Live 360° Video Streaming to Heterogeneous Clients in 5G Networks

Jacob Chakareski and Mahmudur Khan

***Abstract*—We investigate rate-distortion-computing optimized live 360° video streaming to heterogeneous mobile VR clients in 5G networks. The client population comprises devices that feature single (LTE) or dual (LTE/NR) cellular connectivity. The content is compressed using scalable 360° tiling at the origin and sent towards the clients over a single backbone network link. A mobile edge server then adapts the incoming streaming data to the individual clients and their respective down-link transmission rates using formal rate-distortion-computing optimization. Single connectivity clients are served by the edge server a baseline representation/layer of the content adapted to their down-link transmission capacity and device computing capability. A dual connectivity client is served in parallel a baseline content layer on its LTE connectivity and a complementary viewport-specific enhancement layer on its NR connectivity, synergistically adapted to the respective down-links' transmission capacities and its computing capability. We formulate two optimization problems to conduct the operation of the edge server in each case, taking into account the key system components of the delivery process and induced end-to-end latency, aiming to maximize the immersion fidelity delivered to each client. We explore respective geometric programming optimization strategies that compute the optimal solutions at lower complexity. We rigorously analyze the computational complexity of the two optimization algorithms we formulate. In our evaluation, we demonstrate considerable performance gains over multiple assessment factors relative to two state-of-the-art techniques. We also examine the robustness of our approach to inaccurate user navigation prediction, transient NR link loss, dynamic LTE bandwidth variations, and diverse 360° video content. Finally, we contrast our results over five popular video quality metrics. The paper makes a community contribution by publicly sharing a dataset that captures the rate-quality trade-offs of the 360° video content used in our evaluation, for multiple contemporary quality metrics, to stimulate further studies and follow up work.**

***Index Terms*—360° video, 5G networks, cooperative edge-client computation, dual connectivity transmission, geometric programming, live streaming, mobile VR systems, rate-distortion analysis and optimization, scalable 360° tiling.**

Manuscript received 13 October 2023; revised 28 January 2024; accepted 16 March 2024. Date of publication 25 April 2024; date of current version 21 August 2024. This work was supported in part by National Science Foundation (NSF) under Award CCF-2031881, Award ECCS-2032387, Award CNS-2040088, Award CNS-2032033, and Award CNS-2106150, in part by the National Institutes of Health (NIH) under Award R01EY030470, and in part by the Panasonic Chair of Sustainability at the New Jersey Institute for Technology. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Zhi Liu. *(Corresponding author: Jacob Chakareski.)*

Jacob Chakareski is with the New Jersey Institute of Technology, Newark, NJ 07102 USA (e-mail: jacobcha@njit.edu).

Mahmudur Khan was with the New Jersey Institute of Technology, Newark, NJ 07102 USA. He is now with the York College of Pennsylvania, York, PA 17403 USA (e-mail: mkhan17@ycp.edu).

Digital Object Identifier 10.1109/TMM.2024.3382910

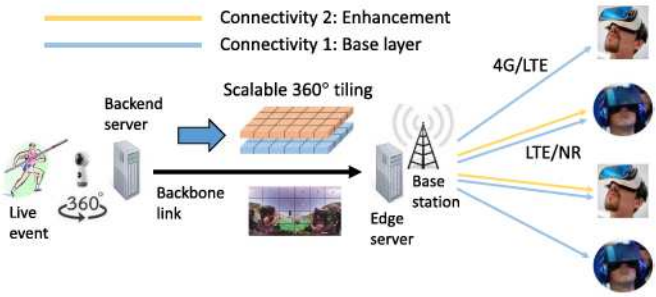

Fig. 1. System setting: Compressed live 360° video content is sent over a backbone link through an edge server to heterogenous mobile VR clients.

I. Introduction

360° video streaming to virtual reality (VR) headsets has applications in education and training, telepresence and telecommuting, healthcare, environmental monitoring, and first responders [1], [2]. Societal changes introduced due to the recent pandemic, e.g., online classes, work from home, and telemedicine, highlight the importance of remote 360° video immersion/communication. Compared to traditional streaming [3], VR-based 360° video streaming introduces further challenges, particularly in the mobile domain, as it requires very high transmission rate, hyper intensive computation, and very low latency [4]. The emergence of 5G networks offers possibilities to address these challenges, as they enable much higher rates and lower latencies, while providing edge-based computation capabilities, relative to existing wireless systems.

In this paper, we explore live 360° video streaming to heterogeneous mobile VR clients in 5G networks. The client population comprises devices that feature single 4 G/LTE or dual LTE/NR (New Radio) cellular connectivity [5]. The captured content is compressed at the origin using scalable 360° tiling and sent over a backbone link to an edge server collocated with the base station serving the clients. The server provides computational assistance, while adapting the incoming data to the individual clients' down-link transmission rates and computing capabilities, using formal optimization. The system setting we study is shown in Fig. 1.

Mobile clients featuring single connectivity are served by the edge server a base layer representation of the content adapted to their down-link transmission capacity and device computing capability, such that their 360° viewport quality is maximized. For enhanced quality of experience, a dual connectivity client is served in parallel a base layer representation of the content on its LTE connectivity and a complementary viewport-specific enhancement layer representation on its NR connectivity,

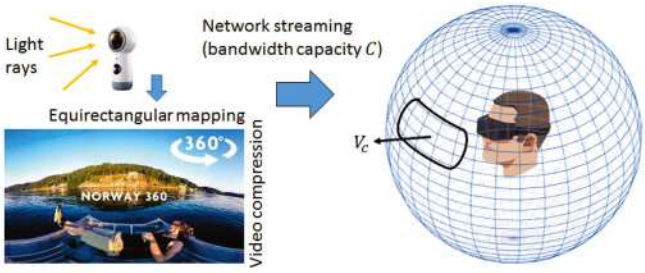

Fig. 2. 360° video capture and streaming, and user viewport V_c.

synergistically adapted to the respective down-links' transmission capacities and its computing capability, such that its 360° viewport quality stemming from the two integrated content layers is maximized and higher.

We formulate two optimization problems to guide the operation of the edge server in each case, accounting for key system components of the delivery process and the induced end-to-end latency, and aiming to maximize the immersion fidelity experienced by each client. We explore respective geometric programming optimization strategies that compute the optimal solutions at lower complexity. The problem analysis, formulation, and optimization build upon our earlier advances that include rate-distortion modeling of compressed 360° tiles, design of scalable 360° tiling, and statistical analysis of a client's 360° navigation actions [6]. We demonstrate considerable performance gains over multiple assessment factors over the state-of-the-art and examine the robustness of our approach to inaccurate user navigation prediction, transient NR link loss, dynamic LTE bandwidth variations, and diverse 360° video content. We contrast our results over five popular video quality metrics, and demonstrate consistent performance benefits.

We advance the state-of-the-art with the following contributions. (i) We explore jointly optimizing the allocation of communication and computing resources of the edge server and VR users' mobile devices. (ii) We introduce and explore for the first time dual-connectivity 5G NR links for enhanced live streaming of 360° videos. (iii) We pursue a rigorous analysis of the fundamental trade-offs between computation and communication that arise in this context. Building upon it, we formulate effective rate-distortion-computation optimization techniques to identify key system resource allocation decisions and maximize the delivered viewport quality across the mobile VR client population. (iv) We rigorously analyze the computational complexity of the two optimization algorithms we formulate. (v) We carry out a comprehensive performance evaluation, addressing a broad set of real-world conditions and scenarios, and demonstrate considerable benefits relative to state-of-the-art reference methods, over five popular video quality metrics (PSNR, WS-PSNR, SSIM, MSE, VMAF).

We make a community contribution by publicly sharing a dataset that captures the rate-quality trade-offs of the 360° video content used in our evaluation, for multiple contemporary quality metrics, to stimulate further studies and follow up work.

II. Background and Related Work

360° video is a recent video format captured by an omnidirectional camera that records light rays incoming from every direction (see Fig. 2, top left). It enables a 3D 360° look-around of the surrounding scene for a remote user, virtually placed at the camera location, on his VR headset (see Fig. 2, right). After capture, the raw spherical video frames are first mapped to an equirectangular panorama (see Fig. 2, bottom left) and then compressed planar video compression, e.g., HEVC. The former intermediate step is introduced, as compression techniques operating directly on spherical data are much less mature and performing relative to traditional video compression operating on 2D video frames.

Streaming services typically deliver the entire monolithic 360° panorama to a user, using standards-based methods, e.g., MPEG-DASH [7]. This negatively impacts the quality of experience, as huge volume of data needs to be sent, typically exceeding by a wide margin the available network bandwidth C. Moreover, the user only looks at a small portion of the spatial panorama, denoted as viewport (see Fig. 2, right). Hence, only lower-quality and lower-resolution 360° videos are delivered online presently. Yet, as traditional server-client Internet architectures are used, the desired low-latency interactivity is also impacted. Finally, the operating conditions are even more challenging in a wireless setting, due to the more constrained communication and computation resources.

The above challenges motivate novel communications systems that holistically integrate the 5G capabilities of edge computing and high rate dual connectivity transmission, with viewport-adaptive scalable 360° streaming, to enable novel high-quality mobile VR applications. This is our goal here.

Similarly to our approach, several studies of single-user on-demand 360° Internet streaming [8], [9], [10] have considered splitting the 360° video content into spatial tiles. We advance these studies by: (i) considering a different and more challenging system setting; (ii) pursuing a rigorous analysis of the fundamental trade-offs between viewport quality, user navigation patterns, coding efficiency, and available system resources; and (iii) enabling much lower latency via holistic integration of edge-based delivery, scalability, and 360° tiling.

Other more broadly related studies include [11], which pursues the design of a view synthesis-based 360° VR caching system over a cloud radio access network, where both mobile edge computing (MEC) and hierarchical caching are supported, to address network bandwidth limitations. Similarly, [12] explores edge cache-assisted 360° live video streaming, to increase the overall quality of the delivered content to users. Ref. [13] investigates reinforcement learning based rate adaptation with adaptive view prediction and content tiling for 360° video streaming, to improve the quality of experience delivered to users. Ref. [14] presents a field-of-view tracking enabled VR streaming system for mobile platforms.

User collaboration in live 360-degree video delivery for accurate field of view (FoV) prediction via an LSTM neural network, coupled with an optimal rate allocation strategy, is explored in [15]. Similarly, deep learning based FoV prediction for proactive adaptive 360-degree video streaming in mobile networks is explored in [16]. Lastly, MEC-based 360° video VR streaming with field-of-view (FoV) prediction, together with joint video coding, proactive caching, computation offloading, and data transmission, is studied in [17].

Multi-path transmission as known as network path diversity has been studied for Internet video delivery in sender-driven and receiver-driven settings. The study in [18] explored the rate-distortion trade-offs of multi-path content delivery in both settings and formulated optimization techniques to address each

case. The study in [19] considered sender-driven path diversity for video communication using unbalanced multiple description coding, to accommodate the prospective heterogeneity in network bandwidth across the employed paths. The trade-offs of video streaming over multiple paths using layered coding or multiple descriptions have been investigated in [20]. Distributed rate-distortion optimized sender-driven multi-path video delivery from multiple servers has been explored in [3]. Most recently, parallel 360° video transmission over multiple wireless technologies has been considered for indoor on-demand mobile VR streaming in [21], [22].

Finally, recent studies examined live 360° video multicast in cellular networks. In [23], a method denoted as $VRCast$ is proposed that uses a grouping algorithm to optimally separate mobile users into multicast groups and allocates radio resources among them. We have implemented VRCast as a state-of-the-art reference method in our experimental evaluation. In [24], a joint unicast-multicast streaming system over a cellular network is proposed that optimizes the resource allocation for a group of users to enhance their experience. The system setting and scope of work of these studies are simpler and narrower relative to ours, as they consider 4G/LTE networks and focus on radio resource allocation optimization only.

Our paper is motivated by a shorter study we carried out in [25] that examined transmission of pre-stored 360° content over parallel WiFi and millimeter wave links, in an indoor setting. The major technical differences of the present paper include the investigation of a different scenario, live 360° layered multicast streaming to heterogeneous mobile VR clients in an outdoor setting, and the additional specific challenges it introduces, the synergistic integration of 5G edge computing and dual-connectivity transmission with scalable 360° tiling and streaming, a more accurate analysis of the end-to-end system latency that integrates the impact of both content decoding and viewport rendering, and the formulation of rigorous rate-distortion-computing optimization to identify key system resource allocation decisions and trade-offs. Moreover, we carry out performance evaluation and contrast our results over two popular immersion video quality metrics.

III. 360° Content Coding and User Navigation

A. Content and Navigation Modeling

The main problem analysis and optimization formulated in this paper build upon modeling advances that we introduced earlier. For completeness, we outline these advances here and refer the reader to [6] for further details, to conserve space.

We leverage tiling of the 360° video frames, as illustrated in Fig. 3, to capture effectively the user viewport over time and exploit the uneven rate-distortion trade-offs that arise across the spatial 360° panorama. We denote as GOP tile (i, j) the collection of tiles across the video frames comprising a Group of Pictures (GOP), at the same spatial location (i, j) in the tiling. Each GOP tile is independently encoded and streamed to the user, according to our analysis and optimization. The rate-distortion trade-off of compressing a GOP tile (i, j) can be accurately captured with a power law function $D_{ij}(R_{ij}) = aR_{ij}^b$, where (a, b) are model fit parameters.

The spatial 360° tiling also facilitates developing a statistical model of user navigation, where each tile (i, j) can be assigned

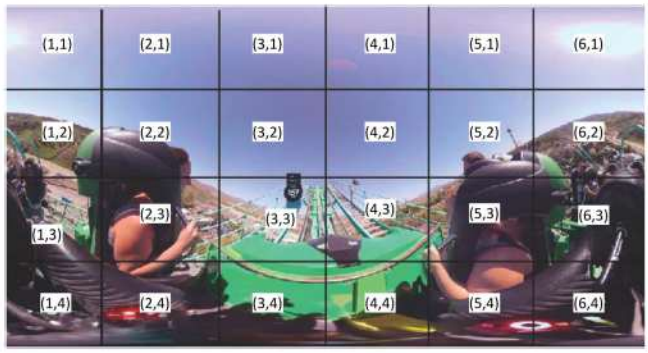

Fig. 3. An example 6×4 spatial tiling of a 360° video frame.

a navigation probability P_{ij} that captures how often the tile overlaps with the viewport over the duration of a GOP. Later, we will integrate all these modeling advances into our analysis and optimization to dynamically select the amount of resources allocated over space and time, for a 360° video as it is being streamed to a user. We compress the GOP tiles into multiple scalable layers to enable an effective implementation of our analysis and optimization.

B. Navigation Modeling Challenges for Live Content

In our setting (live 360° streaming), complete navigation traces and probabilities will not be available in advance. The edge server will need to develop them incrementally during a session. Carrying out this task has been studied before[1] and is beyond the scope of the present paper. Still, we will effectively explore its impact on our analysis and optimization by considering inaccurate navigation likelihoods $\hat{P}_{ij}$ in our evaluation. Concretely, we will adapt the degree of inaccuracy ΔP of the navigation probability distribution $\{\hat{P}_{ij}\}$ relative to the true distribution and examine the reliability or drop in performance of our framework as ΔP is increased.

IV. Edge Server and Mobile Client Modeling

A. Outline

We consider a set of mobile clients U served by a 5G edge server capable of communicating using dual (LTE/NR) cellular connectivity. The server is also equipped with high-end Graphics Processing Units (GPUs) to assist the mobile VR clients in their computing needs to process (decode, render) the highly demanding 360° content delivered to them. Each client features either a single or dual connectivity mobile device linked to a VR headset. Let U_1 and U_2 denote respectively the subsets of single and dual connectivity clients. Using navigation information uplinked by the mobile VR clients, the edge server is able to identify the present 360° viewport of each client and optimize the delivery of the content to the client using the analysis and optimization we describe in Sections V and VI. To single connectivity clients, the server streams a lower-quality (baseline) representation of the 360° content optimized for their downlink and device computing capabilities. To each dual-connectivity

[1] Sample approaches include regression-based prediction, static pyramid-based prediction, and neural network-based prediction, all using navigation history to estimate present actions/likelihoods [26], [27], [28], [29].

TABLE I
KEY NOTATION

Parameteres	Definitions
Set of mobile clients	U
GOP tile	m
Set of all GOP tiles	M
Number of layers in baseline representation	n_b
Number of layers in enhancement representation	n_e
Subset of tiles in user viewport	M_u
Baseline representation encoding rate	R^1_{ij}
Enhancement representation encoding rate	R^2_{ij}
Subset of tiles decoded at server	M^r_u
Temporal playback duration of a GOP	ΔT
Data size of decompressed GOP tile	E_r
Viewport data size after decoding	E_v
Minimum GOP tile encoding rate	$R_{ij,min}$
Maximum GOP tile encoding rate	$R_{ij,max}$
LTE link capacity	C_1
5G NR link capacity	C_2
Server decoding speed	Z_u
Mobile device decoding speed	z_u
Mobile device processing power	r_u
Computed data size per GPU cycle	b_h
Transmission latency of baseline representation	τ^1_u
Transmission latency of enhancement representation	τ^2_u
Decoding delay at server	$\tau_{u,\mathrm{ds}}$
Decoding delay at client mobile device	$\tau_{u,\mathrm{dc}}$
Rendering delay at client mobile device	$\tau_{u,\mathrm{r}}$
Expected viewport distortion for a client	D_u
Navigation likelihood of a GOP tile	P_{ij}
Tile-based WS-MSE weight	w_{ij}

client, the server streams in addition a viewport-specific enhancement representation of the content over the dual NR down link in parallel, benefiting from the enhanced computing and transmission capabilities of the edge server and the NR link, to optimally construct and stream the enhancement representation. The client then integrates synergistically the two representations to produce a higher quality viewport for its user. We provide a detailed description of our edge server and client modeling next. We list the key notations used in the paper in Table I.

B. LTE/NR 5G Dual Connectivity Model

The 5G deployment model we consider has been standardized and has been in use predominantly. It is referred to as non-standalone (NSA). In this deployment method, a cellular device is able to connect simultaneously to collocated 4 G/LTE and NR base stations (a technique known as dual connectivity), served by the same extended packet core network [5].

The NSA model is most suitable for providing enhanced mobile broadband services since NR can act as a capacity overlay to the 4 G/LTE network, supplementing existing network investments.

C. Edge Server Modeling

1) Back-End Server Operation: The remote server initiates the live 360° video streaming session and establishes a backbone link with the edge server. The back-end server encodes the captured content into scalable 360° tiling for every GOP and transmits the compressed data to the edge server, where it is cached for follow-up streaming to mobile clients. When a mobile client makes a request to the back-end server to join the live session, its request is redirected to the edge server, who then initiates the streaming of the content from the present GOP onwards to the client.

2) Baseline and Enhancement Representation Construction: We denote with m_{ij} the GOP tile at spatial location (i, j) and with M the set of all GOP tiles. The edge server constructs a base layer representation of the present GOP of the 360° content, for streaming to a client over its LTE link, by combining the first $n_b(i, j)$ embedded layers for each tile $m_{ij} \in M$. Let R^1_{ij} denote the data rate of the base layer representation for tile m_{ij}. The server selects R^1_{ij} and thus $n_b(i, j)$ according to the analysis and optimization presented later, distinguishing between a single or dual connectivity client in this case. Similarly, the server constructs an enhancement representation, for streaming only to a dual-connectivity client $u \in U_2$ over its NR link, by combining the subsequent $n_e(i, j)$ embedded layers for each tile m_{ij} comprising the client's viewport. Let R^2_{ij} denote the data rate of the enhancement representation for tile $m_{ij} \in M_u$, where $M_u \subset M$ denotes the subset of tiles encompassing the user viewport of client u. The server selects R^2_{ij} and thus $n_e(i, j)$ according to the analysis and optimization presented later, jointly with R^1_{ij} for each tile m_{ij}. A GOP tile m_{ij} is reconstructed at highest quality when all its L layers from the scalable 360° tiling are combined. Let $R_{ij,\max}$ denote the data rate of tile m_{ij} in that case. The tile is reconstructed at lowest quality when only its first embedded layer is used and the associated data rate of the compressed tile in that case is denoted as $R_{ij,\min}$.

3) Edge-Based Tile Decoding for Raw Data Transmission: To explore broader performance trade-offs, we consider transmission of raw (decoded) 360° data for clients in U_2. Using navigation information, the edge server identifies the present viewport of client $u \in U_2$ comprising a subset of GOP tiles M_u. A portion of these tiles $M^{\mathrm{r}}_u \subset M_u$ are decoded by the server. Each such tile m_{ij} is decoded/reconstructed at its highest quality from its highest available data rate $R_{ij,\max}$. Let Z_u denote the maximum decoding speed of the server assigned to client u. Thus, we can formulate the time delay induced to decode/reconstruct the tiles comprising M^{r}_u as:

$$\tau^2_{u,\mathrm{ds}} = \frac{\sum_{(i,j)\in M^{\mathrm{r}}_u} R_{ij,\max}\Delta T}{Z_u},$$

where ΔT denotes the temporal playback duration of a GOP. Let E_{r} denote the decoded data size of GOP tile $m_{ij} \in M^{\mathrm{r}}_u$.

4) Baseline Representation Streaming Over an LTE Link: The server streams a baseline representation of the 360° content to a client $u \in U_1$ over its LTE link. Let C_1 denote the transmission capacity of the link. The transmission latency of sending the baseline representation can be formulated as:

$$\tau^1_u = \frac{\sum_{(i,j)} R^1_{ij}\Delta T}{C_1}.$$

5) Enhancement Representation Streaming Over an NR Link: The server streams in parallel an enhancement representation of the 360° content to every client $u \in U_2$ over its NR link. It comprises $|M^{\mathrm{r}}_u|$ (complete) raw GOP tiles and enhancement representations for the remaining compressed viewport tiles $m_{ij} \in M^{\mathrm{e}}_u = M_u \setminus M^{\mathrm{r}}_u$. Let C_2 denote the transmission capacity of the link. Then, we can formulate the transmission latency

of sending the enhancement representation as:

$$\tau_u^2 = \frac{|M_u^{\mathrm{r}}|E_{\mathrm{r}} + \sum_{(i,j)\in M_u^{\mathrm{e}}} R_{ij}^2 \Delta T}{C_2}.$$

For ease of notation, we use single symbols C_1 and C_2 to indicate the transmission capacities of an LTE link and an NR link, respectively. Still, our analysis is general and allows for C_1 to be uniquely associated with a given single-connectivity client $u \in U_1$, in terms of value and spatiotemporal dynamics, due to user mobility and network interference. The same applies to the pair $\{C_1, C_2\}$ associated with a wireless client $u \in U_2$ equipped with dual connectivity. Moreover, as our analysis is applied per GOP (a short time segment of the content), it allows for considerable flexibility and effectiveness in carrying out client adaptation by the edge server.

D. Mobile Client Modeling

1) 360° Content Decoding and Rendering: A client's mobile device is equipped with a mobile GPU for decompressing and rendering the received 360° video for display on the client's VR headset. The maximum decoding speed of the device is z_u. In the case of single connectivity clients, z_u is exclusively assigned to decode the received baseline representation. Thus, we can formulate the time delay of decoding the baseline representation in this case as:

$$\tau_{u,\mathrm{dc}}^1 = \frac{\sum_{(i,j)} R_{ij}^1 \Delta T}{z_u}.$$

In the case of dual connectivity clients, the decoding capabilities of the device are shared to decompress the received baseline representation and the compressed portion (tiles) of the received enhancement representation. Let z_u^1 and z_u^2 denote the decoding speeds allocated by the device for decompressing the baseline representation and enhancement representation, respectively. It should hold that $z_u^1 + z_u^2 \leq z_u$. We can formulate the time delay of decoding each representation equivalently to the expression above:

$$\tau_{u,\mathrm{dc}}^{2,1} = \frac{\sum_{(i,j)} R_{ij}^1 \Delta T}{z_u^1}, \quad \tau_{u,\mathrm{dc}}^{2,2} = \frac{\sum_{(i,j)\in M_u^{\mathrm{e}}} R_{ij}^2 \Delta T}{z_u^2}.$$

Similarly, let r_u denote the processing power of the mobile device for rendering the user viewport. In the case of single connectivity clients, r_u is exclusively assigned to render the viewport from the decoded baseline representation. Thus, we can formulate the time delay of rendering the viewport in this case as $\tau_{u,\mathrm{r}}^1 = \frac{E_v}{r_u b_h}$, where E_v denotes the data size of the viewport after decoding and b_h denotes the computed data size per GPU cycle on the device, for this activity.

In the case of dual connectivity clients, the rendering capabilities of the device are shared to render the viewport at baseline quality first, from the decoded baseline representation, and then at enhanced quality, from the enhancement representation decoded jointly with the baseline representation. Let r_u^1 and r_u^2 denote the rendering power allocated by the device to render the baseline quality viewport and enhanced quality viewport, respectively. It should hold that $r_u^1 + r_u^2 \leq r_u$. We can formulate the time delay of rendering each viewport equivalently to the expression above: $\tau_{u,\mathrm{r}}^{2,i} = \frac{E_v}{r_u^i b_h}$, for $i = 1, 2$.

2) Expected Viewport Distortion Formulation: Let P_{ij}^u denote the navigation likelihood of GOP tile $m_{ij} \in M_u$ for user u. Similarly, let w_{ij} denote a tile-based WS-MSE weight that captures the normalized surface area on the 360° view sphere covered by tile m_{ij}. We formally design w_{ij} as follows. Let the size of the 360° panorama be $X \times Y$ pixels. Following [30], we first define a spherical pixel weight $w(x, y)$ for location (x, y) in the panorama as $w(x, y) = \cos(\frac{(y+0.5-Y/2)\pi}{Y})$. We then formulate the target weights as $w_{ij} = \sum_{xy\in m_{ij}} w(x, y)/W_{\mathrm{n}}, \forall (i, j) \in M_u$, where $W_{\mathrm{n}} = \sum_{xy} w(x, y)$ is a normalizing factor. For a single connectivity client $u \in U_1$, we formulate its expected viewport distortion over that GOP as:

$$\begin{aligned} D_u^1\left(\{R_{ij}^1\}\right) &= \sum_{(i,j)\in M_u} w_{ij} P_{ij}^u D_{ij}\left(R_{ij}^1\right), \\ &= \sum_{(i,j)\in M_u} w_{ij} P_{ij}^u a_{ij} \left(R_{ij}^1\right)^{b_{ij}}. \end{aligned}$$

Similarly, for a dual connectivity client $u \in U_2$, we formulate its expected viewport distortion over that GOP as:

$$\begin{aligned} D_u^2\left(\{R_{ij}^1, R_{ij}^2\}\right) &= \sum_{(i,j)\in M_u} w_{ij} P_{ij}^u D_{ij}\left(R_{ij}^1, R_{ij}^2\right), \\ &= \sum_{ij\in M_u^{\mathrm{r}}} w_{ij} P_{ij}^u a_{ij} \left(R_{ij,\max}\right)^{b_{ij}} \\ &\quad + \sum_{ij\in M_u^{\mathrm{e}}} w_{ij} P_{ij}^u a_{ij} \left(R_{ij}^1 + R_{ij}^2\right)^{b_{ij}}. \end{aligned}$$

V. Minimizing the Viewport Distortion

A. Outline

We formulate two optimization problems that aim to minimize the expected viewport distortion experienced during a GOP by single-connectivity and dual-connectivity clients, respectively. Because they are applied on a GOP basis (a short time segment of the content), they enable considerable effectiveness in maximizing the immersive quality of experience delivered to a client dynamically. The optimization problems we pursue integrate key system constraints in each case.

We recall that our key notation is listed in Table I and many of the symbols used going forward have been defined earlier.

B. Single-Connectivity Clients

Our objective is to minimize the expected viewport distortion for a client $u \in U_1$, given its LTE down link transmission capacity and mobile device decoding and rendering computing capabilities, and a system latency constraint. We formulate this optimization problem of interest as:

$$\min_{\{R_{ij}^1\}} D_u^1\left(\{R_{ij}^1\}\right), \tag{1}$$

$$\text{s.t.:} \quad \tau_u^1 + \tau_{u,\mathrm{dc}}^1 + \tau_{u,\mathrm{r}}^1 \leq \Delta T, \tag{2}$$

$$R_{ij,\min} \leq R_{ij}^1 \leq R_{ij,\max}, \forall (i, j) \in M_u, \tag{3}$$

Constraint (2) captures the system latency requirement that imposes that the aggregate time consumed to stream the baseline representation, decode it on the mobile device, and render the client's viewport on the device must not exceed ΔT. Constraint (3) requires that the selected transmission rates R^1_{ij} for the tiles $m_{ij} \in M_u$ must lie within the range of data rates $[R_{ij,\min}, R_{ij,\max}]$ enabled by the scalable 360° tiling. We formulate an optimization method to solve the problem captured by (1)–(3) in Section VI-A.

C. Dual-Connectivity Clients

We aim to minimize the expected viewport distortion for a client $u \in U_2$, given its LTE and NR down link transmission capacities and mobile device decoding and rendering computing capability, the edge server's decoding computing capability assigned to client u, and two system latency constraints. We formulate this optimization problem of interest as:

$$\min_{\substack{\{R^1_{ij}\},\{R^2_{ij}\},\\ M^{\mathrm{r}}_u,\{z^i_u\},\{r^i_u\}}} D^2_u\left(\{R^1_{ij}, R^2_{ij}\}\right), \tag{4}$$

$$\text{s.t.:} \quad \tau^1_u + \tau^{2,1}_{u,\mathrm{dc}} + \tau^{2,1}_{u,\mathrm{r}} \leq \Delta T, \tag{5}$$

$$\tau^2_{u,\mathrm{ds}} + \tau^2_u + \tau^{2,2}_{u,\mathrm{dc}} + \tau^{2,2}_{u,\mathrm{r}} \leq \Delta T, \tag{6}$$

$$R_{ij,\min} \leq R^1_{ij} \leq R_{ij,\max}, \forall (i,j) \in M_u, \tag{7}$$

$$0 \leq R^2_{ij} \leq R_{ij,\max} - R^1_{ij}, \forall (i,j) \in M_u, \tag{8}$$

$$r^1_u + r^2_u \leq r_u,\ z^1_u + z^2_u \leq z_u, \tag{9}$$

Constraint (5) captures the system latency requirement associated with the transmission of the baseline representation over the LTE down link. It imposes that the aggregate time consumed to stream the baseline representation, decode it on the mobile device, and render the client's viewport at baseline quality on the device must not exceed ΔT. Constraint (6) captures the system latency requirement associated with the transmission of the enhancement representation over the NR down link. It imposes that the aggregate time consumed to decode the tiles $m_{ij} \in M^{\mathrm{r}}_u$ at the edge server, stream the enhancement representation, decode the remaining tiles $m_{ij} \in M^{\mathrm{e}}_u$ at the client on the mobile device, and render the client's viewport at enhanced quality on the device must not exceed ΔT. Constraint (7) requires that the selected transmission rates R^1_{ij} for the tiles $m_{ij} \in M_u$ must lie within the range of data rates $[R_{ij,\min}, R_{ij,\max}]$ enabled by the scalable 360° tiling. Similarly, constraint (8) requires that the selected transmission rates R^2_{ij} for the tiles $m_{ij} \in M^{\mathrm{e}}_u$ must lie within the range of data rates $[0, R_{ij,\max} - R^1_{ij}]$. Finally, the constraints in (9) capture the decoding speed and rendering power limitations of the client's mobile device.

We note that the transmission latency constraints (5) and (6) are stricter than and imply the respective transmission capacity constraints on the two wireless links, i.e., $\sum_{ij} R^1_{ij} \leq C_1$ and $\sum_{ij} R^2_{ij} \leq C_2$. The same applies to (2) in Section V-B.

Equations (4)–(9) represents a mixed-integer programming problem [31], which is hard to solve optimally in practice. The optimal solution can be achieved via an exhaustive search, which requires searching over all the possible options for the set M^{r}_u, and for every such option, finding the optimal streaming rates $\{R^1_{ij}\}$ and $\{R^2_{ij}\}$, and the client's decoding speed and rendering capability allocations $\{z^i_u\}$ and $\{r^i_u\}$. Instead, we explore a lower complexity approach in Section VI-B.

Algorithm 1: Single Connectivity Optimal Rate Allocation

1: Initialize $t = 0, R^1_{ij}(t) \in [R_{ij,\min}, R_{ij,\max}]$.
2: **while** true **do**
3: $t = t + 1$
4: Determine the optimum $\{R^1_{ij}(t)\}, D^1_u(t)$ by solving (10) using GGPLAB.
5: **if** $|D^1_u(t) - D^1_u(t-1)| \leq \epsilon$ **then**
6: Break
7: **end if**
8: **end while**
9: $D^{1\star}_u = D^1_u(t)$
10: $\{R^{1\star}_{ij}\} = \{R^1_{ij}(t)\}$

VI. Computing the Optimal Resource Allocation

A. Single Connectivity Clients

We note that (1)–(3) represents a Geometric Programming (GP) problem, as the objective function and constraints represent posynomial functions [32]. We formulate an iterative method to solve this problem optimally following [33]. In particular, at each iteration t, we define $D^1_u(t) := D^1_u(\{R^1_{ij}(t)\})$. Then, the optimization problem to be solved at iteration t is:

$$\min_{\{R^1_{ij}(t)\}} D^1_u(t),$$

$$\text{s.t.:} \quad \text{(2) and (3).} \tag{10}$$

We carry out the optimization above iteratively until $|D^1_u(t) - D^1_u(t-1)| \leq \epsilon$, for some small $\epsilon \geq 0$. When this condition is met, we obtain the optimal value of the optimization problem in (10) as $D^{1*}_u = D^1_u(t)$ and the corresponding optimal streaming rate selections as $\{R^{1*}_{ij}\} = \{R^1_{ij}(t)\}$.

A formal description of our proposed iterative optimization method for solving (1)–(3) and computing the optimal baseline representation streaming data rates $\{R^{1*}_{ij}\}$ is provided in Algorithm 1. In Line 4, we numerically implemented the geometric programming solution to (10) using the Matlab GP toolbox GGPLAB [34]. We activate the algorithm every ΔT periods of time, at the onset of each subsequent GOP. Algorithm 1 provably converges to the global optimal solution [33].

We assess the computational complexity of Algorithm 1 as follows. The number of required iterations is $\frac{\log(\mathcal{K}_1)/t_{0,1}\epsilon}{\log \xi}$, where $\mathcal{K}_1$ is the number of constraints in (10), $t_{0,1}$ is the initial point to approximate the accuracy of the interior point method used to solve (10), $0 < \epsilon < 1$ is the stopping criterion for the interior point method, and ξ is used for updating the accuracy of the interior point method [32]. Thus, the total number of computations for Algorithm 1 is on the order of $\frac{\log(\mathcal{K}_1)/t_{0,1}\epsilon}{\log \xi}$. For streaming the 360° content *Runner*, the number of iterations for Algorithm 1 is 2 when $\epsilon = 0.0001$ and it is 1 when $\epsilon = 0.001$, for different values of Z_u (1.5 Gbps to 6 Gbps), z_u (100 Mbps to 500 Mbps), and r_u (1.88 GPixels/s to 28.2 GPixels/s). We can observe that Algorithm 1 is quite robust to the choice of the optimization

threshold ϵ over a wide range of values, for diverse values of the key system parameters under analysis.

B. Clients With Dual Connectivity

1) Outline: We first present an outline of our optimization method here. To address the high-complexity nature of the mixed-integer programming problem (4)–(9), we first intelligently construct a small subset of prospective options for the set M_u^{r} to search over that captures the optimal solution with high likelihood. We then compute the optimal solution to (4)–(9) and the associated optimal value of the objective function $D_u^{2*}(\{R_{ij}^{1*}, R_{ij}^{2*}\})$ in (4), for every such (fixed) candidate M_u^{r}. Finally, we select the optimal solution and respective candidate set M_u^{r} that lead to the smallest value of $D_u^{2*}(\{R_{ij}^{1*}, R_{ij}^{2*}\})$.

2) Optimization Method Formulation: We start by sorting the GOP tiles $m_{ij} \in M_u$ in descending order of their products $w_{ij}P_{ij}^u$. Let M_u^{s} denote the thereby obtained set of sorted tiles. We then construct M_u^{r} such that it comprises the first k tiles from M_u^{s}, for $k = 0, \ldots, |M_u|$. We note that the set M_u^{r} will be empty ($\varnothing$) for the case $k = 0$. Then, all enhancement representation tiles $m_{ij} \in M_u$ will be transmitted as compressed data over the NR down link, and each tile will comprise $n_e(i,j)$ embedded enhancement layers from the scalable 360° tiling, as introduced in Section IV-C.

Let $M_{u,k}^{\mathrm{r}}$ denote the candidate set constructed for a given value of k. Similarly, let $D_{u,k}^2(\{R_{ij}^1, R_{ij}^2\})$ denote the objective function in (4) for that $M_{u,k}^{\mathrm{r}}$ and k. Given these developments, we can rewrite the problem (4)–(9) as

$$\min_{\substack{\{R_{ij}^1\},\{R_{ij}^2\},\\ \{z_u^i\},\{r_u^i\}}} D_{u,k}^2\left(\{R_{ij}^1, R_{ij}^2\}\right),$$

$$\text{s.t.:} \quad (5), (6), (7), (8), \text{ and } (9). \tag{11}$$

The above problem can be solved optimally by converting it to GP. To do so, we first introduce an auxiliary variable $R_{ij}^{12} = R_{ij}^1 + R_{ij}^2$, for $(i,j) \in M_{u,k}^e$. Then, we reformulate accordingly the transmission latency and client decoding latency associated with the enhancement representation:

$$\tau_u^{2\prime} = \frac{|M_{u,k}^{\mathrm{r}}|E_{\mathrm{r}} + \sum_{(i,j)\in M_{u,k}^{\mathrm{e}}}(R_{ij}^{12} - R_{ij}^1)\Delta T}{C_2},$$

$$\tau_{u,\mathrm{dc}}^{2,2\,\prime} = \frac{\sum_{(i,j)\in M_{u,k}^{\mathrm{e}}}(R_{ij}^{12} - R_{ij}^1)\Delta T}{z_u^2}.$$

Next, we reformulate the objective in (11) such that it captures only the impact of tiles m_{ij} sent as compressed data as part of the enhancement representation, i.e., tiles $m_{ij} \in M_{u,k}^{\mathrm{e}}$, for $M_{u,k}^{\mathrm{e}} = M_u \setminus M_{u,k}^{\mathrm{r}}$. We formally define the reformulated objective function as:

$$D_{u,k}^{2,\mathrm{e}}\left(R_{ij}^{12}\right) := \sum_{ij\in M_{u,k}^{\mathrm{e}}} w_{ij}P_{ij}^u a_{ij}\left(R_{ij}^{12}\right)^{b_{ij}}.$$

Finally, given the above advances, we reformulate the optimization problem (11) as:

$$\min_{\substack{\{R_{ij}^{12}\},\\ \{z_u^i\},\{r_u^i\}}} D_{u,k}^{2,\mathrm{e}}\left(R_{ij}^{12}\right), \tag{12}$$

$$\text{s.t.:} \quad (5) \text{ and } (9), \tag{13}$$

$$\tau_{u,\mathrm{ds}}^2 + \tau_u^{2\prime} + \tau_{u,\mathrm{dc}}^{2,2\,\prime} + \tau_{u,\mathrm{r}}^{2,2} \le \Delta T, \tag{14}$$

$$R_{ij}^1 \le R_{ij}^{12} \le R_{ij,\max}, \forall (i,j) \in M_u. \tag{15}$$

We convert the problem (12)–(15) to GP using the single condensation method [33], to efficiently solve it. We will then integrate this solution as a key step in our efficient lower complexity iterative optimization procedure outlined earlier, to solve the entire problem of interest (4)–(9). The design of the latter procedure will be described in detail subsequently.

We set the rates $\{R_{ij}^1\}$ as free parameters in (12)–(15), to enable the conversion to GP. We have empirically established that setting $R_{ij}^1 = R_{ij,\min}, \forall (i,j) \in M_u$, leads to the best performance, as it maximizes the likelihood of the baseline representation being streamed, decoded, and rendered at the client within ΔT. Simultaneously, this enables allocating more of the client's computing resources to decode and render key enhancement representation GOP tiles at higher quality, which in turn will augment the client's viewport quality. Following the condensation method, for a constraint which is a ratio of posynomial functions, the denominator posynomial can be approximated into a monomial. We formulate an iterative method to solve first (12) optimally. At each iteration t, we convert the constraint (14) into a posynomial as follows:

$$\left(\frac{\Delta T Z_u C_2 z_u^2 r_u^2 b_h}{\delta_1(t)}\right)^{-\delta_1(t)} \cdot \left(\frac{\Delta T Z_u z_u^2 r_u^2 b_h \sum_{ij\in M_{u,k}^e} R_{ij}^1}{\delta_2(t)}\right)^{-\delta_2(t)} \cdot \left(\frac{\Delta T Z_u C_2 r_u^2 b_h \sum_{ij\in M_{u,k}^e} R_{ij}^1}{\delta_2(t)}\right)^{-\delta_3(t)} \cdot \left(C_2 z_u^2 r_u^2 b_h \Delta T \sum_{ij\in M_{u,k}^r} R_{ij,max} + Z_u z_u^2 r_u^2 b_h \left(|M_{u,k}^r|E_r + \Delta T \sum_{ij\in M_{u,k}^e} R_{ij}^{12}(t)\right) + Z_u C_u^2 r_u^2 b_h \Delta T \sum_{ij\in M_{u,k}^e} R_{ij}^{12}(t) + Z_u C_u^2 z_u^2 E_v\right) \le 1, \tag{16}$$

where $\delta_i(t) = c_i/c_{\mathrm{s}}$, for $i = 1, 2, 3$ and $c_{\mathrm{s}} = \sum_{i=1}^3 c_i$, and

$$c_1 = \Delta T Z_u C_2 z_u^2(t-1) r_u^2(t-1) b_h,$$

$$c_2 = Z_u z_u^2(t-1) r_u^2(t-1) b_h \sum_{(i,j)\in M_{u,k}^e} R_{ij}^1,$$

$$c_3 = Z_u C_2 r_u^2(t-1) b_h \sum_{(i,j)\in M_{u,k}^e} R_{ij}^1.$$

Now, let $D^{2,\mathrm{e}}_{u,k}(t) := D^{2,\mathrm{e}}_{u,k}(R^{12}_{ij}(t))$. The optimization problem to be solved at iteration t is:

$$\min_{\substack{\{R^{12}_{ij}\},\\ \{z^i_u\},\{r^i_u\}}} \; D^{2,\mathrm{e}}_{u,k}(t),$$

$$\text{s.t.} \quad (13), (14), (15), \text{ and } (16). \tag{17}$$

Here, (17) is a GP problem and we can solve it optimally. We carry out the optimization iteratively until $|D^{2,\mathrm{e}}_{u,k}(t) - D^{2,\mathrm{e}}_{u,k}(t-1)| \leq \epsilon$, for some small $\epsilon \geq 0$. When this condition is met, we obtain the optimal value of the objective function in (17) as $D^{2,\mathrm{e},*}_{u,k} = D^{2,\mathrm{e}}_{u,k}(t)$. Similarly, we obtain the optimal streaming data rate and client computing capability allocations as $\{R^{12*}_{ij}\} = \{R^{12}_{ij}(t)\}$ and $\{z^{i*}_u\} = \{z^i_u(t)\}, \{r^{i*}_u\} = \{r^i_u(t)\}$. Finally, given these developments, we can formulate the optimal value of the overall objective function in (11), for the given $M^r_{u,k}$ and k, as

$$D^{2*}_{u,k} = \sum_{(i,j)\in M^r_{u,k}} w_{ij} P^u_{ij} a_{ij} R^{b_{ij}}_{ij,\max} + D^{2,\mathrm{e},*}_{u,k}. \tag{18}$$

This completes the solution to (11). We repeat the above procedure for every $M^r_{u,k}$ and k, keeping track of the obtained solution and $D^{2*}_{u,k}$ in each case. Then, we select the overall best solution that integrates the choice of optimal set M^r_u as

$$D^{2\star}_u = \min D^{2*}_{u,k}, \tag{19}$$

$$\{R^{12\star}_{ij}\}, \underset{\{z^{i*}_u\},\{r^{i*}_u\}}{M^{\mathrm{r}\star}_u} = \arg\min_{\substack{\{R^{12*}_{ij}\},M^r_{u,k}\\ \{z^{i*}_u\},\{r^{i*}_u\}}} D^{2*}_{u,k}. \tag{20}$$

To complete the solution to the original optimization problem of interest (4)–(9), we set $\{R^{2\star}_{ij}\} = \{R^{12\star}_{ij} - R^{1\star}_{ij}\}$ and recall that $\{R^{1\star}_{ij}\} = \{R_{ij,\min}\}$. A formal description of our iterative optimization framework described herein is provided in Algorithm 2, which is activated every ΔT periods of time. In Line 8, we numerically implemented the geometric programming solution to (17) using the Matlab toolbox GGPLAB.

We assess the computational complexity of Algorithm 2 as follows. Similarly to Algorithm 1, for a given set $M^{\mathrm{r}}_{u,k}$, the number of required iterations for Algorithm 2 is $\frac{\log(\mathcal{K}_2)/t_{0,1}\epsilon}{\log\xi}$, where $\mathcal{K}_2$ is the number of constraints in (17), $t_{0,1}$ is the initial point to approximate the accuracy of the interior point method used to solve (17), $0 < \epsilon < 1$ is the stopping criterion for the interior point method, and ξ is used for updating the accuracy of the interior point method [32]. This inner optimization is carried out in Algorithm 2 $|M^s_u| + 1$ times, for different sets $M^{\mathrm{r}}_{u,k}$. Thus, the overall computational complexity of Algorithm 2 is on the order of $(|M^s_u| + 1)\frac{\log(\mathcal{K}_2)/t_{0,1}\epsilon}{\log\xi}$. Table II shows the number of iterations for Algorithm 2 for streaming different GOPs of the 360° video *Runner* for different values of the NR link capacity, server decoding speed, headset decoding and rendering speeds, and optimization threshold ϵ.

Algorithm 2: Dual Connectivity Optimal Resource Allocation

1: Initialize $D^{2*,\mathrm{set}}_{u,k} = \{\}$, $R^1_{ij} = R_{ij,\min}, \forall (i,j) \in M_u$
2: Sort tiles $m_{ij} \in M_u$ in descending order using $w_{ij}P^u_{ij}$ ⇒ Produce sorted tile set $|M^s_u|$
3: **for** k = 0 to $|M^s_u|$ **do**
4: Set $M^{\mathrm{r}}_{u,k}$ to comprise first k elements of $|M^s_u|$
5: Initial. $t = 0, r^i_u(t) = r_u/2, z^i_u(t) = z_u/2$, for $i = 1, 2$, $R^{12}_{ij}(t) \in [R^1_{ij}, R_{ij,\max}], \forall (i,j) \in M_u$, and $D^{2,\mathrm{e}}_{u,k}(t)$
6: **while** true **do**
7: $t = t + 1$
8: Determine the optimal $r^1_u(t)$, $r^2_u(t)$, $z^1_u(t)$, $z^2_u(t)$, $R^{12}_{ij}(t)$, $D^2_{u,k}(t)$ by solving (17) using GGPLAB
9: **if** $|D^2_{u,k}(t) - D^2_{u,k}(t-1)| \leq \epsilon$ **then**
10: Break
11: **end if**
12: **end while**
13: Set $\{R^{12*}_{ij}\} = \{R^{12}_{ij}(t)\}, \{z^{i*}_u\} = \{z^i_u(t)\}$, and $\{r^{i*}_u\} = \{r^i_u(t)\}$
14: Compute $D^{2*}_{u,k}$ using (18)
15: Set $D^{2*,\mathrm{set}}_{u,k} = D^{2*,\mathrm{set}}_{u,k} \cup D^{2*}_{u,k}$
16: **end for**
17: $D^{2*}_u = \min D^{2*,\mathrm{set}}_{u,k}$
18: $\{R^{12\star}_{ij}\}, M^{\mathrm{r}\star}_u, \{z^{i\star}_u\}, \{r^{i\star}_u\} = \arg\min D^{2*,\mathrm{set}}_{u,k}$
19: Set $\{R^{2\star}_{ij}\} = \{R^{12\star}_{ij} - R^1_{ij}\}$

VII. Simulation Experiments and Analysis

A. 3-DOF Navigation Data Capture

We collected head navigation data from real VR users to evaluate the performance of our proposed system. We used an HTC Vive wireless headset and the software packages SteamVR SDK [35] and Opentrack [36] to capture the navigation information. We collected head movement traces for the two popular 8 K 360° video sequences *Runner* and *Basketball* [37]. We captured navigation data for ten volunteer users individually, in the case of each content. 1080 navigation samples were captured per session, at 30 samples per second. We considered half of these users as single connectivity clients and the other half as dual connectivity clients in our experimental assessment, as indicated later on.

B. Experimental Setup

We evaluated the performance of our system through simulation experiments using the captured navigation data. We assessed the delivered immersion fidelity (expected viewport quality) enabled by our system using its expected WS-PSNR (Peak Signal to Noise Ratio), formulated as $10\log_{10}(255^2/\sum_{ij\in M_u} w_{ij}P^u_{ij}D_{ij})$, where D_{ij} denotes the reconstruction error of GOP tile $m_{ij} \in M_u$, as introduced earlier. We obtain the WS-PSNR for an entire video by computing the WS-PSNR for each GoP and then averaging over them. We streamed the 360° video *Runner* or *Basketball* scalable encoded at different data rates, at 8K-30fps spatial resolution and frame rate to each user. We measured the WS-PSNR of the user viewport in each case. We set the LTE channel capacity to 50 Mbps. We list the key simulation parameter values used in our evaluation in Table III.

TABLE II
THE COMPUTATIONAL COMPLEXITY OF ALGORITHM 2

C_2 (Gbps)	Z_u (Gbps)	z_u (Mbps)	r_u (GPixels/s)	ϵ	Iterations (GOP 1)	Iterations (GOP 18)	Iterations (GOP 36)
0.7	1.5	100	1.88	0.001	4	4	4
0.7	1.5	100	1.88	0.0001	8	8	8
1	3	300	9.4	0.001	6	6	6
1	3	300	9.4	0.0001	12	12	8
1.2	6	500	28.2	0.001	7	7	7
1.2	6	500	28.2	0.0001	14	14	9

TABLE III
SIMULATION PARAMETERS

Minimum GoP tile encoding rate $R_{ij,min}$	27.58 Kb/s
Maximum GoP tile encoding rate $R_{ij,max}$	29.56 Mb/s
360° content spatial resolution	8K
Frame rate	30 fps
System latency requirement, ΔT	1 seconds
Number of GoP tiles	64
LTE link capacity, C_1	50 Mbps
5G NR link capacity, C_2	0.7 Gbps to 1.2 Gbps
Server decoding speed, Z_u	1.5 Gbps to 6 Gbps
Mobile device decoding speed, z_u	100 Mbps to 500 Mbps
Mobile device rendering speed r_u	1.88 GPixels/s to 28.2 GPixels/s

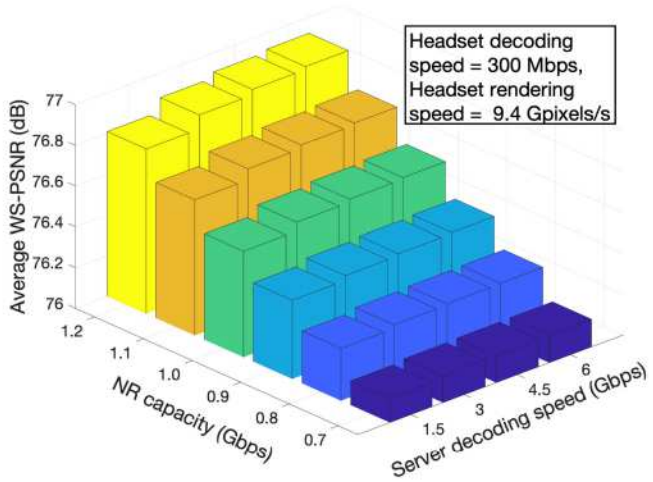

Fig. 4. Impact of NR channel capacity and server decoding speed on delivered immersion fidelity.

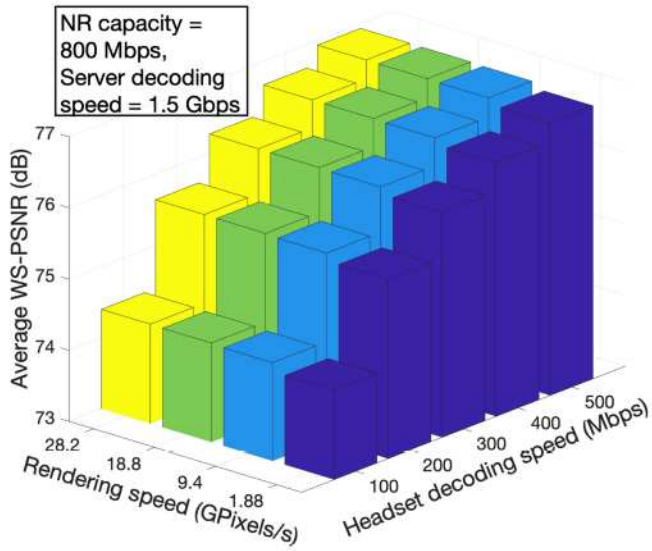

Fig. 5. Viewport quality for different client device computing capability.

C. Results and Analysis

1) Impact of NR Link Capacity and Server Decoding Speed: In Fig. 4, we show how the available NR link rate and the decoding capability of the edge server affects the delivered immersion fidelity. We can see that for NR link capacities between 700 Mbps to 1.2 Gbps, increasing the server decoding speed does not improve the WS-PSNR. At such link capacities, almost all the GoP-tiles comprising the enhancement representation of the user viewport are streamed in compressed format. Very few GoP-tiles are decoded at the server to be streamed as raw data. Thus, the server decoding speed does not impact the delivered immersion fidelity significantly in such cases. As the NR link capacity increases, the number of GoP-tiles to be decoded at the server and streamed as raw data increases, and the GoP-tiles to be streamed as compressed data are encoded at higher rates. Thus, the immersion fidelity improves with an increase in the channel capacity. For example, when the server decoding speed speed is 6 Gbps, the WS-PSNR increases from 76.1 dB to 76.9 dB as the NR link capacitiy increases from 700 Mbps to 1.2 Gbps.

2) Impact of Client Device Computing Capability: We show the impact of the computing capability of a client device on the delivered immersion fidelity in Fig. 5 for an NR link capacity of 800 Mbps and a server decoding speed of 1.5Gbps. We can see that the WS-PSNR increases ≈0.2 dB as the rendering speed increases from 1.88 Gpixels/s to 9.4 Gpixels/s. Further increasing the rendering speed does not impact the delivered immersion fidelity. But we can also observe that an increase in the decoding speed of the device improves the WS-PSNR. For any value of the rendering speed, a gain of ≈2.5 dB is obtained as the device's decoding speed is increased from 100 Mbps to 500 Mbps. As the decoding speed increases, the device can decode GOP tiles compressed at higher data rates, hence the improvement in delivered viewport quality.

3) Impact of Client Decoding Speed and NR Link Capacity: In Fig. 6, we examine the joint impact of NR link capacity and client device decoding capability on the delivered viewport quality. We can see that the WS-PSNR improves significantly as the NR capacity and the device decoding speed increase. For example, the WS-PSNR increases more than 1.5 dB when the NR capacity increases from 700 Mbps to 1.2 Gbps for headset decoding speed of 100 Mbps. Similarly, when the NR capacity is 700 Mbps, the WS-PSNR increases from 74.8 dB to 77.2 dB as the client's decoding speed increases from 100 Mbps to 500 Mbps. Moreover, a WS-PSNR gain of ≈4 dB is achieved when the link capacity is increased from 700 Mbps to 1.2 Gbps

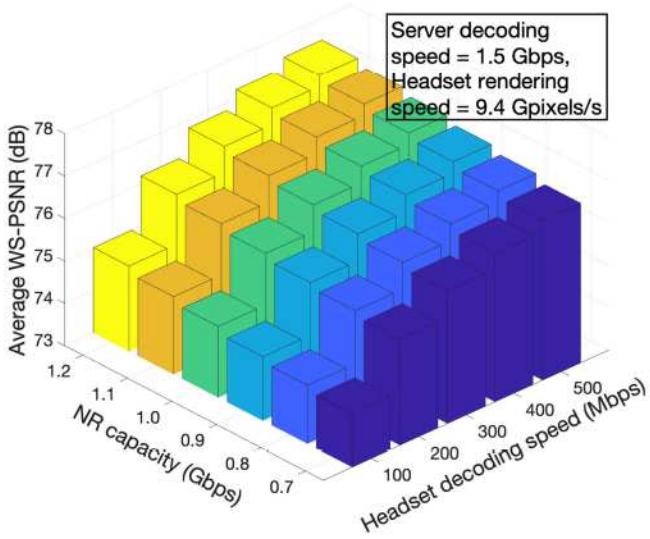

Fig. 6. Viewport quality for different client device decoding speeds and NR link capacities.

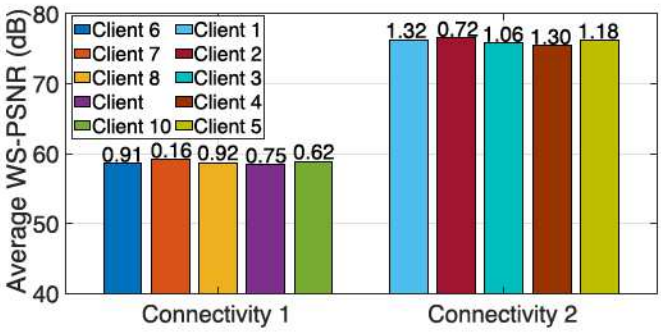

Fig. 7. Comparison of delivered immersion fidelity and its variation between single connectivity (Connectivity 1) clients and dual connectivity (Connectivity 2) clients for the 360° content *Runner*.

and the device decoding speed is increased from 100 Mbps to 500 Mbps. Increasing the value of either of these parameters allows GOP tiles encoded at higher rates to be streamed from the server and decoded in time on the client device, thus improving the WS-PSNR significantly.

4) Single Connectivity vs. Dual Connectivity: In Fig. 7, we show the expected value and standard deviation (shown on top of the respective bar) of WS-PSNR over the duration of the live session, per client, for both single connectivity and dual connectivity cases. We can observe that an up to 20 dB gain in delivered immersion fidelity can be experienced by a client with dual connectivity relative to a client with single connectivity. This is expected as the *Connectivity-1* clients are served a baseline representation of the 360° content via a low capacity LTE link. On the other hand, the *Connectivity-2* clients are served with a viewport-specific enhancment representation over a high capacity NR link in parallel to a baseline representation over the LTE link. We can also see that the variation in WS-PSNR is lower for single connectivity clients compared to dual connectivity clients.

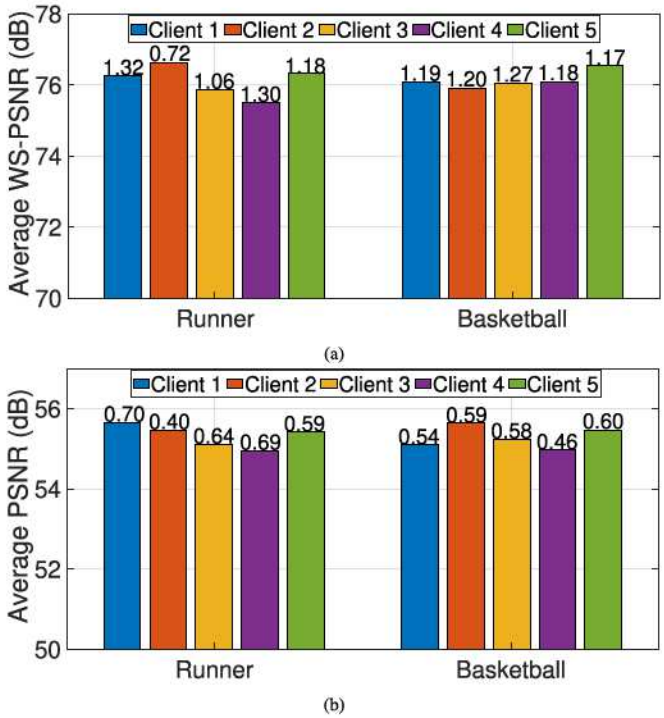

Fig. 8. Immersion fidelity and its variation for different 360° content for clients with dual connectivity (a) in terms of WS-PSNR and (b) in terms of PSNR.

5) Performance Consistency for Different 360° Content: In Fig. 8(a), we examine the delivered expected immersion fidelity per dual connectivity client under our framework (*in terms of WS-PSNR*) and its standard deviation (shown on top of the respective bar), for the two 360° video sequences *Runner* and *Basketball* used in our experiments. We can see that save for the expected slight viewport quality variation across different clients, due to their specific navigation patterns, spatial position, and content served, our system consistently delivers high quality immersion across all clients and for both 360° videos we examined. These advances are enabled by the synergistic integration within our framework of several emerging technologies and our resource allocation optimization that maximizes the system efficiency and delivered quality of experience, for a given client and content.

We observe alike outcomes in Fig. 8(b), which shows the same results but with the delivered expected immersion fidelity and its standard deviation expressed in terms of PSNR. Again, our framework enables consistently high quality immersion across different clients and streamed 360° content, for another popular video quality metric. The minor relative performance differences across clients and content, when contrasting the results in Fig. 8(a) and (b), and the absolute performance differences between WS-PSNR and PSNR results, are expected and stem from the spherical distortion coefficients w_{nm} employed in the WS-PSNR formulation.

6) Impact of Inaccurate Navigation Likelihoods: Here, we investigate how inaccurate navigation likelihoods and navigation prediction, as introduced in Section III-B, can impact the performance of our system. In particular, for each GOP tile $m_{ij} \in M_u$, for a given mobile client $u \in U$ and 360° video, we define a

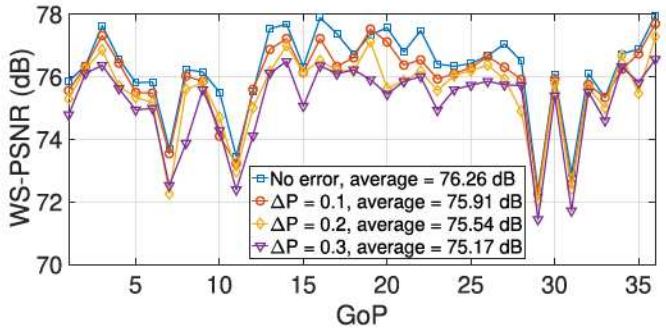

Fig. 9. Impact of inaccurate navigation likelihoods on delivered immersion fidelity for the 360° video *Runner* and dual connectivity streaming.

navigation likelihood error $\Delta P^u_{ij} = \Delta P \cdot P^u_{ij}$, where ΔP is a constraint fraction or percentage point. We then construct inaccurate navigation likelihoods for the current GOP $\{\hat{P}^u_{ij}\}$, as an output of a prospective navigation prediction method, using $\hat{P}^u_{ij} = P^u_{ij} \pm \Delta P^u_{ij}, \forall (i,j) \in M_u$. The sign of the introduced error ΔP^u_{ij} is selected at random and the constructed inaccurate likelihoods are re-normalized to one, to ensure they represent a probability distribution. We then compute the optimal resource allocation solution produced by our optimization strategies from Section VI using these inaccurate likelihoods $\{\hat{P}^u_{ij}\}$. Finally, we use the computed resource allocation to evaluate the actual performance (delivered viewport quality) on client u navigating the given 360° content using the original likelihoods $\{P^u_{ij}\}$, as that is what the client would be using in reality.

We can observe from Fig. 9, that as ΔP increases, the client viewport WS-PSNR decreases, as expected. For example, when there is no error the average viewport WS-PSNR over the streaming session is 76.26 dB. However, when $\Delta P = 0.1, 0.2, 0.3$, the average WS-PSNR reduces to 75.91 dB, 75.54 dB, and 75.17 dB, respectively. We can see that the drop in delivered immersion fidelity is only ≈1.1 dB even when $\Delta P = 0.3$. These results show that our proposed optimization strategies are robust to inaccurate navigation likelihoods and user navigation prediction.

7) WS-PSNR vs. PSNR: In this section, we assess the immersion fidelity enabled by our system, computed as the expected PSNR (Peak Signal to Noise Ratio), which we formulate as $10\log_{10}(255^2/\sum_{ij\in M_u} P^u_{ij} D_{ij})$. In Fig. 10(a) we show the cumulative distribution of the delivered immersion fidelity for a client with single connectivity both in terms of WS-PSNR and PSNR. We can observe that for both *Runner* and *Basketball*, the WS-PSNR is higher than the PSNR. This is expected as the expression for PSNR does not contain the term w_{ij}, which has values less than 1. Thus, the distortion is higher and results in smaller values for PSNR. Similary in Fig. 10(b), we show the PSNR and WS-PSNR for a client with dual connectivity. We can see from both Fig. 10(a) and (b) that the WS-PSNR achieved for *Runner* is slightly higher than that for *Basketball*. We observe the same behaviour in performance also in terms of PSNR. Thus, our results in terms of WS-PSNR and PSNR are consistent.

8) Comparison With the State-of-the-Art: We implemented a state-of-the-art reference method [9] that uses the latest MPEG-DASH streaming standard, to deliver the HEVC-compressed 3DOF 360° content to clients in our system over LTE. We also implemented the 360° multicast method $VRCast$

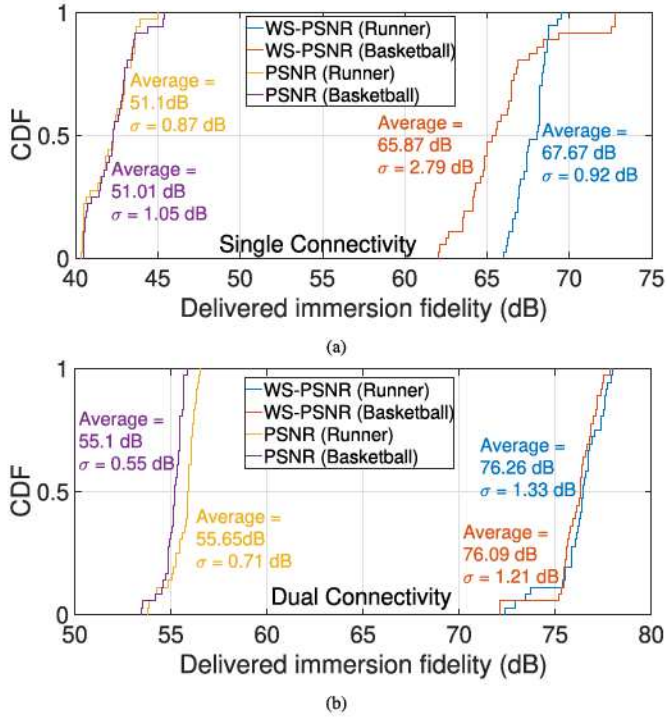

Fig. 10. PSNR and WS-PSNR for a client with (a) single connectivity and (b) dual connectivity.

[23] in our 5G NR dual-connectivity setting. The performances of these methods for streaming the 8K-30fps 360° videos *Runner* and *Basketball* are included in Fig. 11(a) and (b), respectively.

It can be seen that MPEG-DASH enables moderate viewport quality (63.68 dB for *Runner* and 63.99 dB for *Basketball*) at 8K-30fps. These outcomes are in line with present wireless methods that can only stream 360° videos at low to medium quality. VRCast also provides moderate viewport quality (63.72 dB–64.88 dB) even though it is implemented in our daul connectivity setting, since it does not decode any tiles at the edge server and stream them utilizing the capacity of the NR link. The decoding and rendering speeds of the client's mobile device affects the performance. On the other hand, our dual connectivity streaming enables high-fidelity viewport (72 dB–78 dB) at 8K–30fps for both 360° videos, which considerably advances the state-of-the-art. In particular, our approach enables gains of up to 12.5–13 dB gain in immersion fidelity over the two reference methods.

The performance gains of our approach are enabled by several key novelties such as parallel LTE and NR link transmission, scalable multi-layer 360° tiling design, and optimal allocation of communication and computing resources. Similarly, we can see that our single connectivity streaming provides an up to 4.5 dB gain for *Runner* and an up to 3.8 dB gain for *Basketball* over the reference methods, which is also enabled by the proposed scalable multi-layer 360° tiling and optimal allocation of communication resources in this case.

9) Impact of Transient NR Link Loss: Here, we examine the benefits and system reliability enabled by our LTE-NR dual connectivity streaming. In Fig. 12, we examine the delivered immersion fidelity over time for a dual-connectivity user, where we include as a reference a variant of our system that uses only

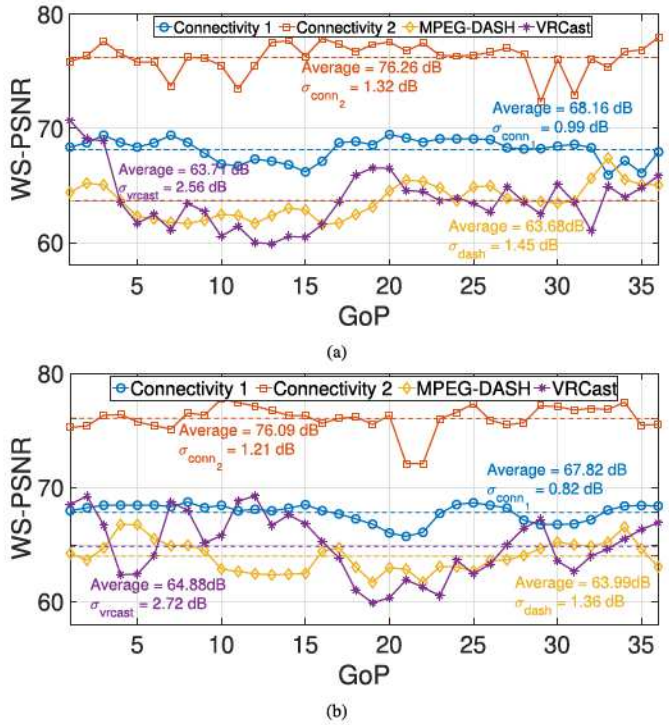

Fig. 11. Performance comparison between our proposed methods (single and dual connectivity) and the state-of-the-art methods MPEG-DASH and VRCast (streamed 360° content (a) *Runner*) and (b) *Basketball*).

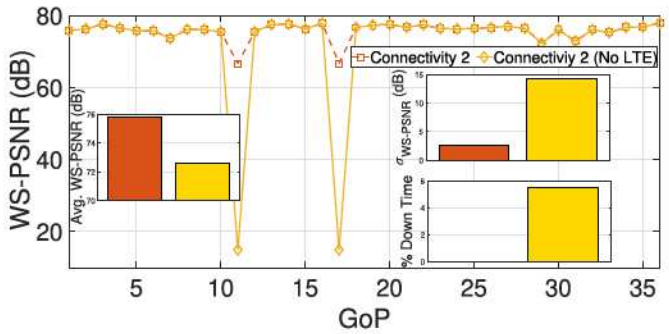

Fig. 12. Performance comparison for our proposed method with and without a parallel LTE link (for streaming the 360° video *Runner*).

the NR link (single connectivity) to stream the content. We can observe Fig. 12 that notable WS-PSNR variation is observed at some time instances for both methods under comparison. These are due to transient NR link drops or a mismatch between the delivered enhancement representation viewport content and the actual user viewport, each caused by rapid head navigation movements. Still, the observed immersion fidelity variation is much lower for our dual-connectivity system, which considerably increases its quality of experience and reliability, relative to *Connectivity 2 (No LTE)* that experiences an application downtime during such instances. Concretely, a viewport quality gain of 4 dB and a six times smaller standard deviation of viewport quality are enabled over the latter method, as seen from the two respective graph insets in Fig. 12. Similarly, we can also observe that the user experienced a 5.5% application downtime under the *Connectivity 2 (No LTE)* method, compared to 0% downtime in the case of our method. These benefits merit the integrated dual-connectivity streaming proposed by our framework.

10) Other Performance Measurement Metrics: In this section, we compare the performance of our methods with MPEG-DASH and VRCast with respect to multiple alternative performance measurement metrics. We obtained these results for two more videos, *Academic* and *Bridge* [37], in addition to *Basketball* and *Runner*. Table IV compiles the performance of the different methods in terms of the video quality metrics: PSNR, Mean-Squared-Error (MSE), Structural Similarity Index Measure (SSIM) [38], and Video Multimethod Assessment Fusion (VMAF) [39]. In particular, these results are shown in both mean and standard deviation values, for each metric, 360° video, and method. Again, we can observe that our proposed methods (Connectivity 1 and Connectivity 2) perform better than MPEG-DASH and VRCast for all four videos in terms of the alternative metrics as well.

11) Impact of LTE Link Capacity: We set the LTE channel capacity to 50 Mbps for our simulation experiments. In this section, we set this capacity uniformly distributed between 20 Mbps–100 Mbps [40] to observe its impact. We conducted the simulation experiment in this setup using the video *Runner* for Connectivity 1, Connectivity 2, MPEG-DASH, and VRCast. The results are presented in Table V. We can see that the varying LTE channel capacity does not cause any significant changes in the results for any of the methods under examination and that our approach continues to demonstrate consistent performance gains.

12) Dataset Contribution: To carry out the evaluations in Tables IV and V, we first generated and recorded the rate-quality trade-offs for each 360° GOP tile of all four 8 K (full UHD) 360° video sequences used in our evaluation, for the four quality metrics considered in Tables IV and V. To promote further research and follow up work, we have organized this information in a dataset that we contribute to the community via this publicly accessible web link [41]. The rate-distortion trade-offs and navigation traces for the 360° content we used have been contributed earlier in [42], and they have been used similarly in recent studies [21], [22].

VIII. Conclusion

We studied rate-distortion-computing optimized live 360° video streaming to heterogeneous mobile VR clients in 5G networks, featuring devices with single (LTE) or dual (LTE/NR) cellular connectivity. We compress the live content using scalable 360° tiling at the origin and send it towards the clients over a single backbone network link. A mobile edge server then adapts the incoming streaming data to the individual clients and their respective down-link transmission rates using formal rate-distortion-computing optimization. Single connectivity clients are served by the edge server a baseline representation/layer of the content adapted to their down-link transmission capacity and device computing capability. A dual connectivity client is served in parallel a baseline content layer on its LTE connectivity and a complementary viewport-specific enhancement layer on its NR connectivity, synergistically adapted to the respective down-links' transmission capacities and the client's computing capability.

TABLE IV
PERFORMANCE RESULTS ACROSS FOUR DIVERSE METRICS AND 360° VIDEOS. MEAN AND STANDARD DEVIATION VALUES ARE SHOWN

	PSNR (Avg.) (dB)	σ_{PSNR} (dB)	MSE (Avg.)	σ_{MSE}	SSIM-Y (Avg.)	σ_{SSIM-Y}	VMAF (Average)	σ_{VMAF}
Academic (Connectivity 1)	54.9638	1.1187	0.2141	0.05547	0.9859	0.0025	91.948	1.0072
Academic (Connectivity 2)	62.4936	1.44465	0.03878	0.01437	0.9947	0.00113	95.554	0.5434
Academic (MPEG-DASH)	50.2078	0.5791	0.6252	0.0835	0.9859	0.0012	92.321	0.5369
Academic (VRCast)	47.1799	3.3462	1.6079	1.0833	0.9622	0.0047	82.402	1.9377
Basketball (Connectivity 1)	51.0085	1.05414	0.5309	0.1356	0.98148	0.00349	90.701	1.4747
Basketball (Connectivity 2)	55.1012	0.5481	0.2026	0.0282	0.9916	0.0021	94.95	0.7083
Baskeball (MPEG-DASH)	47.4316	1.2551	1.2225	0.3468	0.9743	0.0051	89.465	1.4033
Basketball (VRCast)	46.7392	2.42882	1.6095	0.9523	0.9756	0.0025	86.22	0.8562
Bridge (Connectivity 1)	51.1954	0.3269	0.4952	0.0390	0.9789	0.0019	82.613	1.5404
Bridge (Connectivity 2)	53.6444	0.3707	0.2819	0.02462	0.9891	0.0015	90.278	1.1876
Bridge (MPEG-DASH)	49.4778	0.2412	0.7344	0.0394	0.9751	0.0022	83.26	0.5897
Bridge (VRCast)	47.7255	1.3853	1.1512	0.3492	0.9768	0.0009	73.896	0.6937
Runner (Connectivity 1)	51.1001	0.8669	0.51473	0.10479	0.98128	0.00333	88.807	1.662
Runner (Connectivity 2)	55.6547	0.7131	0.1794	0.0328	0.9916	0.0014	92.956	0.7614
Runner (MPEG-DASH)	47.2395	1.3924	1.28967	0.40497	0.97413	0.0038	87.852	0.38036
Runner (VRCast)	45.8767	2.3598	1.91014	0.91326	0.9671	0.00302	79.8	1.086

TABLE V
PERFORMANCE RESULTS ACROSS FOUR DIVERSE METRICS IN THE CASE OF A TIME-VARYING LTE CHANNEL CAPACITY. MEAN AND STANDARD DEVIATION VALUES ARE SHOWN

	PSNR (Avg.) (dB)	σ_{PSNR} (dB)	MSE (Avg.)	σ_{MSE}	SSIM-Y (Avg.)	σ_{SSIM-Y}	VMAF (Average)	σ_{VMAF}
Runner (Connectivity 1)	51.159	1.2554	0.5347	0.2341	0.9811	0.0058	88.801	2.1667
Runner (Connectivity 2)	55.6545	0.7133	0.1794	0.03278	0.9916	0.0014	92.955	0.7616
Runner (MPEG-DASH)	47.3913	2.1387	1.3463	0.7639	0.9727	0.009	87.545	2.6576
Runner (VRCast)	45.8767	2.3598	1.9101	0.9133	0.9677	0.0031	79.899	1.1049

We formulated two optimization problems to conduct the operation of the edge server in each case, taking into account the key system components of the delivery process and induced end-to-end latency, aiming to maximize the immersion fidelity delivered to each client. We explored respective geometric programming optimization strategies that compute the optimal solutions at lower complexity, and rigorously examined their computational complexity. We demonstrated viewport quality gains of up to 4 dB and 13 dB respectively for 4 G/LTE and LTE/NR clients, over two state-of-the art reference methods. We also showed that our system is robust to imprecise navigation likelihood prediction, transient NR link loss, dynamic LTE bandwidth variations, and diverse 360° video content. We contrasted our results over five popular video quality metrics that verified our framework's consistent performance. Lastly, we demonstrated that 8K–30 fps 360° videos can be streamed at high fidelity using our system.

We share with the broad TMM community a dataset that captures the rate-quality trade-offs of the 360° video content used in our evaluation, for multiple contemporary quality metrics, to stimulate further studies and follow up work [41].

ACKNOWLEDGMENT

The authors express their sincere gratitude to the associate editor, anonymous reviewers, editor in chief, and society administrator for the constructive review process that has helped improve the quality of the manuscript. They extend their gratitude to Zhiheng Xu of NJIT for his help in generating the rate-quality trade-offs dataset used in the evaluation.

REFERENCES

[1] J. Chakareski, "UAV-IoT for next generation virtual reality," *IEEE Trans. Image Process.*, vol. 28, no. 12, pp. 5977–5990, Dec. 2019.
[2] J. Chakareski, M. Khan, and M. Yuksel, "Towards enabling next generation societal virtual reality applications for virtual human teleportation," *IEEE Signal Process. Mag.*, vol. 39, no. 5, pp. 22–41, Sep. 2022.
[3] J. Chakareski and P. Frossard, "Distributed collaboration for enhanced sender-driven video streaming," *IEEE Trans. Multimedia*, vol. 10, no. 5, pp. 858–870, Aug. 2008.
[4] B. Begole, "Why the internet pipes will burst when virtual reality takes off," *Forbes Mag.*, Feb. 2016.
[5] Global System for Mobile Communications Alliance (GSMA) Intelligence, "The 5G guide: A reference for operators," Apr. 2019. [Online]. Available: https://www.gsma.com/wp-content/uploads/2019/04/The-5G-Guide_GSMA_2019_04_29_compressed.pdf
[6] J. Chakareski, "Viewport-adaptive scalable multi-user virtual reality mobile-edge streaming," *IEEE Trans. Image Process.*, vol. 29, no. 1, pp. 6330–6342, Dec. 2020.
[7] I. Sodagar, "The MPEG-DASH standard for multimedia streaming over the internet," *IEEE Multimedia*, vol. 18, no. 4, pp. 62–67, Apr. 2011.
[8] C. Ozcinar, A. D Abreu, and A. Smolic, "Viewport-aware adaptive 360° video streaming using tiles for virtual reality," in *Proc. IEEE Int. Conf. Image Process.*, 2017, pp. 2174–2178.
[9] S. Petrangeli, V. Swaminathan, M. Hosseini, and F. D. Turck, "Improving virtual reality streaming using HTTP/2," in *Proc. ACM Multimedia Syst. Conf.*, 2017, pp. 225–228.
[10] M. Hosseini and V. Swaminathan, "Adaptive 360 VR video streaming: Divide and conquer," in *Proc. IEEE Int. Symp. Multimedia*, 2016, pp. 107–110.
[11] J. Dai, Z. Zhang, S. Mao, and D. Liu, "A view synthesis-based 360° VR caching system over MEC-enabled C-RAN," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 10, pp. 3843–3855, Oct. 2020.
[12] P. Maniotis and N. Thomos, "Tile-based edge caching for 360° live video streaming," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 12, pp. 4938–4950, Dec. 2021.

[13] N. Kan, J. Zou, C. Li, W. Dai, and H. Xiong, "RAPT360: Reinforcement learning-based rate adaptation for 360-degree video streaming with adaptive prediction and tiling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 3, pp. 1607–1623, Mar. 2022.
[14] C. Zheng et al., "STC: FoV tracking enabled high-quality 16K VR video streaming on mobile platforms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 2396–2410, Apr. 2022.
[15] L. Sun, Y. Mao, T. Zong, Y. Liu, and Y. Wang, "Live 360 degree video delivery based on user collaboration in a streaming flock," *IEEE Trans. Multimedia*, vol. 25, pp. 2636–2647, 2023.
[16] X. Hou, S. Dey, J. Zhang, and M. Budagavi, "Predictive adaptive streaming to enable mobile 360-degree and VR experiences," *IEEE Trans. Multimedia*, vol. 23, pp. 716–731, 2021.
[17] Q. Cheng et al., "Design and analysis of MEC- and proactive caching-based 360° mobile VR video streaming," *IEEE Trans. Multimedia*, vol. 24, pp. 1529–1544, 2022.
[18] J. Chakareski, "Multi-path content delivery: Efficiency analysis and optimization algorithms," *J. Vis. Commun. Image Representation*, vol. 23, no. 8, pp. 1189–1198, Nov. 2012.
[19] J. G. Apostolopoulos and M. D. Trott, "Path diversity for enhanced media streaming," *IEEE Commun. Mag.*, vol. 42, no. 8, pp. 80–87, Aug. 2004.
[20] J. Chakareski, S. Han, and B. Girod, "Layered coding vs. multiple descriptions for video streaming over multiple paths," *Multimedia Syst.*, vol. 10, pp. 275–285, 2005.
[21] S. Gupta, J. Chakareski, and P. Popovski, "mmWave networking and edge computing for scalable 360-degree video multi-user virtual reality," *IEEE Trans. Image Process.*, vol. 32, pp. 377–391, 2023.
[22] J. Chakareski, M. Khan, T. Ropitault, and S. Blandino, "Millimeter wave and free-space-optics for future dual-connectivity 6DOF mobile multi-user VR streaming," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 19, no. 2, pp. 57:1–57:25, Feb. 2023.
[23] O. Eltobgy, O. Arafa, and M. Hefeeda, "Mobile streaming of live 360-degree videos," *IEEE Trans. Multimedia*, vol. 22, no. 12, pp. 3139–3152, Dec. 2020.
[24] A. Majidi and A. H. Zahran, "Optimized joint unicast-multicast panoramic video streaming in cellular networks," in *Proc. IEEE 28th Int. Conf. Netw. Protoc.*, 2020, pp. 1–6.
[25] J. Chakareski and S. Gupta, "Multi-connectivity and edge computing for ultra-low-latency lifelike virtual reality," in *Proc. Int. Conf. Multimedia Expo.*, 2020, pp. 1–6.
[26] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proc. Workshop Things Cellular: Operations, Appl. Challenges*, 2016, pp. 1–6.
[27] A. T. Nasrabadi, A. Mahzari, J. D. Beshay, and R. Prakash, "Adaptive 360-degree video streaming using scalable video coding," in *Proc. ACM Multimedia Conf.*, 2017, pp. 1689–1697.
[28] H. Ahmadi, O. Eltobgy, and M. Hefeeda, "Adaptive multicast streaming of virtual reality content to mobile users," in *Proc. Thematic Workshops ACM Multimedia*, 2017, pp. 170–178.
[29] R. Aksu, J. Chakareski, and V. Swaminathan, "Viewport-driven rate-distortion optimized scalable live 360° video network multicast," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops*, 2018, pp. 1–6.
[30] Y. Sun, A. Lu, and L. Yu, "Weighted-to-spherically-uniform quality evaluation for omnidirectional video," *IEEE Signal Process. Lett.*, vol. 24, no. 9, pp. 1408–1412, Sep. 2017.
[31] V. Vazirani, *Approximation Algorithms*, 2nd ed. Berlin, Germany: Springer, 2003.
[32] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge Univ. Press, 2004.
[33] G. Xu, "Global optimization of signomial geometric programming problems," *Eur. J. Oper. Res.*, vol. 233, no. 3, pp. 500–510, 2014.
[34] "GGPLAB: A simple MATLAB toolbox for geometric programming," [Online]. Available: http://www.stanford.edu/boyd/ggplab/
[35] "SteamVR SDK," [Online]. Available: https://github.com/ValveSoftware/openvr
[36] "Opentrack VR tracking," [Online]. Available: https://github.com/opentrack/opentrack
[37] X. Liu, Y. Huang, L. Song, R. Xie, and X. Yang, "The SJTU UHD 360-degree immersive video sequence dataset," in *Proc. IEEE Int. Conf. Virtual Reality Visual.*, 2017, pp. 400–401.
[38] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[39] A. Aaron et al., "Challenges in cloud based ingest and encoding for high quality streaming media," in *Proc. IEEE Int. Conf. Image Process.*, 2015, pp. 1732–1736.
[40] J. V. D. Hooft, T. Wauters, F. D. Turck, C. Timmerer, and H. Hellwagner, "Towards 6DOF HTTP adaptive streaming through point cloud compression," in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 2405–2413.
[41] M. Khan and J. Chakareski, "NJIT Full UHD (8K) video rate-quality trade-offs dataset," [Online]. Available: https://www.jakov.org
[42] J. Chakareski, R. Aksu, V. Swaminathan, and M. Zink, "Full UHD 360-degree video dataset and modeling of rate-distortion characteristics and head movement navigation," in *Proc. ACM Multimedia Syst. Conf.*, 2021, pp. 267–273.

Jacob Chakareski completed his Ph.D. degree in electrical and computer engineerin at Rice University, Houston, TX, USA, and Stanford University, Stanford, CA, USA. He is currently an Associate Professor in the Ying Wu College of Computing, New Jersey Institute of Technology (NJIT), Newark, NJ, USA, where he holds the Panasonic Chair of Sustainability. He directs the Laboratory for AI-enabled Wireless XR Systems and Societal Applications at NJIT. He has held research appointments with Microsoft, HP Labs, EPFL and Vidyo, and was on the Advisory Board of Frame, Inc. He has authored one monograph, five book chapters, more than 200 IEEE/ACM international journal/conference publications, and holds nine U.S. patents. His research interests include mobile virtual and augmented reality (VR/AR) systems, UAV-IoT sensing and communication, fast physics-aware reinforcement learning for emerging IoT systems, millimeter wave and free-space optical wireless networking, scalable multi-connectivity 360° video streaming, 5G edge computing and caching, emerging 5G/IoT architectures, AI-enabled edge networks, neural enhanced video streaming, ubiquitous immersive communication, and societal applications. He was the recipient of the Adobe Data Science Faculty Research Award in 2017 and 2018, Swiss NSF Career Award Ambizione, AFOSR Faculty Fellowship in 2016 and 2017, and Best Paper awards at IEEE ICC 2017, ACM MMSys 2021, and ACM CFI 2019. He was a Technical Program Co-Chair of IEEE Packet Video 2012 and IEEE Packet Video AR 2016, Demo/Expo Chair of IEEE ICME 2016, and General Co-Chair of the IEEE SPS Seasonal School on Social Media Processing 2012. He was a Guest Editor of the Springer PPNA March 2015 special issue on P2P-Cloud Systems and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY January 2017 special issue on Mobile Visual Cloud. He is also an Associate Editor for the IEEE TRANSACTIONS ON MULTIMEDIA. He was the organizer of the first NSF Visioning Workshop on networked VR/AR communications in 2018. His research has been supported by the NSF, NIH, AFOSR, Adobe, Tencent Research, NVIDIA, and Microsoft. For further info, please visit www.jakov.org.

Mahmudur Khan received the Ph.D. degree in computer engineering from the University of Central Florida, Orlando, FL, USA, in 2018. He was a Postdoctoral Research Associate with the New Jersey Institute of Technology, Newark, NJ, USA, and The University of Alabama, Tuscaloosa, AL, USA. He is currently an Assistant Professor with the Department of Electrical & Computer Engineering and Computer Science, York College of Pennsylvania (YCP), York, PA, USA. His research interests include free-space optical communications, wireless ad hoc networks, and unmanned aerial vehicle communications.