Communications*, vol. 69, no. 10, pp. 6765–6780, 2021.

[23] N. Q. Hieu, D. T. Hoang, D. Niyato, and D. I. Kim, "Optimal power allocation for rate splitting communications with deep reinforcement learning," *IEEE Wireless Communications Letters*, vol. 10, no. 12, pp. 2820–2823, 2021.

[24] H. T. H. Giang, P. D. Thanh, H. Ko, and S. Pack, "Deep reinforcement learning-based power allocation for downlink RSMA system," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2022, pp. 775–777.

[25] R. Zhang, K. Xiong, Y. Lu, P. Fan, D. W. K. Ng, and K. B. Letaief, "Energy efficiency maximization in ris-assisted swipt networks with rsma: A ppo-based approach," *IEEE Journal on Selected Areas in Communications*, 2023.

[26] D.-T. Hua, Q. T. Do, N.-N. Dao, and S. Cho, "On sum-rate maximization in downlink UAV-aided RSMA systems," *ICT Express*, 2023.

[27] D. T. Hua, Q. T. Do, T. V. Nguyen, C. M. Ho, and S. Cho, "Trajectory design in multi-UAV-assisted RSMA downlink communication," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2022, pp. 1048–1050.

[28] T. P. Truong, N.-N. Dao, and S. Cho, "HAMEC-RSMA: Enhanced aerial computing systems with rate splitting multiple access," *IEEE Access*, vol. 10, pp. 52 398–52 409, 2022.

[29] J. Ji, L. Cai, K. Zhu, and D. Niyato, "Decoupled association with rate splitting multiple access in UAV-assisted cellular networks using multi-agent deep reinforcement learning," *IEEE Transactions on Mobile Computing*, 2023.

[30] T.-V. Nguyen, N. P. Nguyen, C. Kim, and N.-N. Dao, "Intelligent aerial video streaming: Achievements and challenges," *Journal of Network and Computer Applications*, vol. 211, p. 103564, 2023.

[31] T. Huang, R.-X. Zhang, C. Zhou, and L. Sun, "QARC: Video quality aware rate control for real-time video streaming based on deep reinforcement learning," in *Proceedings of the 26th ACM international conference on Multimedia*, 2018, pp. 1208–1216.

[32] R. Hong, Q. Shen, L. Zhang, and J. Wang, "Continuous bitrate & latency control with deep reinforcement learning for live video streaming," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 2637–2641.

[33] Y. Zhang, P. Zhao, K. Bian, Y. Liu, L. Song, and X. Li, "DRL360: 360-degree video streaming with deep reinforcement learning," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 1252–1260.

[34] H. Pang, C. Zhang, F. Wang, J. Liu, and L. Sun, "Towards low latency multi-viewpoint 360 interactive video: A multimodal deep reinforcement learning approach," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 991–999.

[35] N. Q. Hieu, D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz, "When virtual reality meets rate splitting multiple access: A joint communication and computation approach," *IEEE Journal on Selected Areas in Communications*, 2023.

[36] ETSI, "5G; System architecture for the 5G System (5GS) (3GPP TS 23.501 version 16.6.0 Release 16) ," https://www.etsi.org/deliver/etsi_ts/123500_123599/123501/16.06.00_60/ts_123501v160600p.pdf, 2020, (Accessed on December 29, 2022).

[37] A. Mishra, Y. Mao, O. Dizdar, and B. Clerckx, "Rate-splitting multiple access for 6G–part I: Principles, applications and future works," *arXiv preprint arXiv:2205.02548*, 2022.

[38] ——, "Rate-splitting multiple access for downlink multiuser mimo: Precoder optimization and phy-layer design," *IEEE Transactions on Communications*, vol. 70, no. 2, pp. 874–890, 2021.

[39] Z. W. Si, L. Yin, and B. Clerckx, "Rate-splitting multiple access for multigateway multibeam satellite systems with feeder link interference," *IEEE Transactions on Communications*, vol. 70, no. 3, pp. 2147–2162, 2022.

[40] Y. Xu, Y. Mao, O. Dizdar, and B. Clerckx, "Rate-splitting multiple access with finite blocklength for short-packet and low-latency downlink communications," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 11, pp. 12 333–12 337, 2022.

[41] S. K. Singh, K. Agrawal, K. Singh, Y.-M. Chen, and C.-P. Li, "Performance analysis and optimization of RSMA enabled UAV-aided IBL and FBL communication with imperfect SIC and CSI," *IEEE Transactions on Wireless Communications*, 2022.

[42] S. K. Singh, K. Agrawal, K. Singh, B. Clerckx, and C.-P. Li, "RSMA for hybrid RIS-UAV-aided full-duplex communications with finite blocklength codes under imperfect SIC," *IEEE Transactions on Wireless Communications*, 2023.

[43] 3GPP, "Universal Mobile Telecommunications System (UMTS); LTE; 5G; Study on improved streaming Quality of Experience (QoE) reporting in 3GPP services and networks," *3GPP TR 26.909*, 2022.

[44] S. Lederer, C. Müller, and C. Timmerer, "Dynamic adaptive streaming over HTTP dataset," in *Proceedings of the 3rd multimedia systems conference*, 2012, pp. 89–94.

[45] T. Huang, C. Zhou, R.-X. Zhang, C. Wu, X. Yao, and L. Sun, "Comyco: Quality-aware adaptive video streaming via imitation learning," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 429–437.

[46] J. Kua, G. Armitage, and P. Branch, "A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1842–1866, 2017.

[47] D. Bertsekas, *Dynamic programming and optimal control: Volume I*. Athena scientific, 2012, vol. 1.

[48] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *International conference on machine learning*. Pmlr, 2014, pp. 387–395.

[49] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[50] Spinning Up OpenAI, "Deep Deterministic Policy Gradient ," https://spinningup.openai.com/en/latest/algorithms/ddpg.html, (Accessed on January 27, 2023).

[51] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[52] T. P. Truong, T.-V. Nguyen, W. Noh, S. Cho, *et al.*, "Partial computation offloading in noma-assisted mobile-edge computing systems using deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 8, no. 17, pp. 13 196–13 208, 2021.

[53] Z. Chu, Z. Zhu, F. Zhou, M. Zhang, and N. Al-Dhahir, "Intelligent reflecting surface assisted wireless powered sensor networks for internet of things," *IEEE Transactions on Communications*, vol. 69, no. 7, pp. 4877–4889, 2021.

[54] Youtube Help, "Choose live encoder settings, bitrates, and resolutions ," https://support.google.com/youtube/answer/2853702?hl=en#zippy=, (Accessed on March 15, 2023).

[55] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[56] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[57] M. Sewak and M. Sewak, "Actor-critic models and the A3C: The asynchronous advantage actor-critic model," *Deep reinforcement learning: frontiers of artificial intelligence*, pp. 141–152, 2019.

[58] R. Li, C. Wang, Z. Zhao, R. Guo, and H. Zhang, "The LSTM-based advantage actor-critic learning for resource management in network slicing with user mobility," *IEEE Communications Letters*, vol. 24, no. 9, pp. 2005–2009, 2020.

[59] S. Ma, H. Zhou, Y. Mao, X. Liu, Y. Wu, B. Clerckx, Y. Wang, and S. Li, "Robust beamforming design for rate splitting multiple access-aided miso visible light communications," *arXiv preprint arXiv:2108.07014*, 2021.

[60] T. Cai, J. Zhang, S. Yan, L. Meng, J. Sun, and N. Al-Dhahir, "Resource allocation for secure rate-splitting multiple access with adaptive beamforming," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2021, pp. 1–6.

[61] Z. Yang, M. Chen, W. Saad, and M. Shikh-Bahaei, "Optimization of rate allocation and power control for rate splitting multiple access (RSMA)," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 5988–6002, 2021.