







Design and Analysis of MEC- and Proactive Caching-Based 360° Mobile VR Video Streaming

Qi Cheng , Hangguan Shan , *Member, IEEE*, Weihua Zhuang , *Fellow, IEEE*, Lu Yu , *Member, IEEE*, Zhaoyang Zhang , *Member, IEEE*, and Tony Q. S. Quek , *Fellow, IEEE*

Abstract—Recently, 360-degree mobile virtual reality video (MVRV) has become increasingly popular because it can provide users with an immersive experience. However, MVRV is usually recorded in a high resolution and is sensitive to latency, which indicates that broadband, ultra-reliable, and low-latency communication is necessary to guarantee the users' quality of experience. In this paper, we propose a mobile edge computing (MEC)-based 360-degree MVRV streaming scheme with field-of-view (FoV) prediction, which jointly considers video coding, proactive caching, computation offloading, and data transmission. To meet the requirement of stringent end-to-end (E2E) latency, the user's viewpoint prediction is utilized to cache video data proactively, and computing tasks are partially offloaded to the MEC server. In addition, we propose an analytical model based on diffusion process to study the packet transmission process of 360-degree MVRV in multihop wired/wireless networks and analyze the performance of the MEC-enabled scheme. The simulation results verify the accuracy of the analysis and the effectiveness of the proposed MVRV streaming scheme in reducing the E2E delay. Furthermore, the analytical framework sheds some light on the impacts of system parameters, e.g., FoV prediction accuracy and transmission rate, on the balance between computation delay and communication delay.

Index Terms—360-degree mobile virtual reality video streaming, computation offloading, end-to-end delay, field-of-view prediction, mobile edge computing, proactive caching.

Manuscript received June 15, 2020; revised October 7, 2020 and February 3, 2021; accepted March 3, 2021. Date of publication March 19, 2021; date of current version March 29, 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1801104, in part by the National Natural Science Foundation Program of China (NSFC) under Grant 61771427, in part by the Zhejiang Provincial Key Project of Research and Development under Grant 2021C01119, in part by the SUTD-ZJU IDEA Grant for Visiting Professor ZJUVPI800104, in part by the SUTD Growth Plan Grant for AI, and in part by the Huawei Technologies Company Ltd. under Grant YBN2018115223. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Abderrahim Benslimane. (Corresponding author: Hangguan Shan.)

Qi Cheng, Hangguan Shan, Lu Yu, and Zhaoyang Zhang are with the Zhejiang Provincial Key Laboratory of Information Processing Communication Networks, the College of Information Science and Electronic Engineering, and SUTD-ZJU IDEA, Zhejiang University, Hangzhou 310027, China (e-mail: 3150104884@zju.edu.cn; hshan@zju.edu.cn; yul@zju.edu.cn; ning_ming@zju.edu.cn).

Weihua Zhuang is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo N2L 3G1, ON, Canada (e-mail: wzhuang@bcr.uwaterloo.ca).

Tony Q. S. Quek is with the Information System Technology and Design Pillar, Singapore University of Technology and Design, 487372, Singapore (e-mail: tonyquek@sutd.edu.sg).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TMM.2021.3067205>.

Digital Object Identifier 10.1109/TMM.2021.3067205

I. INTRODUCTION

MOBILE virtual reality (VR) is expected to become an extremely popular application on 5 G networks and refers to the transmission of video and sound files from a cloud server to a user's terminal device via a multihop network in order to achieve storage and rendering by the cloud server or edge server in virtual reality business. For example, this application is practical with the aid of cloud computing technology and stable gigabit fiber networks [1]. A 360-degree video, which is also known as a three-degree-of-freedom (3-DoF) spherical video, can provide users with an immersive experience. Because 360-degree mobile virtual reality video (MVRV) combines the multiple requirements of high capacity of enhanced mobile broadband (eMBB) services and stringent latency and reliability of ultra-reliable low-latency communication (URLLC) services, there are currently many technical difficulties in supporting this application [2], [3]. Currently, research in this area focuses not only on traditional approaches including increasing the transmission rate or decreasing the bandwidth requirements but also on jointly utilizing the resources of caching, computation, communication (3 C) [4]–[6].

Both increasing the transmission rate from the perspective of efficiently utilizing network resources and decreasing bandwidth requirements from the perspective of source coding are in response to the large quantity of data used in mobile virtual reality. Methods used to improve the transmission rate in the literature mainly focus on three aspects: multiconnectivity technologies, efficient resource allocation, and millimeter-wave wireless communications [7]–[9]. The methods of reducing bandwidth requirements in the literature mainly focus on the application of advanced video coding technologies [8]. From the perspective of video transmission, there are mainly two kinds of technical routes for online transmission of MVRV: full-view transmission and field-of-view (FoV) transmission [8], [10]. The former transmits all video data to the user, while the latter emphasizes a high-quality transmission of the current FoV. Consequently, the full-view transmission has lower latency at the cost of enormous bandwidth requirements, while the FoV transmission can achieve a better result when bandwidth is insufficient.

Considering the large quantity of data in the MVRV service and the limited and dynamic sharing of communication resources, the method of relying only on increasing transmission rates or decreasing transmission bandwidth does not necessarily support the application of mobile virtual reality. For this

reason, proactive caching of VR videos is seen as a key ability that can improve users' quality of experience (QoE). Recent studies have shown that the head movement of a VR game user in the proceeding hundreds of milliseconds can be predicted with high accuracy [11]. Based on the estimation of the user's future head movement, video frames can be preprocessed on a remote cloud server and cached in the network edge or the user's head-mounted device [12]. To facilitate proactive caching, the video sequence is divided into segments of fixed duration in the time domain, and the same sequence can be divided into tiles in the spatial domain, where each tile is made available in a set of different quality levels to adapt to the time-varying bandwidth and changes in viewpoint.

In a mobile VR system, heavy image processing tasks require abundant computational resources, which are usually insufficient in the local head-mounted display (HMD) graphics processing unit (GPU). Therefore, edge computing is also considered to be a key enabling technology for wireless mobile virtual reality applications [4], [13]. For mobile VR, end-to-end (E2E) delay mainly consists of computation and communication delays. Offloading computing tasks to mobile edge computing (MEC) servers significantly relieves the computing burden from users' HMDs at the expense of incurring additional communication delay in both downlink and uplink directions. The uplink communication delay due to offloading computing tasks to the MEC server is typically very small owing to the small quantity of data needed, e.g., the user tracking data and interactive control decisions. How to balance the computational and communication delays by determining the proportion of offloaded computing tasks is the key to the success of these applications when applying edge computing technologies [14].

In view of the difficulty of implementing mobile VR, combining computation, communication, and storage is considered to be the key to support these applications. Although schemes have been proposed to address the problems in 360-degree MVRV delivery (e.g., [5], [6], [15]), no comprehensive design of such a system jointly considers video coding, data storage, and data delivery in both the core network and edge network while utilizing all aforementioned cornerstones. Furthermore, to understand the overall performance of the system, it is indispensable to develop an analytical framework that includes video coding, source data acquisition, data transmission, and computation offloading. In this work, we aim to fill this gap. The main contributions of this paper are summarized as follows:

- We propose a 360-degree MVRV streaming scheme that jointly considers video coding, proactive caching, computation offloading, and data transmission. To solve the stringent E2E delay problem, user viewpoint prediction is used to cache video data, and computation tasks are offloaded to the MEC server;
- We propose an analytical model based on the diffusion process to study the packet transmission process of 360-degree MVRV in multihop wired/wireless networks and analyze the performance of the MEC-enabled scheme. The probability density function (PDF) of the E2E delay and the delay outage probability are derived from the analytical framework. Therefore, a set of appropriate system

parameters can be tuned to maximize the users' QoE, or the performance can be predicted when system parameters are given;

- We conduct extensive simulations to validate the performance of the designed system and compare our MEC-enabled scheme with a dynamic adaptive streaming over HTTP (DASH) scheme. The simulation results verify the accuracy of the analysis and the effectiveness of the proposed MVRV streaming scheme. Furthermore, we reveal the influence of the network transmission rate, FoV prediction accuracy, MEC computation resource allocation strategy, and other system parameters on the E2E delay and delay outage probability.

The remainder of this paper is organized as follows. In Section II, we review related works. In Section III, we propose the MEC-based 360-degree MVRV streaming scheme with FoV prediction. In Section IV, we analyze the performance of the proposed scheme, where the E2E delay and the delay outage probability are considered to be key indicators of the MEC-enabled scheme. In Section V, we discuss the simulation results to evaluate system performance. Finally, we draw conclusions in Section VI.

II. RELATED WORKS

The QoE of an MVRV user is highly dependent on stringent transmission latency and reliability, which are usually difficult to be guaranteed due to the dynamic resource sharing in multihop wired/wireless networks from the remote cloud server to the user. Therefore, to improve users' QoE, it is desirable to make full use of the computing and storage capacity of the cloud server, edge computing server, and user device and to combine user viewpoint tracking technology with video coding technology to design intelligent transmission systems [4], [6], [9]. The authors in [7] propose a multipath cooperative routing scheme to facilitate the transmission of 360-degree MVRV among edge data centers, base stations, and users in the absence of computation delay of VR data. Sukhmani *et al.* [16] propose improving QoS by coupling edge caching and video coding. A distributed algorithm based on machine learning is proposed in [13] for resource management in 360-degree MVRV streaming using a simplified E2E transmission process.

Proposed in [5] is a novel MEC-based mobile VR delivery framework that formulates the joint caching and computing optimization problem to minimize the average required transmission rate. This study also reports interesting communications-caching-computing tradeoffs when FOVs are homogeneous. In [6], the joint 3 C resource allocation in the 5 G wireless networks is studied, while the role of cloud server in the mobile VR business is not considered.

Specifically, many studies have investigated how to balance communication latency and computation latency, and scheduling offloading tasks with MEC technology in VR video delivery. The authors in [15] propose a mobile edge computing system that reduces communication-resource consumption by fully exploiting the computation and caching resources of the mobile VR device. A computation offloading scheduling scheme

that can be used for MVRV is presented under the constraint of average latency with MPEG media transport (MMT) video modularized technology [17], [18]. Elbamy *et al.* [9] study the impacts of computation offloading, millimeter-wave communication, and proactive caching on latency performance in the VR interactive gaming scene. The authors in [19] analyze and verify that the scalable high efficiency video coding (SHVC) technology can improve the performance of the VR system when high-performance servers can provide multiconnectivity service for VR users.

From the perspective of video streaming control, various techniques in the literature address the problem of time-varying bandwidth adaptation and viewpoint adaptation, including the buffer-based rate adaptation [20] and the tile-based DASH [21]. Take DASH as an example. Its basic idea is to allocate different bit rates to the tiles of the 360-degree video with advanced video coding technologies, such as scalable video coding (SVC), and allow the client to request a specific quality of the video tiles to adapt video streaming to current network conditions, thus achieving a smooth playback without staling and with a better QoE. To enable a DASH client to retrieve a region of interest (RoI) in a high-resolution video while others remain at low resolution, MPEG-DASH uses spatial representation descriptions (SRDs) as features to provide spatial information in its own media presentation description (MPD) file [3], [22]. Furthermore, a few works study the relationship between network dynamics and user's quality of experience. For example, Luan *et al.* [23] model the playback buffer at the receiver by a $G/G/1/\infty$ and $G/G/1/N$ queue, respectively, and develop an analytical framework to investigate the impact of network dynamics on user-perceived video quality. However, this general analytical model is used for traditional video playback, and the characteristics of an ultralow latency service using MVRV are yet to be addressed, which motivates our work. To enable 360-degree video to stream from a server to the user and achieve an ideal experience of MVRV playback, fine-grained decisions must be made based on channel state information. In addition, as a result of the large action space and state space, traditional optimization theory cannot effectively address the problem. Recently, deep reinforcement learning has been used to solve the problem of tile selection [24], [25]. However, the E2E delay and QoE models in these works are oversimplified, and further efforts are necessary to explore the relationship between the decision and E2E delay or user's QoE.

To proactively cache VR video data in the network edge while a user is watching the video, the user's viewpoint needs to be predicted periodically with high accuracy. FoV prediction uses the saliency map method, motion feature detection method, and orientation extraction method, among which the saliency map and orientation extraction are primarily used. There are two ways to produce saliency map: one is to generate the saliency map by traditional digital image processing methods based on the attributes of images, e.g., color, intensity, and semantic information [26]; the other is to directly extract image features by using deep learning. For the second category, the works in [27], [28] use convolution neural networks (CNNs) to generate saliency maps with more high-level features, such as global scene

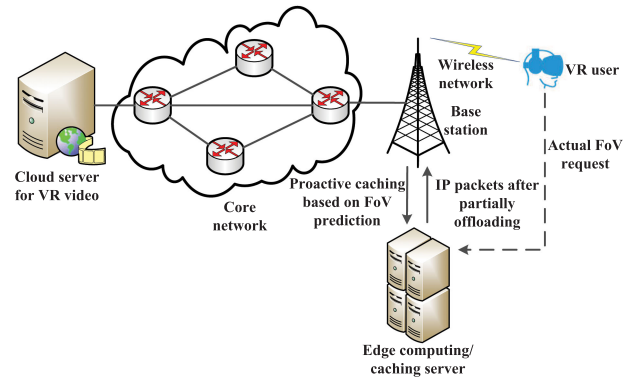


Fig. 1. Proposed 360-degree MVRV streaming system with FoV prediction.

semantic information; the visual geometry group (VGG)-16 network and hyperparameter optimization are used in [29] to find a discriminative image representation; a CNN + long short-term memory (LSTM) + Gaussian mixture model (GMM) multilayer network is trained end-to-end in [30] using the backpropagation method; and the reinforcement learning method in [31] generates a head movement (HM) predicted probability matrix by maximizing the reward of imitating human HM scan-paths through the agent's actions. The motion feature detection method applies a CNN or traditional digital image processing methods to extract motion features [32]. The orientation extraction method calculates predictions based on the sensor data collected by users' head-mounted equipment, including 3D azimuth and acceleration sensor data. The linear regression model can be used to predict the user's viewpoint trace with an accuracy of 92.4% in the coming 0.5 s [33]. Further, the aforementioned three methods can be integrated to obtain the FoV prediction results with higher accuracy utilizing machine learning [34]. In [34], the LSTM network is proposed to learn more long-term dependencies among video frames.

In addition, other studies have focused on improving the accuracy of long-term prediction via the navigation graph approach, which models viewing behaviors in the temporal (segment) and spatial (tile) domains to adapt the rate of the tiled media associated with the view prediction (e.g., [35]).

III. MEC-BASED VIDEO STREAMING DESIGN WITH FOV PREDICTION

Fig. 1 shows the proposed 360-degree MVRV streaming system with FoV prediction via a multihop wired/wireless network.¹ The VR video data are stored, encoded, and compressed in advance by the cloud server. The data will then be cached in the MEC server based on the viewpoint prediction result and transmitted to the VR user based on the corresponding request. The packets will experience different delays in the multihop wired/wireless network, and the task of packet decoding and video rendering can be completed in the MEC server

¹In this study, we only study a single VR user scenario, which can easily be extended into a multi-VR user scenario as long as each user is allocated exclusive 3 C resources along the transmission path.

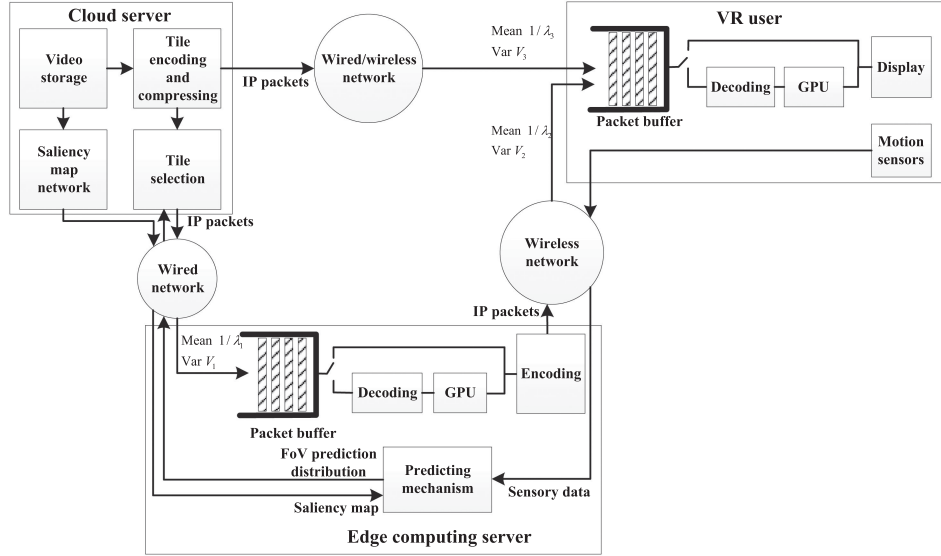


Fig. 2. Functions of network elements in the proposed 360-degree MVRV streaming design with FoV prediction.

TABLE I
KEY NOTATIONS

Notation	Definition
B	Data size per tile per frame
c	Offloading ratio at the MEC server
C_r	Compression ratio of video chunk
D_1	Video chunk duration
$D_{M,l}$	E2E delay in period l of the MEC-enabled scheme
$D_{N,l}$	E2E delay in period l of the MEC-disabled scheme
D_{th}	Preset E2E delay threshold
f	Number of frames per second of a video chunk
h	Data size changing ratio after computation offloading at the MEC server
k	Number of tiles in the vertical or horizontal direction in the FoV
(M, N)	Number of tiles in the vertical and horizontal directions in a projected video plane
P_{out}	Delay outage probability
$R_1(t)$	Transmission rate from the cloud server to the MEC server at time t
$R_2(t)$	Transmission rate from the MEC server to the user at time t
$R_3(t)$	Transmission rate from the cloud server to the user at time t
s	UDP packet size
W_M	Data size processed per second by the MEC server
W_U	Data size processed per second by the user

or the HMD of the VR user. We now describe the system architecture including functions of each component and network model in the first subsection. In the remaining subsections, we discuss in sequence the periodic operating process and delay model, the video coding and FoV prediction model, the MEC caching model, and the computation offloading decision. The key notations used in this paper are listed in Table I.

A. System Architecture and Network Model

Fig. 2 shows the functions of each network element in the proposed MEC-based MVRV streaming design with FoV prediction. The system consists of three key components: the cloud server, the MEC server, and the VR user. The functions of each component are described as follows:

- The cloud server divides a video into $N \times M$ tiles and compresses video tiles for a group of pictures (GOPs). Then, the cloud server transmits data packets to the MEC server for proactive caching based on the FoV prediction results fed back by the MEC server. The MEC server can then produce a saliency map by the CNN + LSTM + GMM network because all video data have been stored in advance.
- The MEC server predicts the user's viewpoint probability distribution based on the saliency map from the cloud server and the sensory data from the user, i.e., the historical FoV requests. The MEC server also caches data packets that come from the cloud server. Moreover, to reduce the burden on the user's HMD and reduce computation delay, the MEC server makes a decision to decode and render a portion of video data on its GPU. Finally, the data packets that have been cached or computed in the MEC server are encoded, compressed, and sent to the VR user.
- The user's HMD sends video requests to the MEC server and cloud server based on the user's actual FoV. The HMD also caches and decodes data packets that come from the MEC server and cloud server and can render video data on the local GPU. Finally, data packets are combined into video frames and sent to the display of the HMD for playback. At the same time, motion sensors collect the user's motion information that will be sent to the MEC server for FoV prediction.

Fig. 2 also shows the transmission process of 360-degree MVRV in a multihop wired/wireless network, which uses the UDP/IP protocol suite.² Let s denote the size of each video data packet in bits. When the MEC server is caching video data proactively, video packets are transmitted from the cloud server to the

²TCP can also be used to support 360 VR, such as DSAH in [3], [21]. However, due to the overhead issue and difficulty in accurately estimating the E2E transmission rate, we consider UDP-based transmission for the proposed MVRV streaming, such as quick UDP Internet connection (QUIC) in [36], [37].

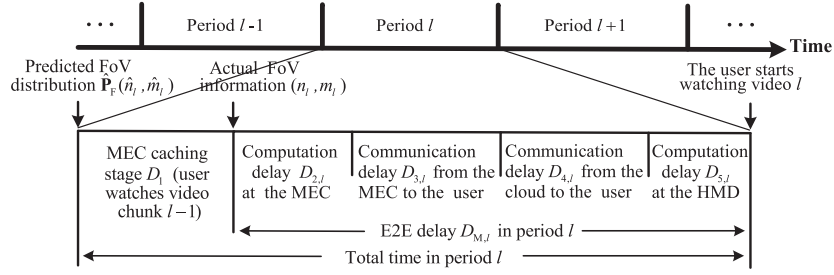


Fig. 3. The periodic working process and delay model of the MEC-enabled scheme.

MEC server over a variable bit rate (VBR) multihop wired network, and the transmission rate at time t is represented by $R_1(t)$. Similarly, when the video packets are transmitted from the MEC server to the user over a VBR single-hop wireless network, the transmission rate at time t is denoted as $R_2(t)$. In this study, where the proposed analytical framework mainly focuses on the impact of multihop networks on the E2E delay, similar to [20], [25], we use random processes that follow certain distributions to represent the transmission rates of both the wireless and wired networks, assuming that the selected transmission rates can be supported by the appropriately designed physical and higher layers. Last, when the video packets are transmitted from the cloud server to the user over a VBR multihop wired/wireless network, we similarly denote the transmission rate at time t as $R_3(t)$. Because the transmission link from the cloud server to the user can be seen as a cascade of the transmission link from the cloud server to the MEC server and that from the MEC server to the user, we have $R_3(t) = \min\{R_1(t), R_2(t)\}$. Due to network dynamics, packets arrive at the MEC server or the user with variable delays. Without loss of generality, we assume that the inter-arrival time of video packets in each of the aforementioned three transmission paths follows a given but arbitrary distribution with mean $\frac{1}{\lambda_i}$ and variance V_i , where $i \in \{1, 2, 3\}$ is the index of the three transmission paths. Then, λ_i and V_i are given by:

$$\frac{1}{\lambda_i} = E\left(\frac{s}{R_i(t)}\right) \quad (1a)$$

$$V_i = \text{Var}\left(\frac{s}{R_i(t)}\right). \quad (1b)$$

B. Periodic Working Process and Delay Model

Fig. 3 shows the periodic working process and delay model of the MEC-enabled scheme. Because the FoV prediction mechanism can only predict the user's viewpoint with high accuracy in a short upcoming period of time, the video should be divided into segments for prediction, which are usually called video chunks. The duration of each video chunk is denoted by D_1 . For the convenience of notation, we use l to denote the index of the video chunk played in period $l + 1$ with a constant duration of D_1 . The system E2E delay denoted by $D_{M,l}$ is the period from the moment that the user sends an FoV request to the MEC server to the moment that the user starts to watch the video chunk.

By caching video chunks in the MEC server, one portion of the corresponding computing tasks for decoding and rendering

can be offloaded to the MEC server, and then, the results can be transmitted to the user, while the remaining portion can be transmitted directly to the user and processed in the HMD. If the MEC server can compute while transmitting the separated tasks, system performance can be improved. However, although it is practical for implementation, this method will find it difficult to obtain the optimal task partition when considering both the time-varying wired/wireless transmission rates in the networks and the impact of queueing networks composed of the cloud server, the MEC server, and the user's HMD. To simplify analysis yet obtain insights analytically, we consider that the system performs computation tasks and communication tasks in a sequential manner. Hence, the result (e.g., the E2E delay performance) derived here represents an upper-bound performance of the system performing both tasks in parallel. Since our MEC-based MVRV streaming system works periodically, we discuss the five stages of the proposed design in detail for a typical period (e.g., period l) as follows:

- 1) Caching stage of the MEC server: The MEC server caches the data of video chunk l based on the predicted FoV result $\hat{\mathbf{P}}_F(\hat{n}_l, \hat{m}_l) = [\hat{P}_F(\hat{n}_l, \hat{m}_l)]_{N \times M}$, while the user is watching video chunk $l - 1$, where $\hat{P}_F(\hat{n}_l, \hat{m}_l)$ represents the probability that the user's viewpoint will fall in tile (\hat{n}_l, \hat{m}_l) at the beginning of the next period, with $\hat{n}_l \in \mathbf{N} = \{1, 2, \dots, N\}$ and $\hat{m}_l \in \mathbf{M} = \{1, 2, \dots, M\}$. We assume that the periodical time of the user watching equals the duration of the video chunk. When a new FoV request is sent to the MEC server, the caching stage is truncated. Thus, the maximum time for the MEC server to cache data packets is D_1 .
- 2) Computation stage of the MEC server: When the MEC server receives a new FoV request (n_l, m_l) from the user with $n_l \in \mathbf{N}$ and $m_l \in \mathbf{M}$, which represents the index of FoV's central tile, it checks whether or not it has relevant data in its cache. To balance the communication delay and computation delay, it decides what percentage of the cached and requested data is to be rendered on its GPU. Let $D_{2,l}$ denote the delay of this computation stage.
- 3) Transmission stage from the MEC server to the user: All requested data packets including both the processed and unprocessed data by the MEC server are delivered to the user, with the delay represented by $D_{3,l}$.
- 4) Transmission stage from the cloud server to the user: The data that the MEC server failed to cache are transmitted from the cloud server to the user directly, with delay represented by $D_{4,l}$.

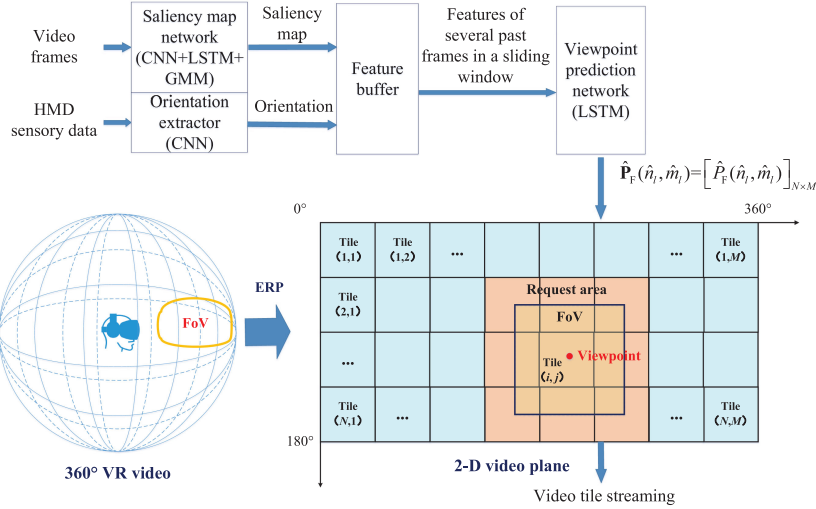


Fig. 4. Video coding and FoV prediction model.

- 5) Computation stage of the user: The HMD's GPU completes the computation task of the data that have not been rendered by the MEC server, with the delay represented by $D_{5,l}$. Then, the HMD sends video frames to the display for playback. At this moment, the tasks of communication and computation in period l are completed, and the user begins watching video chunk l at the beginning of the period $l + 1$.

Consequently, the E2E delay consists of four components: the computation delay of the MEC server $D_{2,l}$, the transmission delay from the MEC server to the user $D_{3,l}$, the transmission delay from the cloud server to the user $D_{4,l}$, and the computation delay of the HMD $D_{5,l}$, i.e.,

$$D_{M,l} = D_{2,l} + D_{3,l} + D_{4,l} + D_{5,l}. \quad (2)$$

Following (2), we do not consider the case that the transmission from the cloud server to the user and that from the MEC server to the user occur in parallel because it complicates delay analysis and also may degrade the delay performance due to, if any, inefficient bandwidth sharing at the wireless link. In addition, (2) does not imply that the E2E delay is independent of duration D_1 of the caching stage because the video compression performance decreases if the video chunk duration D_1 decreases, as encoding and compression take advantage of the temporal correlations of video data. Furthermore, both the accuracy of the FoV prediction and the size of data to be processed in a period are impacted by D_1 .

We study the impact on the E2E delay analytically in the next section.

C. Video Coding and FoV Prediction Model

Fig. 4 illustrates how a 360-degree video chunk, also known as a spherical video chunk, is encoded into tiles and the result of the FoV prediction. Initially, 360-degree MVRV chunk l of frame rate f is projected into a two-dimensional video plane whose angle range is $180^\circ \times 360^\circ$ via equirectangular projection (ERP). Then, it is divided into $N \times M$ tiles, and the data

size of each tile per frame is B bits/tile/frame. As the encoding and compression in the cloud server use temporal and spatial dependencies, the user's HMD or the MEC server requires all video frames of a tile to decode the tile's data packets. Different tiles are encoded and compressed separately, and a full tile must be transmitted even when the FoV area covers only a portion of the tile. Normally, the request area equals $(k + 1)^2$ tiles, which is above the FoV area of $k \times k$ tiles, and the viewpoint might fall in any tile of the video plane.

At the beginning of each period, the MEC server can use the saliency map received from the cloud server and sensory data received from the user to generate the FoV prediction results.

The cloud server can generate the saliency map utilizing the CNN + LSTM + GMM saliency map network proposed in [30]: First, a 3D convolutional network is used to learn the spatiotemporal feature of each frame in a video frame sliding window [38]; Second, the spatiotemporal feature of each frame is input to an LSTM network to capture the long-term dependencies of different GOPs; Finally, because GMM can fit the distribution of an unknown random variable, the output of the LSTM network is utilized to fine tune the parameters of GMM. After training the saliency map network, video frames can be input to the network to obtain the saliency map, i.e., the output of GMM. To obtain the FoV prediction results, the MEC server can also extract orientation data including yaw, pitch, and roll from the sensory data via a CNN [34]. Then, the MEC server uses an LSTM viewpoint prediction network to learn more long-term dependencies among video frames utilizing both the saliency map and orientation information. In each iteration, the LSTM viewpoint prediction network takes features of several past video frames in a video frame feature sliding window as inputs and predicts the viewing probability of tiles with several future video frames in a prediction window as outputs. If only the saliency map or orientation information is available, the FoV can still be predicted [31]. Additionally, [31] provides an efficient variant of FoV prediction network without using the GMM. Since the data size of the saliency map and that of the sensory data are small, the transmission time and network resources they consumed are

neglected in the analysis. It is also noteworthy that because the FoV prediction network can be trained in advance with the raw video data stored in the cloud server and the sensory data from historical users, the training time can generate no impact on the E2E delay.

At the beginning of the caching stage in period l , the prediction mechanism produces a predicted probability matrix $\hat{\mathbf{P}}_F(\hat{n}_l, \hat{m}_l) = [\hat{P}_F(\hat{n}_l, \hat{m}_l)]_{N \times M}$. Because the predicted FoV area may intersect with the border of the video plane, the coordinate set of tiles in the predicted FoV area is defined as $\mathbf{N}_c = \{n - \frac{k}{2}, n - \frac{k}{2} + 1, \dots, n, \dots, n + \frac{k}{2} - 1, n + \frac{k}{2}\} \cap \mathbf{N}$ and $\mathbf{M}_c = \{m - \frac{k}{2}, m - \frac{k}{2} + 1, \dots, m, \dots, m + \frac{k}{2} - 1, m + \frac{k}{2}\} \cap \mathbf{M}$. Then, the request probability matrix $\mathbf{P}_R(n, m) = [P_R(n, m)]_{N \times M}$ is defined, where $P_R(n, m) = \sum_{u \in \mathbf{N}_c} \sum_{v \in \mathbf{M}_c} \hat{P}_F(u, v)$ where $n \in \mathbf{N}$ and $m \in \mathbf{M}$ represents the probability that at least a portion of tile (n, m) will fall in the FoV area at the beginning of the next period. The MEC server sorts all tiles of video chunk l in the descending order of $P_R(n, m)$ and sends the information to the cloud server. Let $\mathbf{A}_{o,l} = [a_{o,l}(n, m)]_{N \times M}$ denote the matrix of which element $a_{o,l}(n, m)$ is the aforementioned order of tile (n, m) with $a_{o,l}(n, m) \in \{1, 2, \dots, NM\}$.

To facilitate the analysis and without loss of generality, we assume that the output, denoted by $\mathbf{P}_F(n_l, m_l) = [P_F(n_l, m_l)]_{N \times M}$, of the prediction mechanism at the beginning of the computational stage of the MEC server in period l is the user's actual FoV distribution at the beginning of the next period (i.e., $l + 1$). The reason is that the viewpoint prediction network trained with the data from the saliency map and user's sensory data (i.e., the historical FoV requests) has a high accuracy in a short upcoming period of time [33].

D. MEC Caching Model

While the VR user is watching video chunk $l - 1$, the MEC server caches video data packets proactively. Although the VR user can also cache video data packets proactively, we let the MEC server cache those data packets instead to overcome the drawback of limited computation capacity at the HMD side; this process fully utilizes the powerful computational capacity of the MEC server. The cloud server encodes and compresses the video data of each tile into packets in advance with a specific compression ratio denoted by C_r . After the cloud server receives a request order matrix $\mathbf{A}_{o,l}$ from the MEC server, it transmits the data packets of different tiles to the MEC server based on the requested order. If the MEC server cannot cache all tiles of video chunk l , the cloud server can transmit the remaining tiles to the user directly after receiving the actual FoV request. Because different tiles are usually encoded and compressed independently while the frames of each tile are processed dependently, we consider that the MEC server will discard packets that do not constitute of all video frames of a tile. Hence, only tiles that are completely received are cached in the MEC server. Let G_l denote the number of tiles cached in the MEC server, within which Q_l tiles are hit (i.e., requested by the user at the end of the caching stage).

E. Computation Offloading Decision

The MEC server and the user's HMD should decode the data packets to obtain the tiles of the projected video and then process the data in the HMD's GPU to reconstruct the spherical video and accommodate the HMD's display size. When analyzing the computation delay, to simplify the analysis, we do not model the computation task of tiles from the perspective of a series of subtasks, including, for example, demodulation, channel decoding, source decoding, and image rendering in FoV in detail. Instead, similar to [39], [40] the computation delay is assumed to be proportional to the size of the raw video data to be processed.³

Because the MEC server has more computing power than the HMD, offloading the computation tasks to the MEC server can reduce the computation delay and the energy consumption of the HMD. However, the size of the data after processing in the MEC server may increase because the specification of the HMD may vary, perhaps due to the resolution of the display, video projection methods, and quantization bit number [3], [19]. Therefore, the transmission delay from the MEC server to the HMD may also increase, thus exceeding the gain from the computation offloading. Hence, to balance the communication delay and computation delay, the MEC server makes a decision about what percentage of the cached and requested data is to be rendered on its GPU. When a new FoV request arrives, the MEC server should first check its cache buffer to determine how much cached video data are requested by the user. The quantity of these data is denoted as $\Gamma_{1,l}$. Then, the MEC server decides to divide the video data into two tasks with an arbitrary ratio c : one with data size of $\Gamma_{1,l}c$ is computed in the MEC's GPU, and the other with data size of $\Gamma_{1,l}(1 - c)$ is computed in the HMD's GPU. Let the computation capability of the MEC server's GPU and that of the HMD's GPU be W_M bit/s and W_U bit/s, respectively. After the computation of the offloaded task, the data size of this part will change, and let h denote the data size ratio between computational output and input; thus, the data size of computation output becomes $\Gamma_{1,l} \cdot c \cdot h$.

IV. PERFORMANCE ANALYSIS

In this section, we apply the diffusion approximation to derive the closed-form expressions of system performance in any period l , i.e., the conditional PDF or probability mass function (PMF) of the delay at each stage and that of the E2E delay, and the delay outage probability. With the obtained results, we can evaluate the impacts of diverse system parameters on the system performance.

A. Number of Proactively Cached Packets in the MEC Server

To evaluate the E2E delay and delay outage probability, we first study the number of proactively cached packets in the MEC server in any period l and apply the diffusion approximation [41], [42] for compact solutions [43]. We denote the number of packets stored by proactive caching in the MEC server at time instant

³If the relationship between each subtask's computation delay and the size of its input data is available, the proposed computation model can be directly extended to consider each subtask individually.

t_1 from the beginning of the period by $Z_1(t_1)$. The diffusion approximation method consists of replacing the discrete number of packets $Z_1(t_1)$ with a continuous process $X_1(t_1)$ [23] and modeling it as Brownian motion:

$$dX_1(t_1) = X_1(t_1 + dt_1) - X_1(t_1) = \alpha_1 dt_1 + O\sqrt{\beta_1 dt_1} \quad (3)$$

where $O \sim \mathcal{N}(0, 1)$ is a normally distributed random variable with zero mean and unit variance, and α_1 and β_1 are called drift and diffusion coefficients, respectively, defined by

$$\alpha_1 = E \left(\lim_{\Delta t_1 \rightarrow 0} \frac{X_1(t_1)}{\Delta t_1} \right) = \lambda_1, \quad (4a)$$

$$\beta_1 = \text{Var} \left(\lim_{\Delta t_1 \rightarrow 0} \frac{X_1(t_1)}{\Delta t_1} \right) = \lambda_1^3 V_1. \quad (4b)$$

We define a conditional PDF $p(x_1, t_1 | x_0)$ of $X_1(t_1)$ at time t_1 as

$$p(x_1, t_1 | x_0) = P(x_1 \leq X_1(t_1) < x_1 + dx_1 | X_1(0) = x_0) \quad (5)$$

where x_0 is the initial queue length. With the diffusion approximation, $p(x_1, t_1 | x_0)$ can be characterized by the diffusion equation [23]:

$$\frac{\partial p(x_1, t_1 | x_0)}{\partial t_1} = \frac{\beta_1}{2} \frac{\partial^2 p(x_1, t_1 | x_0)}{\partial x_1^2} - \alpha_1 \frac{\partial p(x_1, t_1 | x_0)}{\partial x_1} \quad (6)$$

coupled with the initial condition:

$$p(x_1, 0 | 0) = \delta(x_1) \quad (7)$$

and boundary condition:

$$p(b_1, t_1 | 0) = 0. \quad (8)$$

Equation (8) is obtained because the maximum number of data packets for all MN tiles that can be cached is $b_1 = D_1 f \lceil \frac{BC_r}{s} \rceil MN$. Here, by neglecting the overhead of tile/frame information, we use $\lceil \frac{BC_r}{s} \rceil$ to denote the number of packets per tile per frame. It is noteworthy that (8) is imposed by the absorbing barrier in the diffusion process [44].

Solving (6) with (7) and (8) yields

$$p(x_1, t_1 | 0) = \frac{1}{\sqrt{2\pi\beta_1 t_1}} \left[\exp \left\{ -\frac{(x_1 - \alpha_1 t_1)^2}{2\beta_1 t_1} \right\} - \exp \left\{ \frac{2\alpha_1 b_1}{\beta_1} - \frac{(x_1 - 2b_1 - \alpha_1 t_1)^2}{2\beta_1 t_1} \right\} \right]. \quad (9)$$

By substituting D_1 for t_1 into (9), we can obtain the PDF of $X_1(t_1)$ as $p(x_1, D_1 | 0)$.

Then, the PMF of G_l tiles cached by the MEC server in period l can be derived as:

$$\begin{aligned} P(G_l = g) &= P(D_1 f g \left\lceil \frac{BC_r}{s} \right\rceil \leq X_1(D_1) < D_1 f (g+1) \left\lceil \frac{BC_r}{s} \right\rceil) \\ &= \int_{D_1 f g \lceil \frac{BC_r}{s} \rceil}^{D_1 f (g+1) \lceil \frac{BC_r}{s} \rceil} p(x_1, D_1 | 0) dx \end{aligned} \quad (10)$$

where $g \in \mathbf{G} = \{0, 1, 2, \dots, MN\}$.

Video tiles are transmitted in the order of $\mathbf{A}_{o,l}$ at the caching stage, and caching matrix $\mathbf{A}_{c,l} = [a_{c,l}(n, m)]_{N \times M}$ is used to denote whether or not tile (n, m) is cached in the MEC server after this stage, where

$$a_{c,l}(n, m) = \begin{cases} 1 & a_{o,l}(n, m) \leq G_l \\ 0 & \text{otherwise.} \end{cases} \quad (11)$$

Therefore, the data of tiles with a higher prediction probability of falling in the FoV are more likely to be cached in the MEC server. Then, we can derive the PMF of $\mathbf{A}_{c,l}$ as $P(\mathbf{A}_{c,l}) = P(G_l = \sum_{n \in \mathbf{N}} \sum_{m \in \mathbf{M}} a_{c,l}(n, m))$.

B. Computation Delay $D_{2,l}$ of the MEC Server

As shown in Fig. 3, the MEC receives the actual FoV request information (n_l, m_l) at the beginning of the computation stage. We define request matrix $\mathbf{A}_{r,l} = [a_{r,l}(n, m)]_{N \times M}$ to denote whether or not each tile is in the actual FoV area, where

$$a_{r,l}(n, m) = \begin{cases} 1 & n \in \mathbf{N}_r, m \in \mathbf{M}_r \\ 0 & \text{otherwise.} \end{cases} \quad (12)$$

Because the actual FoV area may intersect with the border of video plane, we have $\mathbf{N}_r = \{n_l - \frac{k}{2}, n_l - \frac{k}{2} + 1, \dots, n_l, \dots, n_l + \frac{k}{2} - 1, n_l + \frac{k}{2}\} \cap \mathbf{N}$ and $\mathbf{M}_r = \{m_l - \frac{k}{2}, m_l - \frac{k}{2} + 1, \dots, m_l, \dots, m_l + \frac{k}{2} - 1, m_l + \frac{k}{2}\} \cap \mathbf{M}$, with \mathbf{N}_r and \mathbf{M}_r denoting the vertical and horizontal index sets of tiles in the actual FoV area. Let $E_l = \sum_{n \in \mathbf{N}_r} \sum_{m \in \mathbf{M}_r} a_{r,l}(n, m)$ represent the number of tiles in the actual FoV area in period l .

At the beginning of this computation stage, both caching matrix $\mathbf{A}_{c,l}$ and user viewpoint information (n_l, m_l) are known. The number of tiles which are cached in the MEC server and requested by the user can be obtained as

$$Q_l = \sum_{n \in \mathbf{N}_r} \sum_{m \in \mathbf{M}_r} a_{c,l}(n, m) a_{r,l}(n, m). \quad (13)$$

The hit ratio of tiles in the cache of the MEC server can be expressed as $H_r = Q_l / E_l$ and the PMF of H_r is denoted by $P(H_r)$. Therefore, the quantity of cached data hit by the MEC server is

$$\Gamma_{1,l} = D_1 f B Q_l. \quad (14)$$

To reduce the computation burden on the HMD and thus the computation delay, the MEC server decides to complete a portion of the computation task, in which the ratio between the

offloaded data and the total data is c . Hence, the offloaded data size $\Gamma_{2,l}$ is expressed as

$$\Gamma_{2,l} = \Gamma_{1,l}c. \quad (15)$$

Recalling that h denotes the data size ratio between computational output and input, the size of data processed by the MEC servers may increase to $\Gamma_{2,l} \cdot h$. Depending on the value of h , the performance gain from computation offloading at the MEC server may degrade if the capacity of the wireless link between the MEC server and the user is scarce.

Then, the conditional PMF of the computation delay of the MEC server $D_{2,l}$ is derived as:

$$P\left(D_{2,l} = \frac{\Gamma_{2,l}}{W_M} | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}\right) = 1. \quad (16)$$

C. Transmission Delay $D_{3,l}$ From the MEC Server to the HMD

The data requested by the user in the cache of the MEC server will be transmitted to the user, consisting of two components: one is computed in the MEC server, and the other is not computed yet. Let $\Gamma_{3,l}$ denote the overall data size, which is given as

$$\Gamma_{3,l} = \Gamma_{2,l}h + \Gamma_{1,l} - \Gamma_{2,l} = \Gamma_{1,l}(c \cdot h + 1 - c). \quad (17)$$

We denote the number of packets that the user receives from the MEC server at time t_2 from the beginning of the communication stage from the MEC server to the user by $Z_2(t_2)$. We model the packet transmission process from the MEC server to the user with a diffusion process by replacing the discrete number of packets, $Z_2(t_2)$, with a continuous process, $X_2(t_2)$. Here, the initial number of packets is 0, and the absorbing boundary is $b_{2,l} = \lceil \frac{\Gamma_{3,l}C_r}{s} \rceil$. Based on (9), the conditional PDF of transmission delay $D_{3,l}$ when $X_2(t_2) = b_{2,l}$ is given by

$$\begin{aligned} p_{D_{3,l}}(X_2(t_2) = b_{2,l}, t_2 | X_2(0) = 0, \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) \\ = \frac{1}{\sqrt{2\pi\beta_2 t_2}} \left[\exp\left\{-\frac{(b_{2,l} - \alpha_2 t_2)^2}{2\beta_2 t_2}\right\} \right. \\ \left. - \exp\left\{\frac{2\alpha_2 b_{2,l}}{\beta_2} - \frac{(-b_{2,l} - \alpha_2 t_2)^2}{2\beta_2 t_2}\right\} \right]. \end{aligned} \quad (18)$$

When the diffusion process ends, i.e., $X_2(t_2) = b_{2,l}$, the cumulative distribution function (CDF) of $D_{3,l}$ satisfies

$$\begin{aligned} P_{D_{3,l}}(D_{3,l} = t_2 | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) \\ = P(D_{3,l} \leq t_2 | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) \\ = 1 - \int_0^{b_{2,l}} p_{D_{3,l}}(y, t_2 | X_2(0) = 0, \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) dy. \end{aligned} \quad (19)$$

The PDF of $D_{3,l}$ is hence obtained as

$$\begin{aligned} p_{D_{3,l}}(D_{3,l} = t_2 | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) \\ = -\frac{d}{dt_2} \int_0^{b_{2,l}} p_{D_{3,l}}(y, t_2 | X_2(0) = 0, \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) dy. \end{aligned} \quad (20)$$

Solving (20) with (18) yields

$$\begin{aligned} p_{D_{3,l}}(D_{3,l} = t_2 | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) \\ = \frac{b_{2,l}}{\sqrt{2\pi\beta_2 t_2^3}} \exp\left\{-\frac{(b_{2,l} - \alpha_2 t_2)^2}{2\beta_2 t_2}\right\}. \end{aligned} \quad (21)$$

D. Transmission Delay $D_{4,l}$ From the Cloud Server to the HMD

The cloud server needs to deliver tiles that are not cached in the MEC server but are requested by the user, and the data size $\Gamma_{4,l}$ of this portion can be expressed as:

$$\Gamma_{4,l} = D_1 f B E_l - \Gamma_{1,l} = D_1 f B (E_l - Q_l). \quad (22)$$

We denote the number of packets that the user has received from the cloud server at time t_3 from the beginning of the communication between the cloud server and the user by $Z_3(t_3)$. We also model the packet transmission process from the cloud server to the user with a diffusion process by replacing the discrete number of packets, $Z_3(t_3)$, with a continuous process, $X_3(t_3)$. Here, the initial number of packets is 0, and the absorbing boundary is $b_{3,l} = \lceil \frac{\Gamma_{4,l}C_r}{s} \rceil$.

Based on (21), the conditional PDF of transmission delay $D_{4,l}$ is

$$\begin{aligned} p_{D_{4,l}}(D_{4,l} = t_3 | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) \\ = \frac{b_{3,l}}{\sqrt{2\pi\beta_3 t_3^3}} \exp\left\{-\frac{(b_{3,l} - \alpha_3 t_3)^2}{2\beta_3 t_3}\right\}. \end{aligned} \quad (23)$$

E. Computation Delay $D_{5,l}$ of the HMD

At this stage, the HMD completes the computation task of data that have not been rendered yet. The data consist of two components: one is received from the MEC server, and the other is received from the cloud server. Thus, the size of the data to be computed $\Gamma_{5,l}$ is given by

$$\Gamma_{5,l} = \Gamma_{4,l} + \Gamma_{1,l} - \Gamma_{2,l}. \quad (24)$$

Hence, we can determine the conditional PMF of computation delay $D_{5,l}$ as follows:

$$P\left(D_{5,l} = \frac{\Gamma_{5,l}}{W_U} | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}\right) = 1. \quad (25)$$

F. End-to-End Delay $D_{M,l}$

Solving (2) with (16), (21), (23), and (25), the conditional PDF of E2E delay $D_{M,l}$ can be derived as

$$\begin{aligned} p_{D_{M,l}}(D_{M,l} = t_l | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) \\ = \int_0^{t_l - D_{2,l} - D_{5,l}} p_{D_{3,l}}(t_2 | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) \\ \times p_{D_{4,l}}(t_l - D_{2,l} - D_{5,l} - t_2 | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) dt_2. \end{aligned} \quad (26)$$

Because the request matrix $\mathbf{A}_{r,l}$ is determined by the user's viewpoint, and the caching matrix $\mathbf{A}_{c,l}$ is related to the distribution of transmission rate from the cloud server to the MEC

server, $\mathbf{A}_{r,l}$ and $\mathbf{A}_{c,l}$ are independent. Consequently, the PMF of the request matrix is derived as $P(\mathbf{A}_{r,l}) = P_F(n_l, m_l)$. Hence, the PDF of E2E delay $D_{M,l}$ can be further calculated by

$$\begin{aligned}
 p(D_{M,l} = t_l) &= \sum_{\mathbf{A}_{c,l}} P(\mathbf{A}_{c,l}) \sum_{\mathbf{A}_{r,l}} P(\mathbf{A}_{r,l}) p_{D_{M,l}}(D_{M,l} = t_l | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}) \\
 &= \sum_{g \in \mathbf{G}_l} P(G_l = g) \sum_{n \in \mathbf{N}} \sum_{m \in \mathbf{M}} a_{c,l}(n, m) \\
 &\quad \times \sum_{n_l \in \mathbf{N}} \sum_{m_l \in \mathbf{M}} P_F(n_l, m_l) p_{D_{M,l}}(D_{M,l} = t_l | \mathbf{A}_{c,l}, \mathbf{A}_{r,l}).
 \end{aligned} \quad (27)$$

G. Delay Outage Probability

When the E2E delay is greater than the preset delay threshold D_{th} , the user will feel dizzy, and thus, the user QoE will decrease significantly [3], [9]. Consequently, the delay outage probability can be defined as the probability that the E2E delay $D_{M,l}$ exceeds the threshold:

$$P_{out} = P(D_{M,l} > D_{th}) = \int_{D_{th}}^{\infty} p(D_{M,l} = t_l) dt_l. \quad (28)$$

Based on the proposed analytical framework, it is straightforward to study how to fine-tune system parameters (e.g., video chunk duration D_1 and offloading ratio c) via, for example, numerical searching-based methods to minimize users' mean E2E delay or delay outage probability.

V. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed 360-degree MVRV streaming scheme and study the effect of diverse system parameters on the performance. In the following, we first describe the simulation settings and then present the simulation results.

A. Simulation Setup

We test the performance of the proposed method using the real video "BlueWorld," a 20-second video from Youtube [45]. The video is projected in ERP format, and each tile is encoded with the same bit rate. The resolution of the video is 4096×2048 , and the frame rate is 60 frames per second. In each simulation period, the simulator loads video frames from the video trace file and segments the same size video frame into UDP packets. To generate the FoV prediction results, we adopt the deep reinforcement learning method with CNN + LSTM network in [31].⁴ Specifically, the CNN architecture is given as C32-C32-C32-C32-FC288, where C and FC denote a convolutional layer and a fully connected layer, respectively, and the

⁴The reason why we do not use the head movement data sets (e.g., [46]) for comparison is that these data sets provide the head movement or eye movement of the VR users rather than the predicted viewpoint probability matrix. However, the latter is required in the proposed scheme to determine the order of video tiles to be proactively cached in the MEC server.

TABLE II
SIMULATION DEFAULT PARAMETERS

Variable	Default value
Typical network transmission rate $R(t)$	[0, 160] Mbps, uniform distribution
Transmission rate from the cloud server to the MEC server $R_1(t)$	$R(t)$
Transmission rate from the MEC server to the user $R_2(t)$	$10R(t)$
Transmission rate from the cloud server to the user $R_3(t)$	$R(t)$
Tile number in the vertical and horizontal directions in a projected video plane (M, N)	(8, 4)
Tile number in the vertical or horizontal direction in the FoV k	2
Frame per second of the video chunk f	60 fps
UDP packet size s	1460 bytes
Data size per tile per frame B	20 Mbit/frame/tile
Data size processed per second by the MEC server W_M	1.6 Tbps
Data size processed per second by the user W_U	40 Gbps
Data size changing ratio after computation h	1.5
Offloading ratio of raw video data c	0.99
Compression ratio of video chunk C_r	1/600
Typical actual caching time D_1	0.5 s
Preset E2E delay threshold D_{th}	0.02 s

number following each letter(s) specifies the number of kernels in that layer (e.g., C32 has 32 kernels). The LSTM network has a video frame sliding window of 10 video frames. For training and other network parameter settings, the interested reader can refer to [31] for details.

Without loss of generality, the transmission rate of the network among the cloud server, the MEC server, and the VR user are set based on [19], [23] and varies over time with a uniform distribution but different means and variances. The number of tiles in the vertical and horizontal directions in a projected video plane are 8 and 4, respectively [3], [47]. The data size per tile per frame is set to 20 Mbit [1], [19]. The compression ratio and preset E2E delay threshold are set based on [9], [33] as 1/600 and 0.02 s, respectively. In this work, we set the typical video chunk duration equal to a small value of 0.5 s [3], [47] to achieve a high prediction accuracy instead of using the long-term FoV prediction method in [35]. The default values of important variables used in the simulation are summarized in Table II. For each test, we conduct 5×10^5 simulation periods. To verify our analytical framework, the analytical results of the CDF of the E2E delay and that of the delay outage probability are compared with their simulation counterparts. To evaluate the advantage of enabling the MEC server, we also analyze and compare its performance with that of the MEC-disabled scheme, of which the E2E delay of any period l is denoted by $D_{N,l}$.

To study the benefit of proactively caching and computation offloading in the proposed scheme, we also compared our scheme with a bit-rate adaption-based MEC-disabled DASH 360-degree VR streaming algorithm [48].⁵ In the DASH algorithm, the cloud server preprocesses the tiles with SVC technology as a two-layer coded video, including one base layer and one enhancement layer. The user predicts the average bandwidth of

⁵To the best of our knowledge, in the literature, there is no existing work on MEC-based DASH scheme. Designing such a scheme is a meaningful research topic but beyond the scope of this paper.

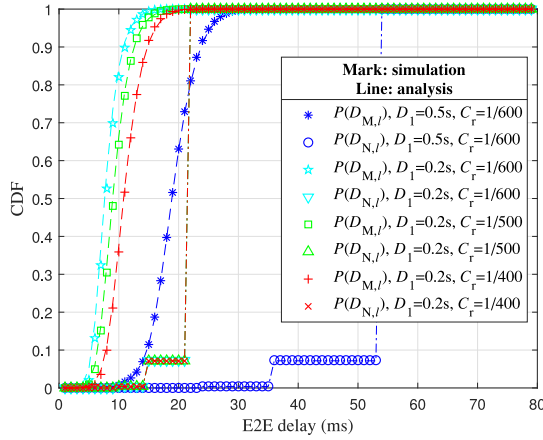


Fig. 5. CDF of E2E delay with different video chunk durations and compression ratios in MEC-enabled and MEC-disabled schemes.

the current period by utilizing the average bandwidth of the past ten periods [49]. At the beginning of each caching stage, the user decides the bit rate of tiles based on the following rules: First, at least the base layer of each tile should be transmitted to the user; second, the enhancement layer of a tile will be transmitted to the user only if the predicted bandwidth still can accommodate it after providing the enhancement layer of other tiles with a higher predicted viewpoint probability. In general, the user tries to make the total bit rate of the video streaming equal to the predicted bandwidth, thus provisioning the smoothness of playback and achieving a low E2E latency.

B. Impact of Video Chunk Duration, Compression Ratio, and Prediction Setting

Fig. 5 compares the E2E delay of the MEC-enabled scheme with that of the MEC-disabled scheme at different video chunk durations and compression ratios. The simulation results match well with the analytical results. Comparing the MEC-enabled scheme with the MEC-disabled scheme at the same video chunk duration and compression ratio, the latency of the MEC-enabled scheme is shown to be superior to that of the MEC-disabled scheme, with a significant decrease of up to 60% on average. This is because the tiles requested by the user are cached in the MEC server, and the computation delay is smaller because the computation capacity of the MEC server is much higher than that of the user's HMD. Given the compression ratio (e.g., 1/600) in the MEC-enabled scheme or MEC-disabled scheme, the average E2E delay with a shorter video chunk duration is below that with a longer video chunk duration. This is because when D_1 is shorter, the data quantity of the video chunk required to be transmitted and computed in each period is less. Additionally, for a given video chunk duration (e.g., $D_1 = 0.2$ s), the cumulative probability of the E2E delay in the MEC-enabled scheme decreases with increasing C_r due to the larger quantity of data to be transmitted. However, when $D_1 = 0.2$ s, the cumulative probability of the E2E delay with the MEC-disabled scheme remains unchanged as the compression ratio C_r increases from 1/600 to 1/400. The reason is that the size of data to be cached

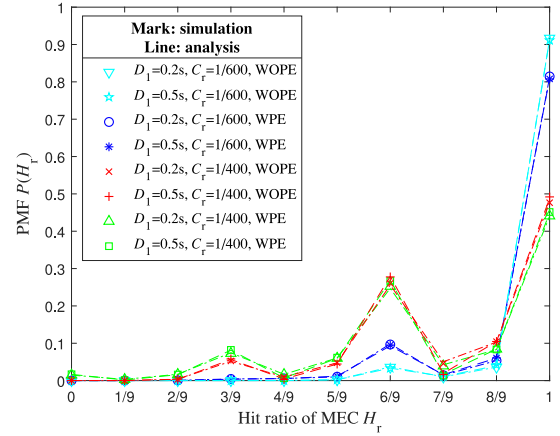


Fig. 6. PMF of the MEC server's hit ratio with different video chunk durations, compression ratios, and prediction settings.

when $D_1 = 0.2$ s is smaller than that when $D_1 = 0.5$ s, and the transmission rate from the cloud server to the user is high enough to cache all video chunk data with compression ratio changing from 1/600 to 1/400. In addition, the computation delay remains unchanged as the compression ratio increases because the computation delay in the proposed analytical model is proportional to the size of the raw data to be processed, which does not vary with changes in the compression ratio. As such, the total delay of computation and communication remains unchanged in this case.

In the MEC-disabled scheme, the CDF of the E2E delay has a step pattern when $D_{N,l} = 24$ ms, 36 ms, and 54 ms. The reason is that given the number k of tiles in the vertical or horizontal direction in the FoV is equal to 2, the number of tiles in the request area E_l might be 4, 6, or 9, which depends on the tile that the viewpoint falls in. Notice that for the MEC-disabled scheme, the requested video chunk tiles can be cached and processed by the HMD directly when the prediction accuracy and transmission rate from the cloud server to the user are high enough. Hence, its E2E delay only consists of computation delay of E_l requested tiles, which are exactly equal to 24 ms, 36 ms, and 54 ms.

In the proposed MEC-enabled 360-degree MVRV streaming scheme, during the MEC server's caching stage tiles will be transmitted from the cloud server to the MEC server with priority based on the FoV prediction result generated at the beginning of this stage, while the actual FoV result is calculated at the end of this stage. Thus, the prediction accuracy highly depends on the video chunk duration. The impact of the video chunk duration and thus the impact of the prediction accuracy are studied in Fig. 6 in terms of the PMF of the MEC server's hit ratio. A big $P(H_r)$ (e.g., 1) indicates that the H_r ratio of the tiles can be cached proactively with a high probability in the MEC server. We investigate the change in $P(H_r)$ because we want to identify methods of parameter configuration in the proposed framework to improve the probability of a high hit ratio. Thus, we evaluate the performance of the MEC-enabled scheme with prediction error (WPE) and that of the scheme without prediction error (WOPE). The predicted viewpoint distribution of the former is

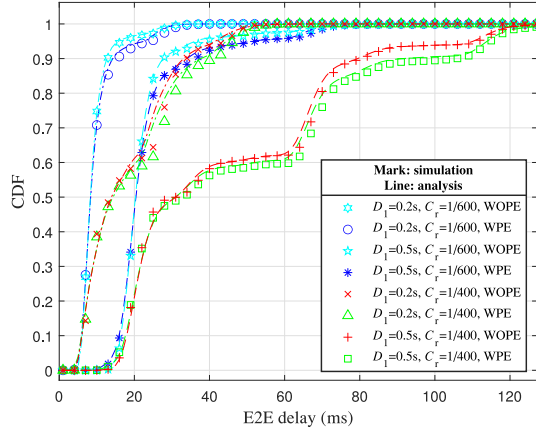


Fig. 7. CDF of E2E delay with different video chunk durations, compression ratios, and prediction settings.

produced by the prediction mechanism at the beginning of the caching stage, while the predicted viewpoint distribution of the latter is set as the actual viewpoint distribution by assuming it is known *a priori*. Thus, in general, the prediction accuracy decreases if the video chunk duration increases. Fig. 6 shows that, given the same video chunk duration and compression ratio for video tiles, the probability of hit ratio equal to 1 with the WPE scheme is lower than that with the WOPE scheme. For example, when $D_1 = 0.5$ s and $C_r = 1/600$, $P(H_r = 1)$ reduces from 90% to 80%, highlighting the impact of prediction accuracy. Additionally, given the same video chunk duration (e.g., 0.5 s) in the WPE and WOPE schemes, the probability of a hit ratio equal to 1 decreases by 37% and 44%, respectively, as the compression ratio increases from $1/600$ to $1/400$. This is because, given the same data to be cached, the quantity of data to be transmitted is smaller with a smaller compression ratio. Thus, the probability of caching all requested video tiles successfully is larger in the MEC caching stage if the compression ratio is smaller. On the other hand, given the same compression ratio (e.g., $1/600$), the probability of the same hit ratio (e.g., $P(H_r = 1)$) with the WPE or WOPE schemes has almost no change as the video chunk duration increases from 0.2 s to 0.5 s. This is because when the video duration is larger, more video data need to be transmitted, while the duration for caching video data increases correspondingly. Thus, the probability of a hit ratio $H_r = 1$ only changes slightly. Consequently, the hit ratio can be improved by reducing the compression ratio, while the impact of video chunk duration is not significant in such a simulation setting.

Fig. 7 shows the CDF results of the E2E delay with the MEC-enabled scheme, given the same settings with Fig. 6. Similar to the hit ratio discussed above, given the same video chunk duration and compression ratio, the cumulative probability of any E2E delay with the WPE scheme is lower than that of the WOPE scheme. The reason is that the prediction's deviation, which is shown as the difference between the FoV predicted probability $\hat{\mathbf{P}}_F(\hat{n}_l, \hat{m}_l)$ and actual probability matrix $\mathbf{P}_F(n_l, m_l)$, can cause the hit ratio to decrease; thus, the delay performance also decreases. For example, when $D_1 = 0.5$ s and

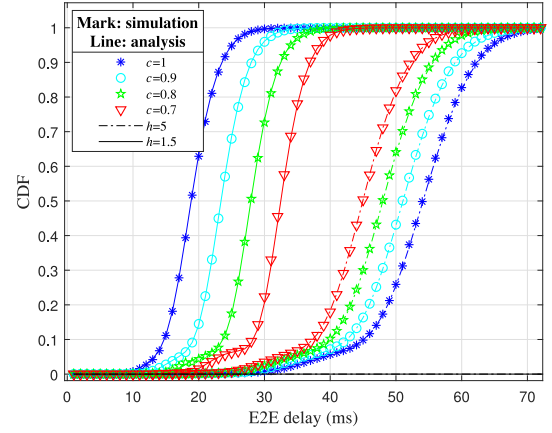


Fig. 8. CDF of E2E delay with different offloading ratios and data size changing ratios.

$C_r = 1/600$, $P(H_r = 1)$ in the WPE scheme decreases by 10% compared with that of the WOPE scheme in Fig. 6, and the cumulative probability of the WPE scheme decreases by 6% when the E2E delay is approximately 24 ms compared with that of the WOPE scheme. Fig. 7 also shows that given a fixed video chunk duration in either the WPE or WOPE schemes, the average E2E delay increases markedly as the compression ratio increases. The reason is that the probability of a hit ratio equal to 1 decreases as the compression ratio increases, and more tiles must be transmitted from the cloud server to the user with a larger probability. Moreover, given a fixed compression ratio in either the WPE or WOPE schemes, the cumulative probability decreases significantly when the video chunk duration increases from 0.2 s to 0.5 s because the quantity of data to be processed with $D_1 = 0.5$ s is above that with $D_1 = 0.2$ s, while the transmission delay remains nearly unchanged.

C. Impact of MEC Computation Offloading Ratio

Fig. 8 describes the impact of the MEC server's computation offloading ratio c , given different data size changing ratios h between computation output and input. The average E2E delay decreases as the computation offloading ratio decreases when $h = 5$. Although the MEC server can reduce computation latency, the transmission delay from the MEC server to the user increases too much in this case due to the increased quantity of data to be transmitted to the user after computation in the MEC server. However, the result reverses when $h = 1.5$. In fact, there is a tradeoff for the MEC server between the computation delay and data transmission delay, which mainly depends on the transmission rate from the MEC server to the user and the change in the ratio of the quantity of data between computation output and input. Moreover, it is not always beneficial to set a large computation offloading ratio because offloading gain can be reduced by the change in the ratio h of the quantity of data between computation output and input, and by the transmission rate from the MEC server to the user as well.

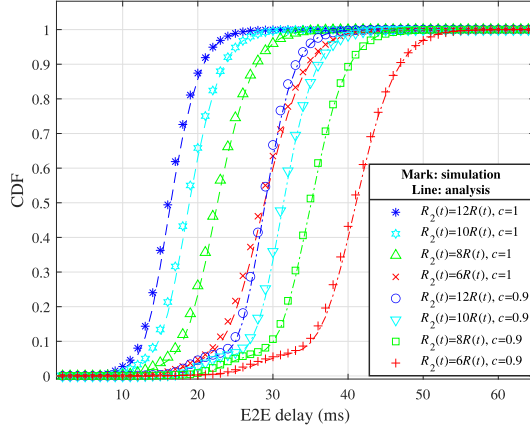


Fig. 9. CDF of E2E delay with different transmission rates from the MEC server to the user and offloading ratios.

D. Impact of Transmission Rate From the MEC Server to the User and Offloading Ratio

Fig. 9 shows the CDF of the E2E delay under the MEC-enabled scheme given different transmission rates from the MEC server to the user at offloading ratio c equal to 0.9 and 1. It is straightforward that given a fixed offloading ratio, the average E2E delay decreases as the transmission rate from the MEC server to the user increases. Note that, compared with the data size of the video chunk, if the transmission rate from the cloud server to the MEC server is sufficiently high, as in our simulation setting, the tiles requested by the user can be cached proactively in the MEC server with a high probability. Hence, the main component of the transmission delay here is the transmission delay from the MEC server to the user, $D_{3,l}$, and in this case, the transmission delay is mainly affected by the link from the MEC server to the user. In addition, the average E2E delay decreases less as the transmission rate from the MEC server to the user increases. This finding implies that, although the performance of the system can be improved by using more resources to increase the transmission rate from the MEC server to the user, $R_2(t)$, the associate gain can decrease due to a new bottleneck such as computation delay in the system. Comparing the curve of $R_2(t) = 6R(t)$ and $c = 1$ and that of $R_2(t) = 12R(t)$ and $c = 0.9$, offloading more computation tasks at the MEC server given a low transmission rate from the MEC to the user is shown to have a similar effect with improving the capacity of the offloading link.

E. Impact of Transmission Rate From the Cloud Server to the MEC Server and Computation Capacity of the MEC Server

The CDF of the E2E delay versus transmission rate $R_1(t)$ from the cloud server to the MEC server and computation capacity W_M of the MEC server in the MEC-enabled scheme is shown in Fig. 10. Given the capacity of the MEC server (e.g., 1.6 Tbps), the average E2E delay of the MEC-enabled scheme increases markedly as the transmission rate from the cloud server to the MEC server decreases from $0.8R(t)$ to $0.4R(t)$. When the

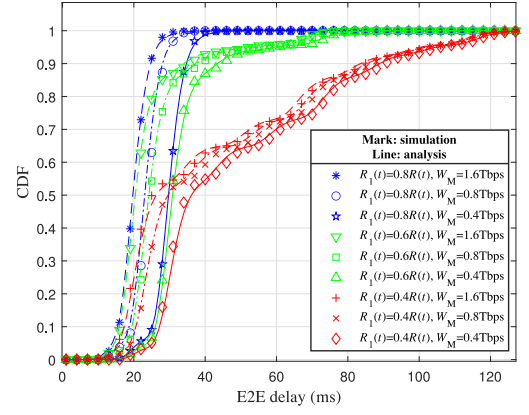


Fig. 10. CDF of E2E delay with different transmission rates from the cloud server to the MEC server and computation capacities of the MEC server.

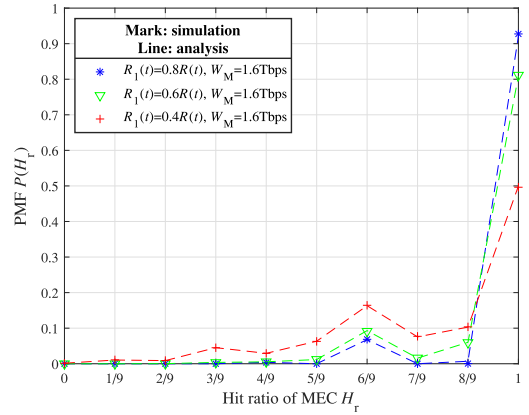


Fig. 11. Hit ratio of the MEC server with different transmission rates from the cloud server to the MEC server and computation capacities of the MEC server.

transmission rate from the cloud server to the MEC server decreases, the probability of a hit ratio equal to 1 decreases greatly with the MEC-enabled scheme, as shown in Fig. 11. Therefore, more video tiles should be transmitted directly from the cloud server to the user, and the transmission delay becomes larger accordingly. Fig. 10 also shows that given the transmission rate from the cloud server to the MEC server (e.g., $0.8R(t)$), the probability of the E2E delay decreases significantly (e.g., 40%) at a small E2E delay (e.g., 21 ms) when the computation capacity decreases from 1.6 Tbps to 0.8 Tbps. Furthermore, there is a tradeoff between the computation delay and communication delay. For example, the average E2E delay of the scheme with transmission rate from the cloud server to the MEC server $0.6R(t)$ and computation capacity of the MEC server 1.6 Tbps is approximately equal to that of the scheme with a transmission rate from the cloud server to the MEC server equal to $0.8R(t)$ but a computation capacity of the MEC server equal to 0.8 Tbps.

F. Delay Outage Probability

The delay outage probability with different transmission rates from the MEC server to the user and computation offloading ratios at the data size changing ratio $h = 1.5$ are shown in

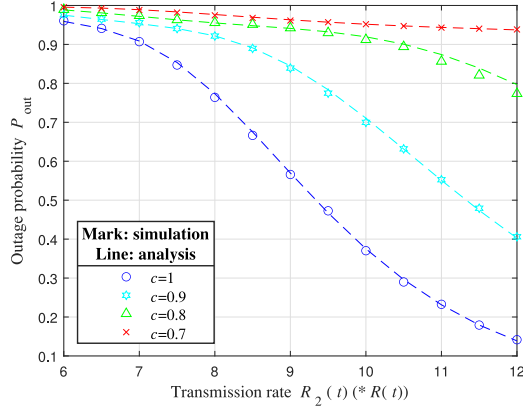


Fig. 12. The delay outage probability with different transmission rates from the MEC server to the user and computation offloading ratios.

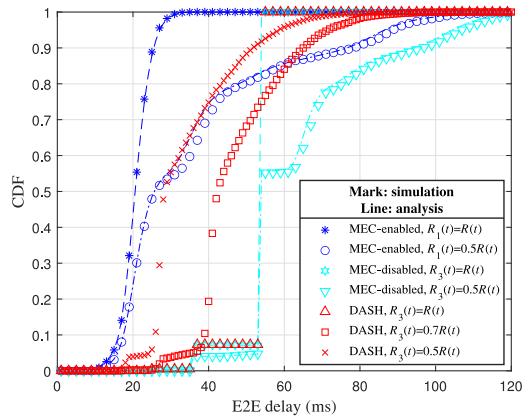


Fig. 13. CDF of E2E delay with different $R_1(t)$'s in the MEC-enabled scheme and $R_3(t)$'s in the MEC-disabled scheme and the DASH scheme.

Fig. 12. The simulation results match well with the analytical results, which implies the effectiveness of the proposed analytical framework in analyzing and predicting the performance of the proposed MVRV streaming system. However, the delay outage probability is affected by many factors, as shown in the proposed analytical framework. Thus, how to determine an appropriate computation offloading ratio of the MEC server, when many parameters of the system are known, for example, the computing capability of MEC server W_M , and the data size change ratio after computation h , must be considered carefully.

G. Comparison With DASH

Fig. 13 compares the MEC-enabled and MEC-disabled schemes with the DASH scheme in terms of the CDF of the E2E delay. For the MEC-enabled scheme, we use different transmission rates from the cloud server to the MEC server $R_1(t)$'s; for the MEC-disabled scheme and the DASH scheme, we use different transmission rates from the cloud server to the user $R_3(t)$'s. Fig. 13 shows that the average E2E delay of the DASH scheme decreases as the transmission rate from the cloud server to the user $R_3(t)$ decreases, which is different from the MEC-enabled and MEC-disabled schemes. The reason is that in the DASH

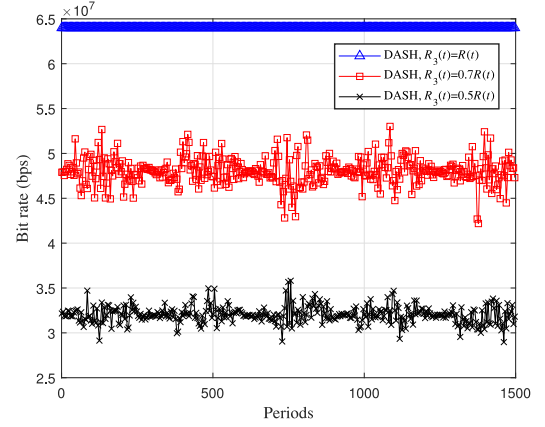


Fig. 14. Bit rate of each period in the DASH scheme.

scheme, the bit rate of the video stream is adapted to the predicted bandwidth, as shown in Fig. 14. Thus, when $R_3(t)$ decreases, the quantity of data to be transmitted from the cloud server to the user and then processed at the HMD decreases. Compared to the MEC-enabled scheme, the average E2E delay of the DASH scheme is larger because the DASH scheme cannot benefit from computation offloading due to the MEC server. However, compared with the MEC-disabled scheme, the average E2E delay of the DASH scheme is smaller, especially when the bandwidth is insufficient to support the bit rate of the video stream in the MEC-disabled scheme. However, the low latency of the DASH scheme is achieved at the expense of a smaller bit rate, which can also reduce user QoE, while the MEC-disabled and MEC-enabled schemes always maintain the highest bit rate for transmitting both the base layer and enhancement layer. When the average bandwidth is sufficient to cache all video data (e.g., $R_3(t) = R(t)$), the CDF of E2E delay of the DASH and MEC-disabled schemes become identical because in this case the E2E delay depends only on the computation delay.

VI. CONCLUSION

In this paper, we have proposed a novel MEC-based 360-degree MVRV streaming scheme, which utilizes the user's FoV to cache data proactively and offloads partial computation tasks to the MEC server within a designed period. By applying diffusion process, we derive the closed-form expressions of the E2E delay and delay outage probability of the system, which are considered to be important QoS indicators of the MVRV services. Based on the proposed analytical framework, the effects of system parameters, including network transmission rate, MEC computation resource allocation strategy, and video chunk duration, can be jointly evaluated to maximize the user QoE. Conversely, given a user's specific requirements for the E2E delay and delay outage probability, we can determine the required parameter setting and deploy the appropriate system. Furthermore, the simulation results verify the accuracy of the analysis and the effectiveness of the proposed MVRV streaming scheme in reducing the E2E delay. In future work, we plan to study the performance of the proposed MVRV streaming scheme based on long-term prediction mechanisms and explore the benefit of

multirate transmission algorithms. In addition, the impacts of channel model, power consumption of the HMD, and congestion and reliable transmission control will be considered.

REFERENCES

- [1] Huawei iLab, "Cloud VR solution whitepaper," Accessed Aug. 26, 2020. [Online] Available: https://www-file.huawei.com/-/media/corporate/pdf/ilab/2018/cloud_vr_solutions_wp_cn.pdf?source=corp_comm
- [2] B. Soret, P. Mogensen, K. I. Pedersen, and M. C. Aguayo-Torres, "Fundamental tradeoffs among reliability, latency and throughput in cellular networks," in *Proc. IEEE Globecom Workshops*, Dec. 2014, pp. 1391–1396.
- [3] M. Zink, R. Sitarman, and K. Nahrstedt, "Scalable 360° video stream delivery: Challenges, solutions, and opportunities," *Proc. IEEE*, vol. 107, no. 4, pp. 639–650, Apr. 2019.
- [4] M. Erol-Kantarci and S. Sukhmani, "Caching and computing at the edge for mobile augmented reality and virtual reality (AR/VR) in 5 G," in *Ad Hoc Networks*, Y. Zhou and T. Kunz, Eds., Cham Switzerland: Springer, 2018.
- [5] Y. Sun, Z. Chen, M. Tao, and H. Liu, "Communications, caching, and computing for mobile virtual reality: Modeling and tradeoff," *IEEE Trans. Commun.*, vol. 67, no. 11, pp. 7573–7586, Nov. 2019.
- [6] J. Chakareski, "VR/AR immersive communication: Caching, edge computing, and transmission trade-offs," in *Proc. ACM Workshop Virtual Reality Augmented Reality Netw.*, 2017, pp. 36–41.
- [7] X. Ge, L. Pan, Q. Li, G. Mao, and S. Tu, "Multipath cooperative communications networks for augmented and virtual reality transmission," *IEEE Trans. Multimedia*, vol. 19, no. 10, pp. 2345–2358, Oct. 2017.
- [8] J. Park, P. Popovski, and O. Simeone, "Minimizing latency to support VR social interactions over wireless cellular systems via bandwidth allocation," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 776–779, Oct. 2018.
- [9] M. S. Elbamby, C. Perfecto, M. Bennis, and K. Doppler, "Toward low-latency and ultra-reliable virtual reality," *IEEE Netw.*, vol. 32, no. 2, pp. 78–84, Mar. 2018.
- [10] Y. Chen *et al.*, "Cooperative communications in ultra-wideband wireless body area networks: Channel modeling and system diversity analysis," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 1, pp. 5–16, Jan. 2009.
- [11] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proc. 5th Workshop Things Cellular: Operations, Appl. Challenges*, 2016, pp. 1–6.
- [12] K. Lee *et al.*, "Outatime: Using speculation to enable low-latency continuous interaction for mobile cloud gaming," in *Proc. Int. Conf. Mobile Syst. Appl., and Serv.*, 2015, pp. 151–165.
- [13] M. Chen, W. Saad, and C. Yin, "Virtual reality over wireless networks: Quality-of-service model and learning-based resource management," *IEEE Trans. Commun.*, vol. 66, no. 11, pp. 5621–5635, Nov. 2018.
- [14] V. W. S. Wong, R. Schober, D. W. K. Ng, and L.-C. Wang, *Key Technologies for 5 G Wireless Systems*. Cambridge, U.K.: Cambridge Univ. Press, 2017.
- [15] X. Yang *et al.*, "Communication-constrained mobile edge computing systems for wireless virtual reality: Scheduling and tradeoff," *IEEE Access*, vol. 6, pp. 16 665–16 677, Mar. 2018.
- [16] S. Sukhmani, M. Sadeghi, M. Erol-Kantarci, and A. El Saddik, "Edge caching and computing in 5G for mobile AR/VR and tactile internet," *IEEE MultiMedia*, vol. 26, no. 1, pp. 21–30, Jan. 2019.
- [17] L. Zhang, Y. Xu, W. Huang, L. Yang, J. Sun, and W. Zhang, "A MMT-based content classification scheme for vod service," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast.*, Jun. 2015, pp. 1–5.
- [18] J. Yang, Q. Yang, K. S. Kwak, and R. R. Rao, "Power-delay tradeoff in wireless powered communication networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3280–3292, Apr. 2017.
- [19] T. T. Le, D. V. Nguyen, and E. Ryu, "Computing offloading over mmWave for mobile VR: Make 360 video streaming alive," *IEEE Access*, vol. 6, pp. 66 576–66 589, Sep. 2018.
- [20] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," in *Proc. ACM Conf. SIGCOMM*, Aug. 2014, pp. 187–198.
- [21] C. Zhou, C. Lin, and Z. Guo, "mDASH: A markov decision-based rate adaptation approach for dynamic http streaming," *IEEE Trans. Multimedia*, vol. 18, no. 4, pp. 738–751, Apr. 2016.
- [22] Standard ISO/IEC 23009-1, "MPEG-DASH (Dynamic Adaptive Streaming Over HTTP)," Accessed on: Aug. 26, 2020. [Online]. Available: https://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-DASH_schema_files/
- [23] T. H. Luan, L. X. Cai, and X. Shen, "Impact of network dynamics on user's video quality: Analytical framework and QoS provision," *IEEE Trans. Multimedia*, vol. 12, no. 1, pp. 64–78, Jan. 2010.
- [24] Y. Zhang, P. Zhao, K. Bian, Y. Liu, L. Song, and X. Li, "DRL360: 360-degree video streaming with deep reinforcement learning," in *Proc. IEEE INFOCOM*, Apr. 2019, pp. 1252–1260.
- [25] 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 *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 991–999.
- [26] T. Judd, K. Ehinger, F. Durand, and A. Torralba, "Learning to predict where humans look," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sep. 2009, pp. 2106–2113.
- [27] S. Dodge and L. Karam, "Visual saliency prediction using a mixture of deep neural networks," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 4080–4090, Aug. 2018.
- [28] M. Kümmerer, L. Theis, and M. Bethge, "Deep gaze I: Boosting saliency prediction with feature maps trained on imagenet," in *Proc. Int. Conf. Learn. Representations Workshop*, 2015, pp. 1–12.
- [29] E. Vig, M. Dorr, and D. Cox, "Large-scale optimization of hierarchical features for saliency prediction in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 2798–2805.
- [30] L. Bazzani, H. Larochelle, and L. Torresani, "Recurrent mixture density network for spatiotemporal visual attention," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 1–17.
- [31] M. Xu, Y. Song, J. Wang, M. Qiao, L. Huo, and Z. Wang, "Predicting head movement in panoramic video: A deep reinforcement learning approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2693–2708, Nov. 2019.
- [32] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Int. Joint Conf. Artif. Intell.*, vol. 2, 1981, pp. 674–679.
- [33] S. LaValle, A. Yershova, M. Katsev, and M. Antonov, "Head tracking for the oculus rift," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 187–194.
- [34] C. Fan, J. Lee, W. Lo, C. Huang, K. Chen, and C. Hsu, "Fixation prediction for 360 degree video streaming in head-mounted virtual reality," in *Proc. ACM Workshop Netw. Operating Syst. Support Digit. Audio Video*, 2017, pp. 67–72.
- [35] J. Park and K. Nahrstedt, "Navigation graph for tiled media streaming," in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 447–455.
- [36] B. Hayes, Y. Chang, and G. Riley, "Controlled unfair adaptive 360 vr video delivery over an MPTCP/QUIC architecture," in *Proc. IEEE Int. Conf. Commun.*, 2018, pp. 1–6.
- [37] C. Yin, Z. Chen, Y. Hu, and K. Yu, "Fine-grained transmission optimization of large-scale web VR scenes," in *Proc. IEEE Int. Conf. Prog. Inf. Comput.*, 2018, pp. 209–214.
- [38] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 4489–4497.
- [39] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Trans. Wirel. Commun.*, vol. 17, no. 8, pp. 5506–5519, Aug. 2018.
- [40] H. A. Alameddine, S. Sharafeddine, S. Sebbah, S. Ayoubi, and C. Assi, "Dynamic task offloading and scheduling for low-latency IOT services in multi-access edge computing," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 668–682, Mar. 2019.
- [41] L. Kleinrock, *Queueing Systems. volume 2: Comput. Appl.*, New York, NY, USA: Leonard Kleinrock Wiley, Oct. 2006.
- [42] G. Louchard and G. Latouche, *Probability Theory and Computer Science*. New York, NY, USA: Academic Press, 1983.
- [43] A. Duda, "Transient diffusion approximation for some queueing systems," in *Proc. ACM SIGMETRICS Conf. Meas. Model. Comput. Syst.*, 1983, pp. 118–128.
- [44] D. R. Cox and H. D. Miller, *The Theory of Stochastic Processes*. U.K.: Champan & Hall/CRC, 1977.
- [45] GoPro, "Swimming with wild Dolphins in the ocean," Accessed on: Jul. 31, 2019. [Online]. Available: https://www.youtube.com/watch?v=9_XPYtiMWE
- [46] E. J. David, J. Gutiérrez, A. Coutrot, M. P. D. Silva, and P. L. Callet, "A dataset of head and eye movements for 360 videos," in *Proc. ACM Multimedia Syst. Conf.*, 2018, pp. 432–437.
- [47] D. V. Nguyen, H. T. T. Tran, A. T. Pham, and T. C. Thang, "An optimal tile-based approach for viewport-adaptive 360-degree video streaming," *IEEE J. Emerg. Sel. Top. Circuits Syst.*, vol. 9, no. 1, pp. 29–42, Feb. 2019.

- [48] A. T. Nasrabadi, A. Mahzari, J. D. Beshay, and R. Prakash, "Adaptive 360-degree video streaming using scalable video coding," in *Proc. ACM Int. Conf. Multimedia*, 2017, pp. 1689–1697.
- [49] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo, "360ProbDash: Improving QoE of 360 video streaming using tile-based HTTP adaptive streaming," in *Proc. ACM Int. Conf. Multimedia*, 2017, pp. 315–323.



Qi Cheng received the B.S. degree in information engineering from Zhejiang University, Hangzhou, China, in 2019. He is currently working toward the M.S. degree in electronics and communication engineering with Zhejiang University, Zhejiang, China. His research interests include multimedia communication, network economy, and reinforcement learning.



Hanguan Shan (Member, IEEE) received the B.Sc. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2004, and the Ph.D. degree in electrical engineering from Fudan University, Shanghai, China, in 2009. From 2009 to 2010, he was a Postdoctoral Research Fellow with the University of Waterloo, Waterloo, ON, Canada. Since 2011, he has been with the College of Information Science Electronic Engineering, Zhejiang University, where he is currently an Associate Professor. He is also with the Zhejiang Provincial Key Laboratory of Information

Processing Communication Networks, Hangzhou and SUTD-ZJU IDEA, Hangzhou, China. His current research interests include cross-layer protocol design, resource allocation, and the quality-of-service provisioning in wireless networks. Dr. Shan was a Technical Program Committee Member of various international conferences, including the *IEEE Global Communications Conference*, the *IEEE International Conference on Communications*, *IEEE Wireless Communications and Networking Conference*, and *IEEE Vehicular Technology Conference*. He was the co-recipient of the Best Industry Paper Award from the 2011 IEEE WCNC held in Quintana Roo, Mexico. He was the Editor of the IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING.



Weihua Zhuang (Fellow, IEEE) has been with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, since 1993, where she is currently a Professor and the Tier I Canada Research Chair of wireless communication networks. Dr. Zhuang is a Fellow of the Royal Society of Canada, the Canadian Academy of Engineering, and the Engineering Institute of Canada. She is also an elected Member of the Board of Governors and VP Publications of the IEEE Vehicular Technology Society. She was the recipient of the 2017 Techni-

cal Recognition Award from the IEEE Communications Society Ad Hoc and Sensor Networks Technical Committee and the co-recipient of several best paper awards from IEEE conferences. She was the Editor-in-Chief of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from 2007 to 2013, the Technical Program Chair or Co-Chair of the IEEE Vehicular Technology Conference Fall 2017 and Fall 2016, and the Technical Program Symposia Chair of the IEEE Global Communications Conference 2011.



Lu Yu (Member, IEEE) received the B.Eng. degree in radio engineering and the Ph.D. degree in communication and systems from Zhejiang University, Hangzhou, China, in 1991 and 1996, respectively. Since 2003, she has been a Professor with the Institute of Information and Communication Engineering, Zhejiang University. In 2002, she was a Senior Visiting Scholar with University Hannover, Hanover, Germany, supported by China Scholarship Council and German Research Foundation (DFG). In 2004, she was a Senior Visiting Scholar with the Chinese University of Hong Kong, Hong Kong, supported by the United College Resident Fellow Scheme. Her research interests include video coding, multimedia communication, and relative ASIC design. Prof. Yu was the Area Editor of the *EURASIP Journal Signal Processing: Image Communication*. She is the General Chair of the Picture Coding Symposium (PCS) 2019, and also the International Steering Committee Member of PCS. She was the Chair of the Video-Subgroup of Audio Video Coding Standard of China from 2005 to 2017 and the Membership and Election Subcommittee of Technical Committee of Visual Signal Processing and Communication. She organized the Packet Video Workshop 2006 as the General Chair and IEEE Workshop of Multimedia Signal Processing 2011 as the Local Chair. She currently acts as the Video Subgroup Chair of ISO/IEC JTC1 SC29 WG11, which is known as MPEG and the elected Chair of the Technical Committee of Education and Outreach of IEEE Society of Circuits and Systems.



Zhaoyang Zhang (Member, IEEE) received the Ph.D. degree from Zhejiang University, Hangzhou, China, in 1998. He is currently a Qiusi Distinguished Professor with Zhejiang University. He has coauthored more than 300 peer-reviewed international journal articles and conference papers. His current research interests include fundamental aspects of wireless communications and networking, which include information theory and coding, network signal processing and distributed learning, AI-empowered communications and networking, network intelligence with synergetic sensing, and computation and communication. He was the co-recipient of seven conference best paper awards including ICC 2019. He was the recipient of the National Natural Science Fund for Distinguished Young Scholars by NSFC in 2017. He is or was the Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON COMMUNICATIONS, and *IET Communications*. He was the General Chair, the TPC Co-Chair or the Symposium Co-Chair of the WCSP 2013–2018, the Globecom 2014 Wireless Communications Symposium, and the VTC-Spring 2017 Workshop HMWC.



Tony Q.S. Quek (Fellow, IEEE) received the B.E. and M.E. degrees in electrical and electronics engineering from the Tokyo Institute of Technology, Tokyo, Japan, in 1998 and 2000, respectively, and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2008. He is currently a Cheng Tsang Man Chair Professor with the Singapore University of Technology and Design (SUTD), Singapore. He is also the Head of the ISTD Pillar, the Sector Lead of the SUTD AI Program, and the Deputy Director of the SUTD-ZJU IDEA. His current research interests include wireless communications and networking, network intelligence, the Internet of Things, URLLC, and big data processing. Dr. Quek is an Elected Member of the IEEE Signal Processing Society SPCOM Technical Committee. He was an Executive Editorial Committee Member of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He was the recipient of the 2008 Philip Yeo Prize for Outstanding Achievement in Research, the 2012 IEEE William R. Bennett Prize, the 2015 SUTD Outstanding Education Awards Excellence in Research, the 2016 IEEE Signal Processing Society Young Author Best Paper Award, the 2017 CTTC Early Achievement Award, the 2017 IEEE ComSoc AP Outstanding Paper Award, the 2020 IEEE Communications Society Young Author Best Paper Award, the 2020 IEEE Stephen O. Rice Prize, and the 2016–2019 Clarivate Analytics Highly Cited Researcher. He has been actively involved in organizing and chairing sessions and was a Member of the Technical Program Committee and symposium chairs in a number of international conferences. He is the Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and the Chair of the IEEE VTS Technical Committee on Deep Learning for Wireless Communications. He was the Editor of the IEEE TRANSACTIONS ON COMMUNICATIONS and IEEE WIRELESS COMMUNICATIONS LETTERS. He is a Distinguished Lecturer of the IEEE Communications Society.