Mobile Streaming of Live 360-Degree Videos

Omar Eltobgy, Omar Arafa, and Mohamed Hefeeda, *Senior Member, IEEE*

***Abstract*—Live streaming of immersive multimedia content, e.g., 360-degree videos, is getting popular due to the recent availability of commercial devices that support interacting with such content such as smartphones/tablets and head-mounted displays. Streaming live content to mobile users using individual connections (i.e., unicast) consumes substantial network resources and does not scale to large number of users. Multicast, on the other hand, offers a scalable solution but it introduces multiple challenges, including handling user interactivity, ensuring smooth quality, conserving the energy of mobile receivers, and achieving fairness among users. We propose a new solution for the problem of live multicast streaming of 360-degree videos to mobile users, which addresses the aforementioned challenges. The proposed solution, referred to as VRCast, is designed for cellular networks that support multicast, such as LTE. We show through trace-driven simulations that VRCast outperforms the closest algorithms in the literature by wide margins across several performance metrics. For example, compared to the state-of-the-art, VRCast improves the viewport quality by up to 2.5 dB. We have implemented VRCast in an LTE testbed to show its practicality. Our experimental results show that VRCast ensures smooth video quality and saves energy for mobile devices.**

***Index Terms*—Mobile multimedia, 360-degree video, adaptive streaming, multicast.**

I. Introduction

MOBILE data traffic has grown 18 folds over the past five years, and it is expected to continue growing at an annual rate of 47% in the coming few years [1]. The majority (60–80%) of the mobile data traffic carries video content. To partially cope with this substantial demand, cellular network operators have recently been considering multicast services for streaming *live* video sessions such as popular sports events and concerts. Unicast services cannot support large-scale live sessions, because the required radio resources grow linearly with the number of users, even when all users receive the same content at the same time. Multicast offers an efficient and scalable approach to stream live videos to many users. The current generation (4G) of cellular networks already has support for multicast services. For example, the evolved Multimedia Broadcast Multicast Service (eMBMS) is part of the LTE standard [2], [3]. Many cellular operators have deployed or experimenting with multicast services, including AT&T, Verizon, Korea Telecom, and China Unicom [4], [5]. Furthermore, smartphone manufacturers have started introducing eMBMS support in their products. For example, Google provides eMBMS support in Android 8.1 for Nexus and Pixel phones [6].

Prior works, e.g., [7]–[12], have proposed various optimizations for mobile multimedia services in terms of bandwidth, video quality, and mobile energy consumption. Most of these works, however, are either designed for traditional, single-view videos or consider the unicast model which does not support large-scale users. We consider mobile multicasting of 360-degree videos, which is a more complex problem than multicasting single-view videos. This is because 360-degree videos offer unprecedented interactivities between users and the content. Specifically, users watching a 360-degree video can dynamically change their viewing direction, resulting in an immersive and engaging experience, but creating challenges for the network that needs to support this interactivity. This is in addition to handling user mobility and varying channel conditions, as in traditional streaming systems.

Immersive multimedia content, including 360-degree and virtual/augmented reality (VR/AR) videos, is projected to be quite popular in the near future. As evidence of this expected popularity, recently, major companies such as Facebook and Google have been integrating support for such rich content in their platforms [13], [14]. Many manufacturers have introduced various consumer devices to render and interact with immersive content, such as HTC Vive, Oculus Rift, Samsung Gear VR, and the enhanced touch screens on recent mobile phones and tablets. Furthermore, mobile data traffic carrying immersive content is expected to grow 11 folds by 2021 compared to 2016 [1]. Therefore, the problem of efficiently delivering immersive multimedia content is of practical importance.

In this paper, we present a new solution (called VRCast) for the problem of *live* streaming of 360-degree videos to mobile users. VRCast supports user interactivity, optimizes the energy consumption for mobile receivers, accounts for the heterogeneous and dynamic nature of wireless channels, ensures the smoothness of the rendered 360-degree content, maintains fairness among mobile users, and achieves high spectral efficiency of the expensive wireless link.

The main contributions of this paper can be summarized as follows:

- New user grouping algorithm that optimally partitions mobile users into multicast groups and divides the available radio resources among these groups.

Manuscript received June 30, 2019; revised December 14, 2019 and January 15, 2020; accepted February 6, 2020. Date of publication February 14, 2020; date of current version November 18, 2020. This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. J. Wu. *(Corresponding author: Mohamed Hefeeda.)*

The authors are with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: oeltobgy@sfu.ca; oarafa@sfu.ca; mhefeeda@cs.sfu.ca).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2020.2973855

- New algorithm for optimally allocating the computed radio resources to different parts of the 360-degree video to maximize the perceived quality for mobile users.
- Simple method for minimizing the energy consumption of mobile users in multicast services.
- Trace-driven simulations which show that VRCast outperforms the state-of-the-art algorithms across multiple performance metrics. For example, VRCast improves the median frame quality by up to 22%, reduces the variation in the spatial quality by up to 53%, and improves the viewport quality by up to 2.5 dB compared to the closest work in the literature.
- Proof-of-concept implementation in an LTE testbed to demonstrate the practicality and efficiency of VRCast.

The remainder of this paper is organized as follows. Section II presents a brief background and defines the addressed problems. Section III presents the proposed solution. Section IV presents our simulations and comparisons against other works. Section V describes our LTE implementation. Section VI summarizes the related work and Section VII concludes the paper.

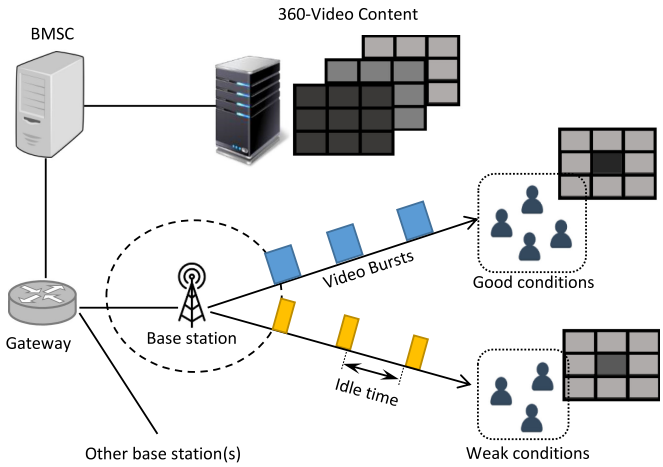


Fig. 1. A high-level illustration of the considered model. The proposed solution has three components. The first divides users into multicast groups and runs on the Gateway. The second assigns different qualities to tiles of 360-degree videos and runs on the BMSC. The third arranges the data into bursts to save energy and runs on the Gateway. Notice that users in the same group can be viewing different parts of the video.

II. Background and Problems Definitions

The objective of our work is to design an efficient system for live streaming of 360-degree videos of popular events, such as sports games, to large-scale mobile users using the *multicast* model. This is a fairly challenging task, as it involves multiple entities (e.g., users, various elements of the cellular network, and content servers). It also requires solving multiple problems at different layers to maximize the utilization of the wireless resources, while optimizing the quality of experience for users interacting with complex and bandwidth-demanding 360-degree videos on energy-constrained mobile devices.

In practice, a single live event can be concurrently multicast to thousands (or even millions) of mobile receivers. Thus, service providers typically provision wireless resources for individual events. Therefore, in this paper, we optimize the distribution of a single 360-degree video to many users, given the pre-allocated wireless resources to that video (i.e., given a specific bandwidth budget allocated for that video). Our solution can be used for multiple concurrent videos, but they are treated separately, which is the usual case in practice. We note that, although some further optimization could be achieved by jointly solving the delivery problem of multiple live events, this brings in many practical complications, including handling the: (i) diverse content types (e.g., various sports have different viewing patterns and all sports are quite different from live talk shows and political debates), (ii) different start/end times of the events, (iii) vast diversity in the popularity of events, and (iv) added computational complexity that may not allow solving the problem in real time.

In the following, we provide a brief overview of multicast in cellular networks, where we define the problems we address and the assumptions we make. To make our discussion clear, we will present examples from LTE (4 G) networks. Our work, however, is independent of the specific network technology and can be employed in future 5 G networks, provided that they support the simple architectural model described below.

Multicast in Cellular Networks: Current cellular networks support multicast services. For example, the eMBMS standard specifies the details of such services in LTE networks [2], [3]. Fig. 1 provides a high-level architecture for mobile multicast services, showing only the relevant entities to our problems. The multimedia content is injected from the content server to a Broadcast Multicast Service Center (BMSC), which manages multicast services in multiple cells in a large geographical region, e.g., a major city. The BMSC interacts with one or more Gateways (called eMBMS Gateway in LTE). Each Gateway controls one or more base stations, where each base station (referred to as eNodeB in LTE) controls the wireless channel to mobile users. The Gateway is responsible for the creation, termination, and management of multicast sessions. The BMSC is the entity that feeds the content data to the Gateway to forward on various multicast sessions.

Radio resources of the wireless channel are divided across time and frequency, typically using orthogonal frequency-division multiple access (OFDMA). The smallest unit of radio resources that can be allocated is referred to as a resource block (RB). In current LTE networks, each RB is 180 kHz wide in frequency and occupies 1 slot (0.5 ms) in time [15]. The channel conditions of each mobile user fluctuate over time. In current networks, mobile users periodically report their channel quality indicator (CQI) to the associated base stations. Each CQI value is mapped to a modulation and coding scheme (MCS), which determines the bitrate per resource block for each user. Higher MCS modes require good channel qualities and lead to higher bitrates.

To overcome the problem of weak wireless signals at cell edges and provide more efficient utilization of the wireless spectrum, recent standards introduced multicast services over a Single Frequency Network (SFN) [3]. A SFN is composed of multiple base stations, where an identical waveform is transmitted from all base stations with tight time synchronization. Using

SFN, the coverage area for a multicast service can be expanded, allowing more users to participate in the service while providing stronger signals to them from multiple base stations. In essence, a SFN provides an abstraction where multiple cells can be viewed as a large entity for multicast purposes. SFN also provides a seamless handoff process, since signals from neighbouring cells are identical and synchronized. Our mathematical modeling and solutions are general and they support multicast in SFNs.

While mobile networks standards provide frameworks for creating, signaling, and managing multicast services, they do not define algorithms for dividing mobile users into groups to accommodate their diverse and dynamic channel conditions. This is a complex problem with various trade offs. Consider, for example, a simple approach that puts all users in the same multicast group. In this case, users with poor channel conditions will impose a restriction on the MCS mode used for the whole group, leading to low quality for all users. On the other end, each user can be served with a unicast session to maximize the quality for that user. But this approach consumes significant resources and does not scale.

The first problem we address in this paper is how to efficiently compute the optimal number of multicast groups and the MCS mode for each group in order to maximize the average bitrate received by all users while maintaining fairness across users. Our solution for this problem, Section III-A, is to be implemented in the Gateway, which manages multicast sessions across multiple base stations serving thousands of users.

360-degree Videos: We consider interactive 360-degree videos served to mobile users who can utilize various devices such as smartphones, tablets, and head-mounted displays (HMDs). Users dynamically choose which parts of the video to watch, via for example, the tablet touchscreen or moving their heads when using HMDs. We refer to the part currently being viewed as *viewport*. The size of a 360-degree video is multiple times larger than regular videos, which by themselves require significant network resources. Nonetheless, at any moment only a small portion of the 360-degree frame is being watched, which is the viewport. Thus, the quality of the viewport is very important to users. However, users can change their viewports and expect immediate response, which poses a challenge for 360-degree video streaming systems. This challenge does not exist in single-view video streaming systems.

The second problem we address is how to optimally transmit various parts of the 360-degree video to the different multicast groups in order to optimize the perceived quality and support user interactivity, given the limited resources.

To address this problem, we consider general and recent trends in encoding and streaming videos, including tiling and client-based adaptive streaming. Specifically, in video tiling, each frame is divided into a grid of tiles, which provides flexibility and efficiency especially for streaming ultra high definition and 360-degree videos. For adaptive streaming, we consider the widely-used Dynamic Adaptive Streaming over HTTP (DASH) standard. In DASH, the video is temporally divided into segments, each is in the order of one to few seconds. Each tile in each segment is encoded at multiple quality representations. DASH provides mechanisms for storing, selecting, and serving different representations to support video adaptation. Tiling and quality representations of 360-degree videos are illustrated in Fig. 1, where different shades represent different quality levels. Information about segments and their bitrates are stored in Media Presentation Description (MPD) files, according to the Spatial Representation Description (SRD) feature of DASH [16].

Now our problem becomes determining which tiles to transmit and the DASH representation for each tile to maximize the quality for all users and maintain spatial smoothness across tiles. The solution for this problem, Section III-B, runs on the BMSC.

Saving Energy for Mobile Devices: Receiving 360-degree videos on battery-powered mobile devices consumes significant energy. We note that the network module consumes about 30% of the total energy of the mobile device during video streaming [17]. Thus, it is important to conserve the reception energy of mobile devices, which is the third problem we address. We propose a simple, yet optimal, solution to save the energy of mobile devices in Section III-C.

Problems we do NOT Address and Assumptions: A mobile 360-degree video streaming systems has many elements that we cannot address in a single paper. We do not address basic functions such as user authorization and low-level signaling to create multicast groups. All such functions are defined in standards documents.

The problem of selecting tiles and their qualities (our second problem) depends on estimating the relative popularity of different viewports, which we do not address in this paper. Multiple prior works have proposed viewport prediction algorithms, e.g., [9], [18]–[21]. We note, however, that our work only needs high-level relative popularity of viewports and not exact viewport prediction on a short time scale for each individual user. That is, our work does *not* predict/prefetch the next viewport for each user. Rather, our work needs aggregate information on the average number of users watching different areas of the video. This is a much easier information to estimate, especially that the the proposed 360-degree multicast framework is designed for large-scale events, where the nature of the content is known. For example, if the framework is used for multicasting a popular football or boxing game, good estimations on the expected popular viewports can be made (e.g., the ring in the boxing game). These estimations can be used throughout the entire streaming session as approximation of the relative popularity of viewports. Alternatively, these estimations can be dynamically refined using feedback from users through the Consumption Reporting procedure of eMBMS (see Sec III-D for a brief description).

Finally, our multicast framework always ensures that *all users receive all tiles* of the 360-degree video *with basic quality*. This is to support rapid viewport switching, which is essential to provide the immersive and interactive experience promised by panoramic 360-degree videos. Thus, even if a user makes a sudden move to a totally unpopular viewport in the video, the user will still be able to view the content, but with potentially lower quality. Notice again that our framework targets large-scale systems with thousands of users. Thus, the cost of sending the basic quality of all tiles is amortized across all users. Given the large number of users and the availability of the panoramic content, it is likely that some users will randomly explore different parts of

TABLE I
SYMBOLS USED IN THE PAPER

Symbol	Description
C	Number of MCS modes
M	Number of mobile users
Q	Number of quality representations
R	Number of resource blocks (RBs)
S	Number of time slots
T	Number of tiles
$b_{t,q}$	Bitrate of quality representation q for tile t
k	Number of multicast groups
$\mathbb{G}_g$	Set of users in multicast group g
c_i	MCS for user i (bits/RB)
$\hat{c}_g$	MCS for multicast group g (bits/RB)
w_t	Weight of tile t
x_g	RBs assigned to multicast group g
$y_{g,t,q}$	Indicator variable; equals 1 if quality q is assigned to tile t for group g, 0 otherwise

the video at various times, and thus the basic-quality tiles will not be wasted.

III. PROPOSED SOLUTION

In each of the following three subsections, we describe one of the problems addressed in this paper and present our solution for it. In Section III-D, we describe how the three parts fit together. An illustrative exampling applying all steps in a small scenario is presented in the Appendix, which is submitted as supplementary materials with this manuscript (due to space limitations).

Symbols used in this paper are listed in Table I. We use the following conventions. Constants are denoted by capital letters, e.g., C, and variables by small letters, e.g., x. A symbol like $\mathbb{G}$ denotes a set of elements.

A. Grouping Users

Problem Modeling: The first problem we address is dividing users into multicast groups and assigning wireless resource blocks to each group. This is a general problem and its solution can be used for multicast of 360-degree and regular videos. The problem can be stated as follows. Given a budget of R resource blocks and a number of mobile users M with different channel conditions (and hence different MCS modes denoted by $c_1, c_2, \ldots, c_M$), we would like to partition the users into the optimal number k^* of multicast groups $\mathbb{G}_1, \mathbb{G}_2, \ldots, \mathbb{G}_{k^*}$, where each user is assigned to one and only one multicast group. Users in the same group are all assigned the same MCS mode, which is the MCS mode of the user with the weakest channel condition in that group. In addition, we would like to assign the optimal number of resource blocks x_g^* to each multicast group $\mathbb{G}_g$, where $g = 1, 2, \ldots, k^*$ such that the average bitrate received by all users is maximized while at the same time achieving fairness among users in different multicast groups.

This is a complex optimization problem, with multiple interdependent variables, k^* and $x_1^*, x_2^*, \ldots, x_{k^*}^*$, for which we need to *concurrently* compute their optimal values. Furthermore, computing the optimal solution should be done efficiently in order to accommodate the dynamic nature of mobile users. To address this complexity without sacrificing the optimality of the solution, we carefully model the problem as a two-level nested optimization problem. Then, we develop an efficient dynamic programming algorithm to solve it.

Mathematical Formulation: We mathematically model the problem as follows:

$$\max_{\mathbb{G}_g} \sum_{g=1}^{k} \frac{|\mathbb{G}_g|}{M} \times \frac{\hat{c}_g \times x_g}{S} \tag{1a}$$

$$\text{subject to} \quad \hat{c}_g = \min_{i \in \mathbb{G}_g} c_i \qquad \forall g \leq k \tag{1b}$$

$$|\mathbb{G}_1 \cup \mathbb{G}_2 \cup \cdots \cup \mathbb{G}_k| = M \tag{1c}$$

$$\mathbb{G}_j \cap \mathbb{G}_l = \phi, \qquad \forall j, l \leq k, j \neq l \tag{1d}$$

$$x_g = \arg\max_{x_g} \left\{ \sum_{g=1}^{k} |\mathbb{G}_g| \log\left(\frac{\hat{c}_g \times x_g}{S}\right) : \sum_{g=1}^{k} x_g \leqslant R \right\} \tag{1e}$$

$$\text{variables} \quad k, x_g, \mathbb{G}_g \tag{1f}$$

The solution for the problem in Eq. (1) determines the optimal: (i) number of multicast groups k^*, (ii) division of all M users among the k^* groups $\mathbb{G}_1^*, \mathbb{G}_2^*, \ldots, \mathbb{G}_{k^*}^*$, and (iii) number of resource blocks assigned to each group $x_1^*, x_2^*, \ldots, x_{k^*}^*$.

The outer optimization computes the optimal grouping of users to maximize the average bitrate as shown in Eq. (1a). The bitrate for each user is the number of bits sent to that user divided by the total scheduling time, represented by the number of available time slots S. The number of bits sent equals the product of the MCS mode of the group $\hat{c}_g$ that the user belongs to and the number of RBs allocated to this group according to the inner optimization problem x_g. The constraint in Eq. (1b) guarantees that all users in a multicast group can receive all data by setting the MCS for this group according to the user with minimum MCS. The constraints in Eq. (1c) and Eq. (1d) ensure that each user belongs to only one multicast group.

The inner optimization, Eq. (1e), computes the optimal allocation of resource blocks to multicast groups. It uses the sum of the log of bitrates as the utility function in the optimization. This utility function maximizes the proportional fairness across users and makes it possible to compute the optimal solution analytically as shown in [22]. The optimal solution for the inner optimization is to distribute the resource blocks among groups proportional to the number of users in each group, that is $x_g^* = \frac{|\mathbb{G}_g|}{M} \times R, \;\; g = 1, 2, \ldots, k^*$.

Proposed Algorithm: We design an efficient algorithm using dynamic programming to compute the optimal solution; the pseudo code is shown in Algorithm 1. At a high level, we can think of the problem as placing partitions between users, and we want to decide the optimal number and positions of these partitions. For each k (total number of multicast groups), the best solution for g groups is calculated from the optimal solution for $g-1$ groups plus the utility of the new group. The dynamic programming array used $U_k(g, i)$ is the best utility to form g

Algorithm 1: Group Users

```
1  Function GroupUsers
       Input  : c_1, c_2, ..., c_M
       Output: k*, G*_g, x*_g
2      for k = 1 : C do
3          // Find the optimal g^th multicast group G_g
4          for g = 1 : k do
5              // Partition users with MCS between i and j
6              for i = 1 : C do
7                  for j = 1 : i − 1 do
8                      G_g = {∀u, j < c_u ≤ i}
9                      if U_k(g, i) < U_k(g − 1, j) + util(G_g)
                        then
10                         U_k(g, i) = U_k(g − 1, j) + util(G_g)
11                         P_k(g, i) = j
12                     end
13                 end
14             end
15         end
16         if U* < U_k(k, C) then
17             k* = k
18         end
19     end
20     // Backtrack using P_{k*} to find optimal G*_g and x*_g
21     j = C
22     for g = k* : 1 do
23         i = j
24         j = P_{k*}(g, i)
25         G*_g = {∀u, j < c_u ≤ i}
26         x*_g = (|G*_g| / M) × R
27     end
28     return k*, G*_g, x*_g
29 Function util (G_g)
30     x*_g = (|G_g| / M) × R // Optimal allocation of RBs
31     ĉ_g = min_{u∈G_g} c_u
32     return u_g = (|G_g| / M) × (ĉ_g × x*_g / S)
```

groups from users with MCS at most i ($c_u \leq i$):

$$U_k(g,i) = \max_{\forall j<i} U_k(g-1,j) + utility(\{\forall u, j < c_u \leq i\}).$$

In order to find the maximum $U_k(g,i)$, we search among the optimal utilities $U_k(g-1,j)$ for $g-1$ groups from users with all possible MCS values j where $j < i$ (lines 6–14). To calculate the new utility, we add $U_k(g-1,j)$ to the utility of the new group number g which consists of users with MCS between j and i (line 10). The utility of any group G_g is computed in lines 29–32 according to Eq. (1a). For each $U_k(g,i)$, we keep a reference to the best j that gives the maximum utility in the parent array $P_k(g,i)$ (line 11) so that we can backtrack and reconstruct the optimal $\mathbb{G}_g^*$ and x_g^* (lines 20–27).

Optimality and Time Complexity: The following theorem proves the optimality of the grouping algorithm.

Theorem 1 (Optimality): The GroupUsers algorithm (Algorithm 1) divides users into the optimal number of multicast groups and computes the optimal number of resource blocks for each group in order to maximize the average bitrate received by all users.

Proof: We prove this theorem by induction. The base case is to find the optimal solution for the first group of users with MCS mode at most i. There is only one solution in this case, which is putting all users in the same group. For the induction step, the algorithm computes the optimal solution for g groups from users with MCS mode at most i, assuming that the algorithm has already computed the optimal solutions for groups of size less than g and MCS less than i. The algorithm considers every possible group that can be formed by grouping users with all MCS ranges ending with MCS i. Then, it chooses the group number g that gives the maximum g-group utility. The g-group utility depends on the utility of the group number g plus the corresponding $(g-1)$-group utility. The algorithm computes the optimal utility for every possible k searching for k^* that gives the best utility. To calculate the utility for the group number g, the algorithm assigns a number of resource blocks x_g^* proportional to the number of users in the group. This proportional assignment maximizes the fairness which is proved in [23] using Lagrangian multipliers. ■

Time Complexity: The time complexity of the GroupUsers algorithm is $O(C^4)$, where C is a *constant* that represents the maximum number of MCS modes. This is because the algorithm has four nested loops, each has a maximum (conservative) range of C. We note that, in current LTE networks, the maximum possible value of C is 30 [3], while in practice the feasible number of MCS modes is much less than that, especially for multicast networks. In addition, the proposed solution targets large-scale multi-cell multicast networks, in which multiple cells typically form a Single Frequency Network (SFN), as described in Section II. In SFNs, signals from neighbouring cells are designed to be constructively added near cell edges. Thus, as a mobile user moves towards the edge of one cell, it starts receiving and adding signals from neighbouring cells such that the total signal strength does not drop significantly. This means that the MCS mode of the mobile user may not widely change as it moves within the SFN. Furthermore, all steps performed by the algorithm are simple scalar operations. Thus, the proposed algorithm can easily be deployed in real systems.

Deployment and Frequency of Invocation: The GroupUsers algorithm runs on the Gateway in Fig. 1. It is invoked when a new user joins to place that user in the correct group. The algorithm may need to be invoked when an existing user leaves, because the current grouping may become sub-optimal. The low time complexity of the algorithm allows it to be frequently invoked, on a sub-second scale if needed. Thus, the algorithm can handle the dynamic nature of the mobile streaming environment.

Cost of Changing Groups: Changing the multicast group that a user is assigned to does not impose any significant cost. This is because the channel assignment is periodically transmitted in the signalling of the MAC layer. For the user, it is a matter of selecting the appropriate channel to receive the data on in the following period [24].

B. Selecting Quality Representations for Tiles

The second problem we address is how to allocate the resource blocks computed in the previous problem for each multicast group to tiles transmitted to that group. We solve the problem independently for each multicast group $\mathbb{G}_g^*$ with MCS mode $\hat{c}_g$ and x_g^* resource blocks assigned to it. Therefore, for simplicity, we remove the group subscript g in this subsection.

Problem Modeling and Mathematical Formulation: The goal is to select a quality representation q for each tile t to maximize the average viewport quality and minimize the spatial variance between viewport tiles of each user while maintaining fairness among users with different viewports. Recall that all tiles of the 360-degree video being streamed are encoded at multiple quality levels according to the DASH protocol. And that all multicast groups receive all tiles at the lowest quality at all times to support fast viewport switching. The problem here is to decide which tiles should be transmitted at higher qualities than the basic quality. This is done based on the importance, or weight, of each tile. The weight w_t of a tile t represents the fraction of users watching that tile as part of their viewports. As we discussed in Section II, the relative popularity of viewports, and hence the tile weights, are inputs to our problem.

The mathematical formulation for the quality selection problem is given in Eq. (2), which computes the optimal quality representation for each tile.

$$\max_{y_{t,q}} \sum_{t=1}^{T} \sum_{q=1}^{Q} w_t \times \log(b_{t,q}) \times y_{t,q} \tag{2a}$$

$$\text{subject to:} \sum_{t=1}^{T} \sum_{q=1}^{Q} \left\lceil \frac{b_{t,q}}{\hat{c}} \right\rceil \times y_{t,q} \leqslant x^* \tag{2b}$$

$$\sum_{q=1}^{Q} y_{t,q} = 1, \forall t \tag{2c}$$

$$y_{t,q} \in \{0,1\}, \forall t, q \tag{2d}$$

$$\text{variables} \quad y_{t,q} \tag{2e}$$

In the formulation, we use the weighted product of tiles bitrates (WPTB) as our utility. WPTB is defined as $\prod_t b_{t,q}^{w_t}$, which is equivalent to $\sum_t w_t \times \log(b_{t,q})$. WPTB improves the spatial smoothness because maximizing the product of tiles bitrates leads to a balanced solution with small variance unlike the the weighted sum utility [25]. For example, if we have two tiles with equal weights, the best weighted product utility is achieved by dividing RBs between them equally (balanced). While the best weighted sum product utility can lead to assigning all RBs to one tile and nothing to the other (unbalanced). The constraint in Eq. (2b) restricts the available number of resource blocks to x^*. The constraints in Eq. (2d) and Eq. (2e) ensure that only one quality representation is assigned to each tile. Note that $y_{t,q}$ is a decision variable that determines whether the quality q is assigned to tile t.

Proposed Algorithm: We design a dynamic programming algorithm, shown in Algorithm 2, to solve the optimization problem in Eq. (2). The algorithm first assigns a minimum quality representation to each tile (line 2), since all tiles need to be transmitted to each multicast group. It then searches across all possible assignments of quality to tiles using dynamic programming. The best solution for t tiles is computed from the optimal solution for $t-1$ tiles plus the utility of the new tile. The dynamic programming array used $V(t,\tau)$ is the best utility for the first t tiles using τ RBs such that

$$V(t,\tau) = \max_{\forall q} V(t-1, \tau - R(t,q)) + utility(t,q),$$

where $R(t,q) = \frac{b_{t,q}}{\hat{c}}$ is the number of resource blocks needed to stream tile t with quality q.

In order to find the maximum $V(t,\tau)$, the algorithm searches among the optimal utilities for $t-1$ tiles after assigning each feasible quality q to tile t and reducing the required RBs for this assignment (lines 6–15). To calculate the new utility, the algorithm adds the utility of selecting every feasible quality q for tile t to $V(t-1,\tau')$, where τ' is the available RBs after subtracting the RBs needed for this tile quality $\tau' = \tau - \frac{b_{t,q}}{\hat{c}}$ (line 9). The utility of choosing quality q to tile t is computed in line 8 according to Eq. (2a). For each $V(t,\tau)$, the algorithm keeps a reference to the best $b_{t,q}$ that maximizes the utility in the parent array $P(t,\tau)$ (line 12) so that it can backtrack and reconstruct the optimal b_t^* (lines 18–23).

Optimality: The following theorem proves the optimality of the tile quality selection algorithm for each group.

Theorem 2: The SelectTileQuality algorithm (Algorithm 2) computes the optimal quality representation for each tile in the

Algorithm 2: Select Tile Quality

```
1  Function SelectTileQuality
       Output: b_t^* ← Selected bitrate for each tile.
2      ∀t, b_t^* = b_{t,1} // Assign min bitrates to tiles
3      ∀τ, V(0,τ) = 0
4      for t = 1 : T do
5          for τ = b_{t,1}/ĉ : x^* do
6              for q = 1 : Q do
7                  if τ − b_{t,q}/ĉ ≥ 0 then
8                      u_{t,q} = w_t × log(b_{t,q})
9                      u = V(t − 1, τ − b_{t,q}/ĉ) + u_{t,q}
10                     if u > V(t,τ) then
11                         V(t,τ) = u
12                         P(t,τ) = b_{t,q}
13                     end
14                 end
15             end
16         end
17     end
18     // Backtrack to find optimal solution b_t^* using P
19     τ = x^*
20     for t = T : 1 do
21         b_t^* = P(t,τ)
22         τ = τ − P(t,τ)/ĉ
23     end
24     return b_t^*
```

video such that the total number of resource blocks assigned to the multicast group is not exceeded.

Proof: We use induction to prove this theorem. The base case is to find the optimal solution for one tile given a budget of τ RBs. The algorithm searches among all feasible quality assignments for this tile and chooses the one with the maximum quality. A feasible quality assignment means that there are enough RBs to stream it. Then, for the induction step, the algorithm computes the optimal solution for tile number t given τ RBs assuming it already has the optimal solutions for all tiles less than t and RBs less than τ. The algorithm considers every feasible quality assignments for this tile such that it has enough RBs to send. Then, it chooses the quality that gives the maximum utility. The t-tile utility depends on the utility for tile number t after choosing a quality and the corresponding $(t-1)$-tile utility after subtracting the RBs required for the chosen quality. ■

Time Complexity: The time complexity of the SelectTileQuality algorithm is $O(T \times \sum_{g=1}^{k^*} x_g^* \times Q)$, where T is the number of tiles, Q is the number of quality representations, and x_g^* is the number of available resource blocks for multicast group g. This is because the algorithm has three nested loops with ranges T, x^*, and Q respectively, and it runs for every multicast group g.

Deployment and Frequency of Invocation: We first shed some light on the practical ranges of the three parameters T, Q, and x_g^* that affect the time complexity of the SelectTileQuality algorithm. The number of tiles T is in the order of few tens. For example, recent works that adopt tiling, e.g., [18], [19], [26], set T around 50. The number of DASH quality representations Q is typically less than 10 in practice. The total number of resource blocks for *all* multicast groups ($\sum_{g=1}^{k^*} x_g^*$) in one second (the typical scheduling window) is in the order of thousands [3], [27].

The SelectTileQuality algorithm runs on the BMSC (Fig. 1) and it is invoked on every DASH segment of the video (in the order of 1 to few seconds). Given the numbers above and noting that all steps include only simple operations, the algorithm can easily be deployed in real systems.

C. Energy Minimization for Mobile Users

The third problem we address is how to minimize the *reception* energy for mobile users.

Burst Transmission: Our solution for the reception energy minimization problem is to make the base station transmit the video data (the tiles) in *bursts*. This allows mobile devices to wake up the reception component to receive the data during the burst and put it in low-energy (idle/sleep) mode when there is no burst. The energy saving problem becomes determining the start and end of each burst as well as assigning bursts to resource blocks. Recall that the wireless resource blocks span two dimensions: frequency and time. Naively choosing resource blocks for bursts may spread them over longer time than necessary, and thus reduces the opportunity for energy saving for mobile devices.

We propose a simple method to allocate bursts to resource blocks. First, note that our user grouping algorithm (Section III-A) does not share resource blocks across multicast groups; each resource block is allocated to only one group. Thus, to maximize the idle time for the reception component (i.e., minimize energy consumption), we arrange the resource blocks of each multicast group contiguously, i.e., no resource blocks from a multicast group is allocated in the middle of the resource blocks of another group. Then, we set the start time of the burst as the beginning of the earliest resource block and the end time as the end of the last resource block.

In LTE networks, burst transmission is implemented by setting the the Discontinuous Reception Mechanism (DRX) parameter [28], which instructs the reception component of mobile devices to wake up only during the transmission period of their group's resource blocks and sleep otherwise. The energy optimization method is to be implemented in the Gateway (Fig. 1).

D. Putting All Pieces Together: VRCast

In the previous subsections, we presented our solutions for three important problems in mobile multicast systems designed for 360-degree videos. In this subsection, we discuss, at high level, how the whole system (denoted by VRCast) works using the eMBMS specifications of LTE [29] and the DASH streaming model over mobile networks [30].

End-to-End Pipeline: At the content provider, the 360-degree video is divided into tiles and encoded at multiple quality representations, which are specified in the MPD file. The content provider then supplies the MPD file to the BMSC over HTTP.

The GroupUsers algorithm, running on the Gateway, computes the optimal solution for dividing users into multicast groups and assigning resource blocks to each group. This solution is passed to the BSMC (over a control connection), which runs our SelectTileQuality algorithm to select the optimal quality representation for each tile in each group. The BMSC then uses DASH to download the selected qualities from the content provider. After that, the video data is transferred from the BMSC to the Gateway, which puts them in bursts to save energy. The Gateway forwards the bursts to each multicast group using the File Delivery over Unidirectional Transport (FLUTE) protocol [30], [31].

Mobile users in a geographical area (e.g., in/around a stadium or within a city) interested in receiving a live 360-degree video run an eMBMS-compatible client to discover and register with the system. Each mobile user joins a specific multicast group according to his/her channel condition and the output of the GroupUsers algorithm. Then, they receive the video data in bursts from the BMSC using FLUTE.

Handling Various Dynamics: We note that the presented solution supports the dynamic nature of the wireless channels and interactivity provided by the 360-degree content. The dynamics of wireless channels are handled through periodically (order of seconds) invoking the GroupUsers algorithm.

User regrouping can trigger changing the quality assignment of different tiles. The tile quality is also impacted by the relative weights of tiles. The eMBMS standard provides a mechanism that can help in estimating the relative weights of tiles, which is the Consumption Reporting procedure. This procedure allows mobile clients to send consumption reports to the BMSC. Since

these are XML files, they can easily be extended to include information about the viewports watched by the users in the previous period. This can help in estimating the relative popularity of viewports.

Supporting Multiple Cells: As mentioned in Section II, the proposed modeling and solution support multi-cell networks that utilize Single Frequency Networks (SFNs). In a SFN, multiple neighbouring base stations work on the same frequency and they are managed by a single Gateway and BMSC. Our SelectTileQuality algorithm runs on the BMSC, which provides the tiled video content to all base stations through the Gateway (Refer to Fig. 1). Our GroupUsers algorithm runs on the Gateway and arranges users across all cells in multicast groups. Similarly, our burst transmission algorithm runs on the Gateway and constructs bursts for all cells. Recall that SFN provides an abstract network across all participating cells.

Handling Packet Losses: Our work considers a large-scale one-to-many multicast model, where a 360-degree video data is delivered to thousands of users in real time. In this model, retransmission is not possible. To mitigate potential packet losses, multicast systems typically use error concealment methods and/or add redundant data using error correcting codes. These are all established methods and they can easily be integrated with our platform. Specifically, an error concealment method can be implemented on the receiver (player) side, which is independent of the delivery mechanism. A Forward Error Correction (FEC) function can be added to the BMSC to apply it on the video data before sending the data to the Gateway. The error concealment and correction functions are orthogonal to our work and we do not consider them in our evaluations.

No Additional Delay: An important feature of the proposed components of VRCast is that they do not introduce any additional delay in the video delivery process, which is important for live streaming. For example, the SelectTileQuality algorithm at the BMSC selects the quality representations for different tiles in real time for each scheduling window Δ (in the order of 1–2 sec). Meaning, while users are consuming data for the window at time t_1, the decisions for the window at $t_1 + \Delta$ are computed. Since our solution is computationally efficient (takes milliseconds to compute, which is much smaller than the window size), it does not introduce any additional delay into the live streaming session. A similar argument applies to the GroupUsers algorithm: the decision to change groups is done for future time windows while users are consuming data for the current window.

IV. Evaluation Using Trace-Driven Simulation

This section presents trace-driven simulations to compare the proposed approach against the closest works in the literature. We refer to our approach as VRCast in the figures and discussion. VRCast includes the solutions of the three problems considered in this paper: user grouping, tile quality selection, and energy saving.

We compare VRCast against the closest algorithm in the literature, which is called Multicast for Virtual Reality (MVR) [32]. MVR solves the problem of 360-degree mobile video multicast, and it was shown to outperform the previous approaches in the literature. MVR uses a greedy algorithm to partition users into groups and selects tile qualities of 360-degree videos. In addition, we compare against another state-of-the-art grouping algorithm, which is called Proportional Fairness (PF) [22]. PF handles the heterogeneity of channel conditions by partitioning users into multicast groups so that users with good signal strength do not suffer by being grouped together with users with poor signal strength. PF, however, was not designed for tiled streaming. We slightly modified it to support tiling, by uniformly distributing the computed bitrate for each frame across all tiles in that frame.

A. Simulation Setup

360-degree Videos and User Interactivity Traces: We used two publicly-available datasets of 360-degree videos in our experiments to cover a wide variety of content and user viewing behavior. The first dataset [20] contains 16 videos at 4 K resolution; we refer to these videos as V1 to V16. The length of each video is around 30 seconds. The dataset contains different video categories including sports, landscape, and entertainment. Each of the 16 videos was watched multiple times by 153 volunteers, resulting in a total of 985 recording sessions. In each session, the view angle of the user is reported every 0.1 second, in the form of Euler angles.

The second dataset [33] contains 7 videos at 4 K resolution, which we refer to as V17 to V23. The length of each video is around 60 seconds. The dataset contains different video categories including sports, entertainment, and documentary. These videos were watched by 59 participants in a total of 350 sessions. The view angles were recorded in the form of Hamilton quaternions. We converted the Hamilton quaternions to Euler angles. The recording of view angles in this dataset was not done at a constant interval. We used linear interpolation to have uniform samples every 0.1 seconds, as in the first dataset.

In total, we created unified traces with 1,335 sessions of 23 diverse 360-degree videos, where each video is watched on average by 50 users and the viewport is reported every 0.1 second. To exercise different network conditions and user mobility with realistic speeds, we generated longer traces of length 15 minutes by concatenating sessions from the 23 videos together.

We divided each video in the traces into an 8x4 grid of tiles, similar to [32], and encoded each tile at five quality representations using Kvazaar [34], an open source HEVC video encoder that supports tiling. We used variable bitrate encoding (VBR) with different quantization parameters (QP) of {18, 24, 30, 36, 42}. Then, we used GPAC [16] to segment the video into one-second segments and generate the MPD DASH manifest.

LTE Network Configuration and User Mobility: We simulate a mobile network using the LTE module in the NS3 simulator. There are 300 mobile users randomly distributed over an area of 10×10 km. Users move according to the Self-similar Least Action Walk (SLAW) mobility model [35], which is more realistic than the random-way point model. It represents mobility of users within a community, such as students on a university campus and visitors of a theme park. We generate mobility traces

of length 15 minutes each, using BonnMotion [36], which is a Java-based tool commonly used for investigation of mobile network characteristics. We configure the tool to generate mobility traces with different speeds to represent cases where some users are walking and others are riding buses or cars. We simulate users with moving speed of 1 m/sec (walking) and others with speed up to 10 m/s (riding a bus). We import the generated mobility traces into NS3 to control the movements of mobile users. The movements of users affect their channel conditions, and hence the MCS modes used and the bitrates they can receive.

Performance Metrics. We consider the following metrics, which are important for multicast streaming systems and have been used in previous works, e.g., [8], [18], [26], [32].

- **Frame Quality:** the quality of the whole frame, measured by the total bitrate allocated to each multicast group containing all tiles.
- **Viewport Quality:** the quality of the viewport perceived by each user, measured by two metrics that capture the video quality: (i) peak signal to noise ratio (PSNR in dB), and (ii) structural similarity index metric (SSIM) metric. Both metrics are computed for the tiles in the viewport.
- **Spatial Variance:** the variance in the quality of tiles in the viewport, which captures the smoothness of the rendered video content to users. A recent work [18] showed that the spatial variance has a direct effect on QoE.
- **Reception Energy Saving:** the fraction of time that the reception component of a mobile device is put in low-energy mode (i.e., sleep).
- **Fairness:** the relative quality observed by each user compared to others. It is defined as the Jain's index of the ratio between each user viewport bitrate and the total frame bitrate.
- **Spectral Efficiency:** the total transmitted data rate (in bits per second) divided by the channel bandwidth (in Hertz). This metric shows the efficiency of the streaming system in using the cellular network resources, which is an important aspect for network operators.
- **Scalability:** the total execution time of the algorithm with different number of users.

We also analyze the buffering behavior of various clients and how their viewport quality changes with time as users change their viewports.

Experiments: In every simulation experiment, mobile users move in the cellular network according to the generated mobility traces and their channel conditions change accordingly. Each user randomly selects one of the interactivity traces and interacts with the video (i.e., changes viewports) according to that trace. The user grouping algorithm runs periodically every 1 sec. The tile quality selection algorithm runs for each video segment; the length of the segment is set to 1 sec. We note that both the user grouping and tile quality selection algorithms do not have access or use the video traces. Rather, the traces are used only by the mobile users to mimic user interactions according the timestamps in the traces, as in real life. For comparisons, we run each experiment three times: once with VRCast, another with MVR, and the third with PF. Then, each experiment with each multicast approach is repeated 30 times with different random seeds. We plot and analyze the average results across all 30 repetitions.

B. Comparison Against State-of-the-Art

We compute and plot the cumulative distribution function (CDF) for each performance metric across all users and all video traces. Figs. 2 and 3 summarize some of our comparison results for all metrics. The figures show that VRCast substantially outperforms MVR and PF. Specifically, Figs. 2(a) and 2(b) show that the PSNR and SSIM quality metrics for the viewports generated by VRCast are better than MVR and PF. For example, as shown in Fig. 2(a), VRCast improves the viewport PSNR quality by at least 1 dB compared to MVR for about 60% of all viewports, and more importantly these viewports are the ones with quality less than 44 dB (left part of the figure). This is important because improvements in this range are visually perceived by users. Furthermore, for the lower 20% of the viewports (less than 38 dB), the improvements are up to 2.5 dB, which is substantial (dB uses log scale). The improvements of VRCast compared to PF are even higher, where up to 4.5 dB can be observed in Fig. 2(a).

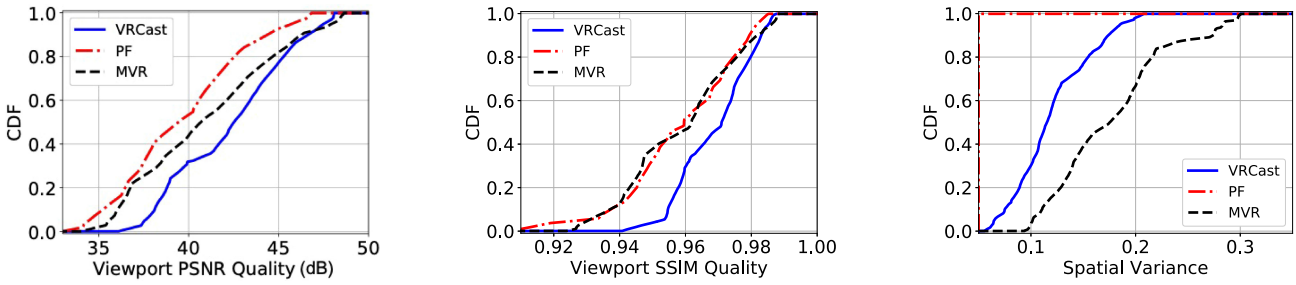
In addition, as shown in Fig. 2(c), VRCast distributes the allocated bitrate of each frame to viewport tiles not only to improve the viewport quality but also to achieve smooth quality across all tiles. The smooth quality is shown by the much lower spatial variance achieved by VRCast compared to MVR. The figure shows a reduction in the median of the spatial variance by up to 53% compared to MVR. We note that PF assigns all tiles the same bitrate and thus there is no variance in quality. PF, however, yields poor quality for the viewports (Fig. 2(a)).

Fig. 3(a) shows that VRCast results in much higher quality for all video frames than MVR and PF. For example, for VRCast, about 25% of the frames are assigned a bitrate of 7.7 Mbps or higher, while none of the frames is assigned that bitrate for MVR and PF. The median frame quality for MVR is 5.97 Mbps, while it is 7.25 Mbps for VRCast, which is an improvement of 22%.

As shown in Fig. 3(b), VRCast achieves substantial improvements in the reception energy saving compared to the other algorithms. For example, using MVR and PF, no mobile user was able to achieve energy saving more than 20% (i.e., turn off the receiving components in the mobile device 20% of the time). Whereas using VRCast, more than 80% of the users achieved at least 20% energy saving.

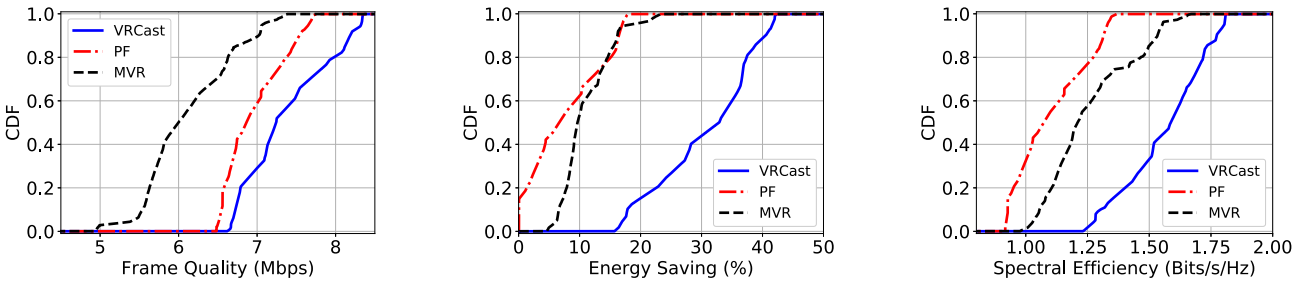
An important metric for cellular network operators is the spectral efficiency, which indicates how the (expensive) spectrum of the wireless channel is utilized to carry bits. Fig. 3(c) shows that VRCast is much more efficient in utilizing the wireless spectrum than the other algorithms. As an example, VRCast achieves a spectral efficiency of at least 1.6 bits/s/Hz for 40% of the time, while PF and MVR almost never achieve this efficiency.

Maintaining fairness among different users is another desirable feature of VRCast. As illustrated in Fig. 4, VRCast does not sacrifice the fairness among users to achieve the quality improvements shown in Fig. 3(a). The fairness index of VRCast is fairly high and close to 1. For example, Fig. 4 shows that



(a) Viewport PSNR Quality (b) Viewport SSIM Quality (c) Spatial Variance

Fig. 2. Performance of VRCast against other algorithms.



(a) Frame Quality (b) Energy Saving (c) Spectral Efficiency

Fig. 3. Performance of VRCast against other algorithms (continued).

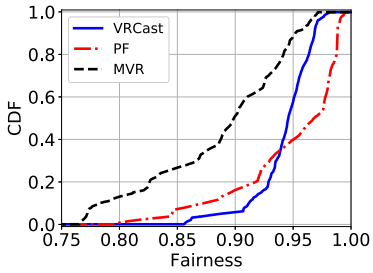
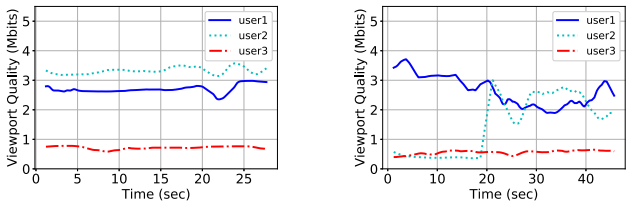


Fig. 4. Fairness of VRCast.



(a) Boxing Match (b) Diving

Fig. 5. Sample results of VRCast with different videos.

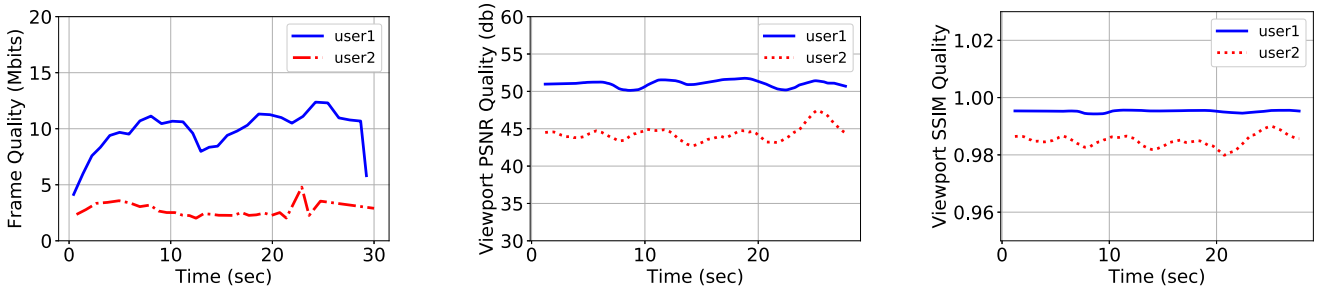
VRCast achieves a fairness index of more than 0.95 for at least 50% of the users, which is much higher than the fairness index achieved by MVR. PF achieves a similar fairness index, but it yields much less video quality than VRCast.

C. Analysis of VRCast

We take a closer look at the performance of VRCast. We analyze the viewport quality of users across time. We choose 3 users with high (user 1), medium (user 2), and poor (user 3) channel conditions. We select three videos: boxing match, diving scene, and football training. We analyze the viewport bitrate while these users watch and interact with these videos. This experiment is to show that VRCast efficiently adapts to dynamic changes in channel conditions and viewports.

Due to space limitations, we only present the results for two videos: boxing and diving. Fig. 5(a) shows the viewport bitrate across time for the boxing video. User 1 and user 2 are in the same multicast group with high MCS while user 3 is in another group with low MCS. The viewport bitrate of each user does not change significantly across time, because in the boxing match most users watch the same region of interest (the boxing ring). As a result, each user experiences almost the same viewport quality across time (i.e., low temporal variance). Although users 1 and 2 are in the same group, user 2 receives higher viewport bitrate because s/he is watching more popular tiles. During the time between 20 and 25, user 1 changes his/her viewport to less popular tiles (the audience) and receives a lower quality.

Fig. 5(b) shows the viewport bitrate across time for the diving video. Due to the fast mobility of user 2, their channel conditions change during watching the video. In the first 15 seconds, user 2 has relatively poor channel condition and is grouped with user 3. After that, the channel condition of user 2 improves, thus is grouped with user 1. The viewport bitrate of each user changes substantially across time, because in the diving scene, the viewports of users are distributed on the whole video frame without a specific region of interest. Therefore, the weights of tiles change from time to time leading to changes in the viewport bitrate.



(a) Frame Quality (b) Viewport PSNR Quality (c) Viewport SSIM Quality

Fig. 6. Performance of VRCast in the LTE testbed.

V. Evaluation in LTE Testbed

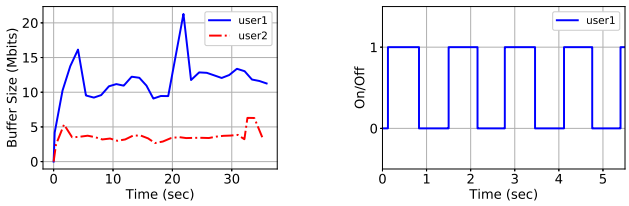
A. Testbed Implementation

Building LTE testbeds for *multicast* services is quite challenging as there are many hardware and software components that need to be developed and integrated correctly to get the testbed running. We summarize our experience in building an LTE testbed in the following.

The testbed is similar to Fig. 1 and it has base station, mobile devices, and BMSC server. The hardware of the base station is implemented using the Ettus USRP B210 software defined radio to transmit/receive data to/from mobile devices. The software of the base station is the Amarisoft LTE stack [37], which implements the functions of the eNodeB, Evolved Packet Core (EPC), and eMBMS Gateway according to the 3GPP specifications. We configured the eMBMS gateway to have multiple physical multicast channels (PMCH), one channel for each multicast group. Each channel has one multicast service that is assigned a specific IP address. The components of VRCast are implemented and integrated with the base station. The decisions from VRCast are mapped to various configuration parameters of the Amarisoft LTE software such as the number of allocated subframes for each service and the minimum MCS for each PMCH.

The testbed has two phones. We placed one of them close to the base station and the other a few meters away (the range of the base station is small as we did not use power amplifiers). The close phone had an MCS mode of 23 and the far away one had MCS mode of 12. The phones run the Expway eMBMS middleware [38], which allows them to discover multicast services and connect to them. Each phone represents a user. We make each phone run one of the user interactivity traces described in Section IV-A.

Clearly, we cannot get enough phones in our lab to fully analyze a mobile multicast service. To *partially* mitigate this issue and consider the effect of user mobility, we integrate simulated users in the testbed. Specifically, in addition to the two real users, we include 298 simulated mobile users with different channel conditions and user interactivities. The information about all users (real and simulated) is periodically given to VRCast, which dynamically divides users into groups, decides on the quality for individual tiles, and determines the start and end times for bursts. Thus, although we have only two real users, the network situation is continuously changing because of the simulated users dynamics (mobility and interactivity with the video). The results and analysis in this section are only from the two real phones.



(a) Buffer Size (b) Reception Energy

Fig. 7. Performance of VRCast in the LTE testbed.

B. Empirical Results From Testbed

The testbed is a proof-of-concept to show the practicality and correctness of VRCast. To this end, we measured the actual frame quality and viewport quality received on the phones. We plot the results in Fig. 6. User 1 has good channel conditions, and thus experiences better frame and viewport qualities while user 2 has poor channel conditions and gets lower qualities. Figs. 6(b) and 6(c) show that the smoothness of the viewport quality is maintained over time, although users do change their viewports according to the interactivity traces.

Next, we analyze the buffer level at the two phones as the streaming session progresses. The maximum buffer size was set at 50 Mbits. Fig. 7(a) plots the buffer level as the time progresses, which shows that the playback of the video was smooth without any stalls or rebuffering events because the buffer always has data. Also the buffer level never exceeded the maximum value.

Finally, we analyze the burst transmission behavior of VRCast. At each mobile user, we record the start and end of the data reception (burst). We plot the results for user 1 in Fig. 7(b) for a 5-sec period. The figure shows that the data of the 360-degree video was sent in bursts allowing mobile receivers to turn off the reception circuits to save energy, while the quality and smoothness of the video are maintained as shown in Fig. 6.

VI. Related Work

Mobile video streaming of traditional single-view video streams has been extensively studied in the literature, for both

unicast and multicast. For example, Almowuena *et al.* [8] divide users into groups based on their channel conditions. Chen *et al.* [22] discuss fairness among users in mutlicast groups and present a resource allocation method for user grouping. Chen *et al.* [12] introduce a unicast framework for adaptive streaming of single-view videos over LTE networks. Yu *et al.* [39], [40] utilize scalable video coding (SVC) to optimize the energy consumption of mobile devices. Lentisco *et al.* [11] propose a method to reduce the latency in mobile multimedia networks by limiting the buffer size at the clients. Xie *et al.* [41], [42] leverage physical layer information to improve the estimation of available bandwidth in LTE networks, which is used to facilitate video rate adaptation [41] or improve web loading latency [42]. Tan *et al.* [10] propose utilizing network slicing and Network Function Virtualization (NFV) in 5 G networks to support ultra high definition (UHD) video broadcast/multicast services. Finally, Chen *et al.* [7] propose a forward error correction method that uses unequal error protection (UEP) and consider the dependency among different blocks in the frame.

None of the above works, however, considers characteristics of 360-degree videos, such as user interactions with the content.

Recent works addressed various aspects of 360-degree video streaming under the unicast model, e.g., [18], [19], [43], [44]. Two common themes can be identified in these works: tiling and region-of-interest (ROI) streaming. In tile-based streaming, a 360-degree video is projected on an equirectangular map and divided into a grid of tiles. Tiles in the viewport of the user are streamed with high quality and other tiles are either streamed with lower quality or not sent at all. Tiling has been used in multiple works, including [18], [19]. The recent High Efficiency Video Coding (HEVC) standard supports tiling [45], which further helps in adopting tile-based streaming. Qian *et al.* [46] present a system for unicast streaming of 360-degree videos to mobile users. He *et al.* [47] propose a 360-degree video streaming framework that enables mobile devices to decode and display high quality tiles in real time. Graf *et al.* [26] discuss streaming of 360-degree videos over HTTP by exploring different tiling patterns. We use the analysis in [26] to select tile sizes in our work. Petrangeli *et al.* [19] design a 360-degree streaming framework over HTTP/2. Xiao *et al.* [48] compute the optimal tiling to minimize storage and bandwidth. Nasrabadi *et al.* [49] investigate a buffer-efficient approach using scalable video coding in 360-degree video streaming and provide a comprehensive buffer analysis. Xie *et al.* [18] propose a probabilistic tile-based adaptive streaming system.

In ROI-based streaming, a 360-degree video is projected onto a geometric structure such as a pyramid or cube map, where regions of the structure represent different user viewports. For each viewport, a video version is created where more bits are allocated to parts of the 360-degree video in the viewport and less bits are allocated to the other parts. Versions are stored on the server. During streaming, the version that aligns the most with the user's current viewport is served to that user. Zhou *et al*. [43] analyze the ROI-based streaming approach used by Facebook and its impact on user experience and bandwidth consumption. Corbillon *et al.* [44] investigate the effect of various projections and quality arrangements on the video quality displayed to the user.

All of the above works consider unicast streaming, where each user is served with a separate connection. For popular live streaming events, unicast is not efficient.

Multicast of 360-degree videos has received less attention in the literature. We compare against the recent algorithm in [32], referred to as MVR (Multicast of Virtual Reality) content, which was shown to outperform others. We also compare VRCast against the closest work for single-view video streaming [22] after adding some extensions to support 360-degree videos. The algorithm in [22] divides users into multicast groups to maximize the proportional fairness.

VII. Conclusions and Future Work

Multicast is an intuitive choice to stream *live* 360-degree videos to a large number of mobile users. However, it is a challenging task due to the huge volume of the video data, dynamic nature of the wireless channel conditions, and high user interactivities with the content. To address this challenge, we presented VRCast, a system for live multicast streaming of 360-degree videos to mobile users over cellular networks. VRCast supports user interactivity and viewport switching, optimizes the energy consumption for mobile receivers, accounts for the heterogeneous and dynamic nature of wireless channel conditions, ensures the smoothness of the rendered 360-degree content, maintains fairness among mobile users, and achieves high spectral efficiency of the expensive wireless link. VRCast utilizes tile video coding and DASH streaming, where a 360-degree video is encoded into tiles and each tile has multiple quality representations. VRCast optimally divides mobile users into multiple multicast groups and assigns a quality representation to each tile to maximize the user-perceived video quality. It transmits data in bursts to save energy without compromising the quality.

We evaluated VRCast using trace-based simulations. Our results showed that VRCast significantly outperforms the closest algorithms in the literature. Furthermore, we developed an LTE testbed and evaluated VRCast in it. Our empirical results showed that VRCast achieves smooth quality (no stalls or buffer overflows) and it transmits data in bursts to save energy of mobile devices.

The work in this paper can be extended in multiple directions. For example, we focused on multicasting one 360-degree video to many users. For large-scale concurrent events, network resources are typically pre-allocated separately to multicast sessions. However, further optimizations can be achieved by generalizing our models and solutions to concurrently manage multiple multicasting sessions of 360-degree videos.

Appendix
Illustrative Example

We explain the steps of our solution using the following simple scenario. A budget of 54 resources blocks (RBs) is available in a time-frequency grid of 6 time slots and 9 frequency subcarriers (9 RBs/slot). Nine users are streaming a live 360-degree video. The 360-degree video is divided into 3 tiles and each tile is encoded into 3 quality representations of bitrates 4, 20, and 32 bits respectively. The channel condition (MCS value) and

TABLE II
MCS VALUES AND VIEWPORT TILES

User Index	MCS	Viewport tiles
User 1	1	1, 2
User 2	1	3
User 3	2	2
User 4	2	1, 3
User 5	2	1, 2
User 6	2	2, 3
User 7	4	1, 2
User 8	4	2
User 9	4	2, 3

TABLE III
POSSIBLE GROUPINGS IN THE EXAMPLE

	G1	G2		G3		G4		
MC grps	1,2,4	1,2	4	1	2,4	1	2	4
Users No.	9	6	3	2	7	2	4	3
Min MCS	1	1	4	1	2	1	2	4
RBs No.	54	36	18	12	42	12	24	18
Bits No.	54	36	72	12	84	12	48	72
Bitrate	9	6	12	2	14	2	8	12
Utility	9	8		**11.33**		8		

TABLE IV
RBS REQUIRED TO SEND EACH QUALITY REPRESENTATION

MCS / Bitrate	1 bit/RB	2 bits/RB	4 bits/RB
4 bits (L)	4 RBs	2 RBs	1 RBs
20 bits (M)	20 RBs	10 RBs	5 RBs
32 bits (H)	32 RBs	16 RBs	8 RBs

viewport tiles of each user are shown in Table II. In real LTE networks, there is a direct mapping between the channel conditions of the user to an MCS value. The MCS value and number of assigned RBs define the bit capacity of each RB (bits/RB). However, for simplicity, we define MCS values as number of bits per RB.

Applying the GroupUsers algorithm, we have three different MCS values (1, 2, and 4) with number of users equal (2, 4, and 3) for each MCS, respectively. There are 4 possible groupings as shown in Table III. For each grouping, we have one or more multicast groups (row 1). Each multicast group is served by the minimum MCS of all users in the group (row 3). The number of RBs assigned to each group (row 4) is proportional to the number of users in the group. The number of bits received by each group (row 5) is the multiplication of the number of RBs and the minimum MCS of the group users (bits/RB). Group bitrate (row 6) equals the number of bits divided by the total number of time slots (6 slots). Users average bitrate (row 7), our utility, is calculated as a weighted average of group bitrates according to the number of users per group (row 2). As shown in III, grouping 3 is optimal and maximizes the utility.

Applying the SelectTileQuality algorithm, we have 3 tiles and 3 quality representations for each tile of bitrates: 4 (denoted by L–low quality), 20 (M–medium quality), and 32 (H–high quality) bits. Table IV shows the required RBs to send each quality representation to users of different channel conditions (MCS).

The number of possible quality assignments is 27. For each group, we calculate the utility for each feasible assignment. For Group 1, we have 2 users of minimum MCS 1 and 12 RBs assigned to it and the tile weights are 1, 1, 1. There is only one feasible quality assignment for $MCS = 1$ and $RBs \leqslant 12$ which is to set the low quality representation (4 bits) for each tile (LLL). For Group 2, we have 7 users of minimum MCS 2 and 42 RBs assigned to it and the tile weights are 3, 6, 3. All quality assignments except HHH are feasible for $MCS = 2$ and $RBs \leqslant 42$. As the middle tile has the maximum weight, the optimal solution assigns the highest quality to the middle tile. So, HHM or MHH gives the highest utility. Table V shows the quality assignments with highest utilities for group 2.

TABLE V
FEASIBLE QUALITY ASSIGNMENTS FOR GROUP 2

Quality Assignment	Utility	RBs
MMM	35.95	30
MHM	38.79	36
HHL, LHH	35.35	34
HHM, MHH	**40.18**	42
LHM, MHL	33.94	28

REFERENCES

[1] V. Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update," White Paper, Mar. 2017.

[2] 3GPP, "Enhanced Television Services over 3GPP eMBMS," Oct. 2017. [Online]. Available: http://www.3gpp.org/news-events/3gpp-news/1905-embms_r14

[3] D. Lecompte and F. Gabin, "Evolved multimedia broadcast/multicast service (eMBMS) in LTE-advanced: Overview and rel-11 enhancements," *IEEE Commun. Mag.*, vol. 50, no. 11, pp. 68–74, Nov. 2012.

[4] Verizon, "Verizon Delivers LTE Multicast Over Commercial 4G LTE Network in Indy," May 2014. [Online]. Available: http://www.verizon.com/about/news/vzw/2014/05/verizon-wireless-lte-multicast

[5] Ericsson, "Gigabit Network Launched in China," Aug. 2017. [Online]. Available: https: //www.ericsson.com/en/press-releases/2017/8/gigabit-network-launched-in-china

[6] Google, "eMBMS Support Added in Android 8.1 for Reduced Mobile Network Congestion," Nov. 2017. [Online]. Available: https: //developer.android.com /reference/android/telephony/mbms/package-summary

[7] H. Chen, X. Zhang, Y. Xu, Z. Ma, and W. Zhang, "Efficient mobile video streaming via context-aware raptorq-based unequal error protection," *IEEE Trans. Multimedia*, vol. 22, no. 2, pp. 459–473, Feb. 2020.

[8] S. Almowuena, M. M. Rahman, C. Hsu, A. A. Hassan, and M. Hefeeda, "Energy-aware and bandwidth-efficient hybrid video streaming over mobile networks," *IEEE Trans. Multimedia*, vol. 18, no. 1, pp. 102–115, Jan. 2016.

[9] V. R. Gaddam, M. Riegler, R. Eg, C. Griwodz, and P. Halvorsen, "Tiling in interactive panoramic video: Approaches and evaluation," *IEEE Trans. Multimedia*, vol. 18, no. 9, pp. 1819–1831, Sep. 2016.

[10] B. Tan *et al.*, "Analog coded softcast: A network slice design for multimedia broadcast/multicast," *IEEE Trans. Multimedia*, vol. 19, no. 10, pp. 2293–2306, Oct. 2017.

[11] C. Lentisco, L. Bellido, and E. Pastor, "Reducing latency for multimedia broadcast services over mobile networks," *IEEE Trans. Multimedia*, vol. 19, no. 1, pp. 173–182, Jan. 2017.

[12] J. Chen, R. Mahindra, M. A. Khojastepour, S. Rangarajan, and M. Chiang, "A Scheduling framework for adaptive video delivery over cellular networks," in *Proc. 19th Annu. Int. Conf. Mobile Comput. Netw.*, Miami, FL, USA, Oct. 2013, pp. 389–400.

[13] Facebook, "Enhancing High-Resolution 360 Streaming With View prediction," Apr. 2017. [Online]. Available: https://code.facebook.com/posts/118926451990297/enhancing-high-resolution-360-streaming-with-view-prediction/

[14] Google, "The latest on VR and AR at Google I/O," May 2017. [Online]. Available: https://www.blog.google/products/google-vr/latest-vr-and-ar-google-io/

[15] 3GPP, "TS 36.211, E-UTRA, Physical Channels and Modulation," 2017. [Online]. Available: www.3gpp.org/dynareport/36211.htm

[16] O. A. Niamut *et al.*, "MPEG DASH SRD: Spatial relationship description," in *Proc. ACM ACM Multimedia Syst. Conf.*, Klagenfurt, Austria, May 2016, pp. 1–8.
[17] N. Jiang, V. Swaminathan, and S. Wei, "Power evaluation of 360 VR video streaming on head mounted display devices," in *Proc. 27th Workshop Netw. Oper. Syst. Support Digital Audio Video*, Taipei, Taiwan, Jun. 2017, pp. 55–60.
[18] 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*, Mountain View, CA, USA, Oct. 2017, pp. 315–323.
[19] S. Petrangeli, V. Swaminathan, M. Hosseini, and F. De Turck, "An HTTP/2-based adaptive streaming framework for 360 virtual reality videos," in *Proc. 25th ACM Int. Conf. Multimedia*, Mountain View, CA, USA, Oct. 2017, pp. 306–314.
[20] Y. Bao, H. Wu, T. Zhang, A. A. Ramli, and X. Liu, "Shooting a moving target: Motion-prediction-based transmission for 360-degree videos," in *Proc. IEEE Big Data*, Washington, DC, USA, Dec. 2016, pp. 1161–1170.
[21] F. Qian, B. Han, Q. Xiao, and V. Gopalakrishnan, "Flare: Practical viewport-adaptive 360-degree video streaming for mobile devices," in *Proc. ACM Conf. Mobile Comput. Netw.*, Oct. 2018, pp. 99–114.
[22] J. Chen *et al.*, "Fair and optimal resource allocation for LTE multicast (eMBMS): Group partitioning and dynamics," in *Proc. IEEE INFOCOM*, Hong Kong, China, Apr. 2015, pp. 1266–1274.
[23] F. Kelly, "Charging and rate control for elastic traffic," *Eur. Trans. Telecommun.*, vol. 8, no. 1, pp. 33–37, 1997.
[24] N.-D. Nguyen, R. Knopp, N. Nikaein, and C. Bonnet, "Implementation and validation of multimedia broadcast multicast service for LTE/LTE-advanced in openairinterface platform," in *Proc. IEEE Workshop Perform. Manage. Wireless Mobile Netw.*, Oct. 2013, pp. 70–76.
[25] C. Tofallis, "Add or multiply? a tutorial on ranking and choosing with multiple criteria," *INFORMS Trans. Edu.*, vol. 14, no. 3, pp. 109–119, 2014.
[26] M. Graf, C. Timmerer, and C. Mueller, "Towards bandwidth efficient adaptive streaming of omnidirectional video over HTTP: Design, implementation, and evaluation," in *Proc. ACM Multimedia Syst. Conf.*, Taipei, Taiwan, Jun. 2017, pp. 261–271.
[27] 3GPP, "TS 36.104, E-UTRA, Base Station (BS) Radio Transmission and Reception," 2017. [Online]. Available: http://www.3gpp.org/DynaReport/36104.htm
[28] 3GPP, "TS 36.331, E-UTRA; Radio Resource Control (RRC) Protocol Specification," 2017. [Online]. Available: http://www.3gpp.org/DynaReport/36331.htm
[29] 3GPP, "TS 26.346, Multimedia Broadcast/Multicast Service, Protocols and Codecs," 2017. [Online]. Available: http://www.3gpp.org/DynaReport/26346.htm
[30] T. Stockhammer and M. G. Luby, "Dash in mobile networks and services," in *Proc. Visual Commun. Image Process.*, San Diego, CA, Nov. 2012, pp. 1–6.
[31] T. Paila, R. Walsh, M. Luby, V. Roca, and R. Lehtonen, "FLUTE-file delivery over unidirectional transport," Internet Engineering Task Force (IETF) RFC 6726, Nov. 2012.
[32] H. Ahmadi, O. Eltobgy, and M. Hefeeda, "Adaptive multicast streaming of virtual reality content to mobile users," in *Proc. ACM Multimedia Conf.*, Mountain View, CA, Oct. 2017, pp. 601–605.
[33] X. Corbillon, F. De Simone, and G. Simon, "360-Degree video head movement dataset," in *Proc. ACM ACM Multimedia Syst. Conf.*, Taipei, Taiwan, Jun. 2017, pp. 199–204.
[34] M. Viitanen, A. Koivula, A. Lemmetti, J. Vanne, and T. D. Hämäläinen, "Kvazaar HEVC encoder for efficient intra-coding," in *Proc. IEEE Int. Symp. Circuits Syst.*, Lisbon, Portugal, May 2015, pp. 1662–1665.
[35] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong, "SLAW: A new mobility model for human walks," in *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr. 2009, pp. 855–863.
[36] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn, "BonnMotion: A mobility scenario generation and analysis tool," in *Proc. 3rd Int. ICST Conf. Simul. Tools Techn.*, Malaga, Spain, Mar. 2010, Art. no. 51.
[37] Amarisoft, "Amarisoft full LTE software suite solution," 2018. [Online]. Available: https://www.amarisoft.com/software-enb-epc-ue-simulator/
[38] Expway, "Expway Broadcast Multicast Service Center (Expway BMSC)," Jan. 2017. [Online]. Available: http://www.expway.com/embms/
[39] Y.-J. Yu, P.-C. Hsiu, and A.-C. Pang, "Energy-efficient video multicast in 4G wireless systems," *IEEE Trans. Mobile Comput.*, vol. 11, no. 10, pp. 1508–1522, Oct. 2012.
[40] S. Sharangi, R. Krishnamurti, and M. Hefeeda, "Energy-efficient multicasting of scalable video streams over wimax networks," *IEEE Trans. Multimedia*, vol. 13, no. 1, pp. 102–115, Feb. 2011.
[41] X. Xie, X. Zhang, S. Kumar, and L. E. Li, "piStream: Physical layer informed adaptive video streaming over LTE," in *Proc. 24th Annu. Int. Conf. Mobile Comput. Netw.*, New York, NY, USA, Sep. 2015, pp. 413–425.
[42] X. Xie, X. Zhang, and S. Zhu, "Accelerating mobile web loading using cellular link information," in *Proc. 15th Annu. Int. Conf. Mobile Syst., Appl., Services*, New York, NY, Jun. 2017, pp. 427–439.
[43] C. Zhou, Z. Li, and Y. Liu, "A measurement study of oculus 360 degree video streaming," in *Proc. 8th ACM Multimedia Syst. Conf.*, Taipei, Taiwan, 2017, pp. 27–37.
[44] X. Corbillon, G. Simon, A. Devlic, and J. Chakareski, "Viewport-adaptive navigable 360-degree video delivery," in *Proc. IEEE Int. Conf. Commun.*, May 2017, pp. 1–7.
[45] A. Zare, A. Aminlou, M. M. Hannuksela, and M. Gabbouj, "HEVC-compliant tile-based streaming of panoramic video for virtual reality applications," in *Proc. 24th ACM Int. Conf. Multimedia*, Amsterdam, the Netherlands, Oct. 2016, pp. 170–178.
[46] F. Qian, B. Han, Q. Xiao, and V. Gopalakrishnan, "Flare: Practical viewport-adaptive 360-degree video streaming for mobile devices," in *Proc. 24th Annu. Int. Conf. Mobile Comput. Netw.*, New Delhi, India, Nov. 2018, pp. 99–114.
[47] J. He *et al.*, "Rubiks: Practical 360-degree streaming for smartphones," in *Proc. 16th Annu. Int. Conf. Mobile Syst., Appl., and Services*, New York, NY, USA, Jun. 2018, pp. 482–494.
[48] M. Xiao, C. Zhou, Y. Liu, and S. Chen, "OpTile: Toward optimal tiling in 360-degree video streaming," in *Proc. 25th ACM Int. Conf. Multimedia*, Mountain View, CA, Oct. 2017, pp. 708–716.
[49] A. T. Nasrabadi, A. Mahzari, J. D. Beshay, and R. Prakash, "Adaptive 360-degree video streaming using scalable video coding," in *Proc. 25th ACM Int. Conf. Multimedia*, Mountain View, CA, Oct. 2017, pp. 1689–1697.

Omar Eltobgy received the B.Sc. degree in computer science and engineering from Alexandria University, Alexandria, Egypt, in 2016, and the M.Sc. degree in computer science from Simon Fraser University, Burnaby, BC, Canada, in 2018. He is currently working with Adobe, Ottawa, ON, Canada.

Omar Arafa received the B.Sc. degree in computer science and engineering from Alexandria University, Alexandria, Egypt, in 2016, and the M.Sc. degree in computer science from Simon Fraser University, Burnaby, BC, Canada, in 2019. He is currently working at Fortinet, Burnaby, BC, Canada.

Mohamed Hefeeda (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees from Mansoura University, Mansoura, Egypt, in 1994 and 1997, respectively, and the Ph.D. degree from Purdue University, West Lafayette, IN, USA, in 2004. He is a Professor with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada, where he leads the Network Systems Laboratory. He has co-authored more than 120 papers and multiple granted patents. His research interests include multimedia networking over wired and wireless networks, mobile multimedia, and network protocols. In 2011, he was awarded one of the prestigious NSERC Discovery Accelerator Supplements, which are granted to a select group of distinguished researchers in all Science and Engineering disciplines in Canada.