Adaptive Multicast Streaming of Virtual Reality Content to Mobile Users

Hamed Ahmadi
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada

Omar Eltobgy
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada

Mohamed Hefeeda
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada

ABSTRACT

Streaming virtual reality (VR) content is becoming increasingly popular. Advances in VR technologies now allow providing users with an immersive experience by live streaming popular events, such as the Super Bowl, in the form of 360-degree videos. Such services are highly interactive and impose substantial load on the network, especially cellular networks with inconsistent link capacities. In this paper, we perform rigorous analysis of 1300 VR head traces and propose a multicast DASH-based tiled streaming solution, including a new tile weighting approach and a rate adaptation algorithm, to be utilized in mobile networks that support multicast such as LTE. Our proposed solution weighs video tiles based on user's viewports, divides users into subgroups based on their channel conditions and tile weights, and determines the bitrate for each tile in each subgroup. Tiles in the viewports of users are assigned the highest bitrate, while other tiles are assigned bitrates proportional to the probability of users changing their viewports to include those tiles. We compare the proposed solution against the closest ones in the literature using simulated LTE networks and show that it substantially outperforms them. For example, it assigns up to 46% higher video bitrates to video tiles in the users' viewports than current approaches which substantially improves the video quality experienced by the users, without increasing the total load imposed on the network.

CCS CONCEPTS

•Information systems → Multimedia streaming; •Networks → Mobile networks;

KEYWORDS

Mobile Multimedia, Video Streaming, Adaptive Streaming, Multicast.

1 INTRODUCTION

The virtual reality (VR) market is predicted to flourish sixfold reaching $30 billion by 2020 [3]. Recently, technology giants, such as Facebook and YouTube, have been integrating VR services in their existing platforms. This makes VR content more accessible than ever before. In this paper, VR content refers to immersive 360-degree videos, where users can change their view angles. Moreover, next generation massive MIMO cellular networks will help users stream such content on their mobile devices and enjoy an immersive panoramic viewing experience.

Providing users with an immersive experience requires VR content with 4K+ resolutions and up to 60 fps, which is very bandwidth demanding. A recent measurement study shows that current commercial 360 video delivery platforms are utilizing a delivery scheme similar to traditional Internet videos [19]. In such systems, the entire video is streamed to users. However, employing such approach to stream interactive VR videos wastes significant network bandwidth, because users can only watch a portion of the video at a time depending on the utilized VR headset, while the rest of the transmitted data outside the user viewing range is wasted. For example, for Oculus DK2, the visible portion is 110° horizontally and 90° vertically. It also increases the power consumption of mobile devices. A trivial solution would be to stream only the currently viewed parts of the video. However, it is common in VR applications that users change their viewing directions and if the new viewport is not found in the buffer, users would be shown a blank or freezing picture and lose their immersive experience.

There have been a few attempts to utilize tiling techniques to decrease the required bandwidth for VR content. In such techniques, the video is first divided into smaller tiles. Each tile is then encoded at multiple bitrates. Tiles required to construct the viewport are streamed with high bitrate and the others with low bitrate. The chief challenges of tiling are how to choose tile sizes and weights based on users' viewing behavior, and optimize the streaming tiles bitrates to reduce the total required bandwidth while maintaining the user perceived quality.

Although tiling helps with bandwidth reduction, it is not enough. Streaming a live VR video to millions of users still would not be practical without utilizing multicast schemes. Thanks to the Evolved Multimedia Broadcast Multicast Services (E-MBMS) feature in Long Term Evolution (LTE) networks, base stations can employ point-to-multipoint bearers to serve a huge number of users consuming simultaneously the same content [2] [14]. One of the key challenges of the E-MBMS is the radio resource management. In fact, resource allocation strategies significantly affect the received Signal to Noise Ratio (SNR), the user perceived quality and the network throughput. Recent papers [9], [6] employ a multicast grouping strategy to serve the users with similar channel conditions with the same video quality level, maximizing the spectral efficiency. However, these approaches are not suitable for streaming VR content because of the VR-specific characteristics. For example, users watching a traditional live stream can be served with a single multicast session, while in the case of 360 live streaming, different users may be

ThematicWorkshops'17, October 23–27, 2017, Mountain View, CA, USA

DOI: http://dx.doi.org/10.1145/3126686.3126743

watching different parts of the video, and even the same user can change his/her viewport during the streaming session.

In this work, we propose a hybrid tiling multicast solution for live streaming of VR content. It includes a new tile weighting approach and a rate adaptation algorithm. To do so, we first prepare a fairly big dataset, including over 1300 VR head traces. We perform rigorous analysis to investigate users' viewing behavior and use this information to develop a probabilistic tile weighting technique to better address viewport changes in VR applications. Then, we formulate the problem of multicast adaptive tiled streaming with the goal of maximizing the average data rate of served video streams while keeping the data rates proportional to tile weights. We show that this problem is NP-hard and propose a heuristic algorithm to solve it. Out rate adaptation algorithm can be used with other tile weighting approaches. However, our tile weighting approach better captures users' viewport changes during the rate adaptation cycle, hence allowing longer adaptation cycles and consequently less feedback overhead. Through our extensive simulations, we show that the proposed algorithm can achieve substantial improvements in the overall data rate of video sessions. For example, it provides up to 46.81% more bit rates for the video tiles that are required to construct the viewports with the same amount of radio resources.

The remainder of this paper is organized as follows. Section 2 summarizes the related work in the literature. Section 3 describes the considered system model and operation. Section 4 explains our head traces dataset preparation and analysis, and demonstrates the proposed tile weighting approach. Section 5 formulates the rate adaptation problem and presents our proposed algorithm. Section 6 presents the simulation results and comparisons against other works in the literature. Finally, Section 7 concludes the paper.

2 RELATED WORK

2.1 Virtual Reality Streaming

Streaming virtual reality content had not been feasible until recently. Light headsets with high-quality screens and high-accuracy head tracking enabled this to happen. Multiple works and experiments have been carried out to discover the appropriate settings for streaming such content. In [12], the video is divided into tiles and each tile is encoded at multiple bitrates. The quality of the tiles is determined based on the total available bandwidth and the user's desired view. However, in this work, the viewports are pre-defined. Gaddam et al. [10] leverage user controls such as pan-tilt-zoom and head tracking signals to determine the tiles in the current viewport. They conclude that a pyramidal tile weighting approach is a good trade-off between bandwidth saving and perceived video quality. Qian and Han [19] conduct a measurement study on Facebook and YouTube 360-degree video platforms revealing that both utilize progressive download over HTTP. Then, through trace-driven simulations, they show that tiled streaming could save 40-80% of the bandwidth depending on the video content and number of tiles.

The motion-constrained tile set (MCTS) feature of HEVC [15] has been investigated in [25] [13] [21] to reduce the transport complexity and support devices with a single hardware decoder. Bao et al. [8] provide a dataset to show that head movements can be predicted on a time scale of 100-500ms, leading to 45% bandwidth reduction. They did not consider dynamic adaptive streaming over HTTP (DASH), though.

All of the above works have focused on unicast streaming, while we focus on multicast streaming of popular live events to large-scale mobile users.

2.2 Mobile Multicast

Multiple works have considered incorporating the notion of adaptive streaming into multicast settings. Monserrat et al. [16] utilize a joint unicast-multicast streaming approach, in which mobile terminals are served using unicast or multicast connections. If a multicast session is initiated, the base station will set its transmission power based on the worst channel condition to accommodate terminals at the cell edge. Chen et al. [9] also apply a joint unicast-multicast streaming approach to maximize the average data rate at mobile terminals. Almowuena et al. [6] propose an energy-aware resource allocation algorithm to utilize joint unicast-multicast in both independent cells and single frequency networks (SFNs). Wang et al. [22] propose a mixed resolution tiling scheme for broadcasting zoomable videos on wireless LANs. Their subjective experiments on the impact of having different qualities for tiles in the viewport [23] show that minor differences in resolution are hardly noticed.

Different from prior works, which are designed for traditional 2D videos, we propose a complete multicast VR streaming solution including a new tile weighting approach and a rate adaptation algorithm. Both take 360 video characteristics and user interactivity into consideration.

3 SYSTEM MODEL AND OPERATION

3.1 Mobile Network

We consider a cellular network with multiple mobile terminals as shown in Figure 1. The cellular network supports multicast services such as E-MBMS [14]. Mobile terminals move through the network and may enter or leave MBMS-enabled cells as they move. The link capacity for each mobile terminal fluctuates over time due to physical mobility and time-varying channel impairments such as shadowing, multipath fading, and variations in other traffic served by the same base station. Mobile terminals report the channel state information or channel quality indicator (CQI) to the associated base station via an uplink channel. The granularity of channel state information is at the subcarrier level. Higher modulation and coding scheme (MCS) modes require good channel quality and lead to higher per-resource block capacity. On the other hand, lower MCS modes are more robust and usable for diverse (both good and weak) channel qualities.

The radio resources of a base station are divided along both time, represented by sub-frame, and frequency, represented by subcarrier. A resource block is the smallest unit that can be allocated by the base station.

3.2 Virtual Reality Content

High-quality 360 videos are chiefly recorded by camera array systems. The captured videos are first stitched and warped onto a 3D sphere and then projected to a 2D map using several mapping schemes. The map is encoded and streamed over the network to users. On the client side, a 360 video player calculates and displays

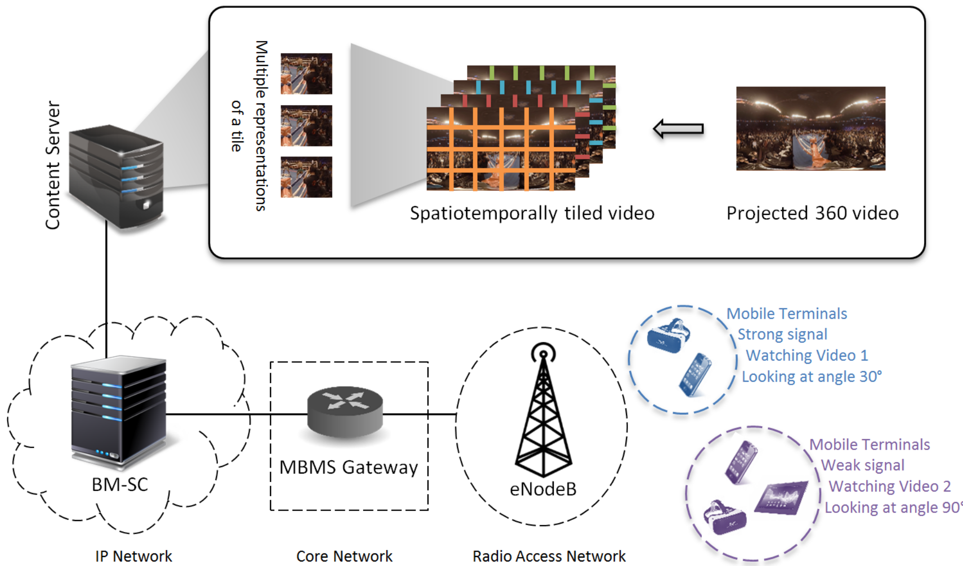


Figure 1: Architecture of a mobile network, in which heterogeneous clients share radio resources and view 360-degree videos.

the viewing area based on the user's viewing direction and field of view (FoV) or viewports.

The projected 360 video, e.g., equirectangular [11] or cubic mappings [17], are first temporally divided into a sequence of small segments. Then, each frame in a segment is spatially divided into a grid of tiles. Tiles are pre-encoded in multiple quality representations, each with a particular video bit rate. These representations are stored on the content server. For each video, there is a Media Presentation Description (MPD) file which informs streaming clients about the bitrate and quality of the available representations, and helps them locate the video tiles. This file utilizes the Spatial Representation Description (SRD) feature of DASH to describe spatial relationships between associated pieces of video content [4] [18].

3.3 Video Streaming Pipeline

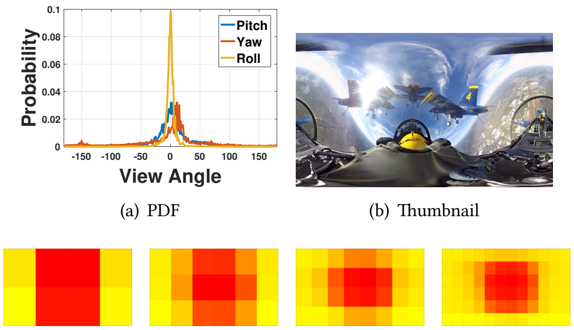
According to the MBMS specifications [2], the content provider supplies the media, as well as the service descriptions and control data, to the Broadcast Multicast Service Center (BM-SC). For DASH streaming, the MBMS download delivery method should be used [2], [20]. In this method, the media content is transferred via HTTP from the content provider to the BM-SC, and via the FLUTE protocol [5] from the BM-SC to the MBMS clients. Associated delivery methods, such as file-repair, are also available and offered on HTTP. Moreover, the delivery method provides functionalities such as security and key distribution, and reliability control by means of forward-error-correction (FEC) techniques.

It is worthy to mention that the specifications allow continuity between DASH over MBMS and DASH over HTTP so that when a mobile terminal exits the MBMS service coverage, it can still receive the service although using the conventional unicast DASH system [2] [14]. It is recommended to have a separate MPD file for each of unicast and multicast schemes.

The content provider creates an MBMS user service on the BM-SC and updates the MPD file periodically via Update Session procedure. This MPD file contains all representations available on the content provider server. The BM-SC modifies the MPD file, determining the suitable representation for each client as explained later, and then sends it to clients. An MBMS-compatible DASH client discovers the MBMS user services via User Service Discovery/Announcement procedure and then chooses the desired video stream. Then, it registers to that service and activates the MBMS service bearer, specified in the service description. At this time, the MBMS delivery function in BM-SC is triggered, the MBMS service bearer is activated, and the content is transferred to all listening clients.

During the session, the MBMS client informs the BM-SC about which video tiles are required for rendering the viewport for that client. This is carried out periodically through the Consumption Reporting procedure, which has been provisioned in the MBMS specification to best utilize the network resources in supporting unicast and multicast services. The reporting interval, that shall be set according to the duration of the scheduling window, is included in the service description by the MBMS service provider. Consumption reports are based on XML and therefore can be easily extended to include extra information. They allow MBMS clients to inform the BM-SC about the importance (weight) of each video tile according to the user's current viewport and his/her predicted viewport in the short time ahead. Using such information and running our rate adaptation algorithm, the BM-SC determines the appropriate representation for each video tile that should be streamed to clients. The details of the algorithm are described in Section 5.

According to the output of the algorithm, the BM-SC updates the MPD file. Specifically, from all available representations of a video tile, only the ones that the rate adaptation algorithm has determined will be published as available in the updated MPD file.



(a) PDF (b) Thumbnail

(c) 2×4 (d) 3×6 (e) 4×8 (f) 6×12

Figure 2: Probability distribution functions (PDFs) of the Yaw, Pitch, and Roll angles and average percentage that each tile is required to construct the viewport for a sample 4K video.

The algorithm might consider more than one multicast group for a video tile. Therefore, the BM-SC specifies the corresponding CQI range for each representation. Upon the receipt of the updated MPD file, the client knows, according to its CQI, what representation should be streamed for each video tile and starts downloading them. When the MBMS service is over or the user just wants to leave, the client deregisters from the service and deactivates the MBMS service bearer.

4 TILING ANALYSIS

As part of our proposed solution, we first prepare a head traces dataset and analyze it to find suitable tiling configurations based on bitrate overhead, viewport change distribution, and the ROI coverage. It is worth mentioning that the method we utilize to determine tiling configurations can be applied to any head traces datasets.

4.1 Dataset Preparation

We combine two head traces datasets to cover a variety of VR video content and viewing behaviors. They include a total of 1335 sessions. The first dataset [8] contains sixteen 360-degree video clips, mostly at 4K resolution, downloaded from YouTube and spanning sports, landscape, and entertainment categories. We refer to them as V1-V16. They were watched by 153 volunteers resulting in a total of 985 recording sessions. Each session includes the view angles of the subject for every 0.1 seconds, in the form of Euler angles.

The second dataset [1] contains seven 360-degree videos from YouTube: Elephants, Rhinos, Diving, Rollercoaster, Timelapse, Venice, and Paris. We refer to them as V17-23. A total of 350 sessions from 59 volunteers were collected. The view angles are stored in the form of Hamilton quaternions representation. Quaternions are preferred over Euler angles in computer graphics because they are more computationally efficient and overcome the Gimbal lock issue. However, Euler angles are used for visualization purposes.

The sampling frequency is not constant, therefore we use linear interpolation, as utilized in [8], to have uniform samples every 0.1 seconds.

4.2 Tile Sizes

The size of video tiles is important in several aspects. First, dividing the video into tiles negatively affects the efficiency of the video encoder, since the motion vector search in the encoding process becomes restricted around the tile borders. Second, smaller tile sizes, i.e., more number of tiles, will result in more multicast sessions and consequently requires more resource blocks. On the other hand, big tile sizes do not capture the benefits of the region of interest (ROI). More specifically, when big tile sizes are used, a large subset of tiles are always required to construct the viewport, even though only a small portion of the video is watched by the user at a time. Hence, the maximum gain of rate adaptation decreases.

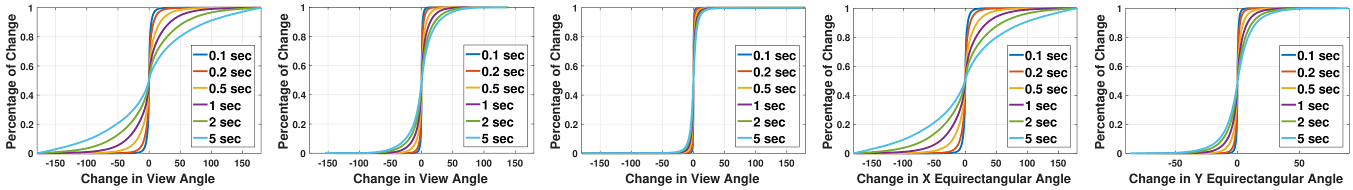
Figure 2 shows the average percentage of required tiles to construct the userfis viewport for a sample video. It also shows the probability distribution function (PDF) of Euler angles for the video for all the users who watched it. This video is 30-second long, and watched by about 50 subjects. The color spectrum from red to yellow represents the range from 100% to 0%. As can be seen, as the tile size decreases, the region of interest is discerned more precisely, resulting in more efficient bandwidth saving. Note that the size of the ROI is constant; however, the number of tiles in the ROI changes as the tile size changes. The average percentage of tiles in non-ROI regions across the videos in our dataset is 44.74%, 56.95%, 67.54%, and 71.87% for 2×4, 3×6, 4×8, and 6×12, respectively. The value for 6×12 is 4.33% more than that for 4×8, however choosing it would be at the cost of having 40 more files per segment on the server. Moreover, the average bitrate overhead over all the videos in the dataset, resulting from tiling, is about 6% for 6×12, while that is 3.78% for 4×8. Therefore, 4×8 tiling is preferred here.

4.3 Tile Weights

Interactivity plays a key role in VR applications. When the user changes the viewport, if a video tile required to construct the new viewport does not exist at the client, a blank or static image has to be temporarily shown, which causes dramatic video quality degradation. Therefore, it is recommended to stream all tiles in such highly interactive applications, albeit with different bit rates for each tile. However, determining the bitrates of the tiles is a challenging task, because it should consider current viewport and its change in near future. To address this issue, tiles are weighed and their bitrates are assigned proportional to their weights. Here, we first review binary and pyramidal weightings [19]. Then, we analyze the viewport changes in terms of view angles and tiles, and propose a new data driven probabilistic tile weighting approach.

Binary. This is a straightforward approach in which the tiles in the viewport have the maximum weight and other tiles have the minimum weight. The drawback of this approach manifests itself when a user changes the viewport and a neighboring tile is required to construct the new viewport. Since that tile's weight is set to the minimum, the video quality degradation would be significant.

Pyramidal. The weight of a tile gradually decreases according to its distance to the viewport. All neighboring tiles of the viewport tiles are at distance one. Neighbors of neighbors are at distance two, and so on. The challenge of this approach is how to determine tile weights at a specific distance. A simple idea is to normalize the weights based on the distance range. For panoramic videos, the



(a) Yaw (b) Pitch (c) Roll (d) Longitude (e) Latitude

Figure 3: Cumulative distribution functions (CDFs) of view angle change

authors of [19] suggest to set the weights according to the zoom level and tilt and pan tendency.

Probabilistic. We propose to weigh the tiles based on the probability distributions of the viewport changes, which we infer by analyzing the head traces in our dataset. Figure 3 illustrates the cumulative distribution function of view angle change in three directions on different time scales in our dataset. As can be seen, small angle changes are far more likely than big angle changes, especially when the time scale is small. Also, the amount of change in Yaw is bigger than that of Pitch and Roll. For one-second time scale, almost 80% of the viewport changes are in the ranges [-38.1, 36.2], [-14.5, 15.2], and [-5.3, 5.0] for Yaw, Pitch, and Roll angles, respectively.

We can perform the same calculation in the 2D projected domain. For example, Figure 3 also represents the view changes along the equirectangular axes. Therefore, to assign the tile weights, we calculate the probability of tile changes rather than view angles. We want to know how far the viewport tiles at the beginning of the time window will probabilistically move during the time window. Let W_{N_p} be the weight of neighborhood N_p whose distance from the current viewport is p. It can be calculated as follows.

$$W_{N_p} = \frac{\sum_i \sum_s \sum_q \sum_{t \in N_{0,i}^s} h(t \in N_{p,i}^{s+q\delta})}{\sum_i \sum_s Q|N_{0,i}^s|}, \tag{1}$$

where s is the start time of each time window, Q is the number of interpolated viewport samples in each time window, δ is the interpolation interval, $N_{p,i}^s$ is the set of tiles in the neighborhood with distance p from the user i's viewport at time instance s. Note that $N_{0,i}^s$ is user i's viewport at time instance s. Function $h(\cdot)$ returns one if its argument is satisfied, otherwise returns zero. Eq. (1) calculates how probable a tile at distance p from the viewport at the beginning of the time window turns out to be a viewport tile during the time window.

5 RATE ADAPTATION

5.1 Problem Statement

Our rate adaptation problem for streaming virtual reality content over cellular networks can be stated as follows.

Given the video requests, viewports, and channel conditions of multiple mobile terminals streaming tiled 360 videos, determine the optimal transmission scheduling for each base station that assigns the available resource blocks to multicast sessions, decides the number of multicast groups, and determines the data rate of each tile stream for each mobile terminal such that the average video quality for all mobile terminals is maximized.

$$\max_x \sum_i^{|L|} \sum_j^{|T|} \sum_k^{|M|} x_{i,j,k} U(\xi_i, R(t_j, m_k)) \tag{2a}$$

$$\text{s. t. } \forall i \in L, \sum_j^{|T|} \sum_k^{|M|} x_{i,j,k} = 1 \tag{2b}$$

$$\sum_j^{|T|} \sum_k^{|M|} y_{j,k} \frac{R(t_j, m_k)}{\alpha_{j,k}} \leqslant \Pi \tag{2c}$$

$$y_{j,k} = \begin{cases} 1 & \exists i \in L, x_{i,j,k} = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2d}$$

The mathematical formulation of the problem is given in Eq. (1). Symbols used in the paper are listed in Table 1. Let L be the set of mobile terminals, and T the set of all video tiles available. Since we are considering a live streaming scenario, we assume that all users who are streaming the same video are requesting the same segment index. Moreover, VR applications are interactive and users can freely change their viewports. If a video tile is not sent and a user happens to change his/her viewport to that video tile, the client video player would be unable to construct that part of the scene, resulting in huge video quality degradation. Therefore, all video tiles are streamed at all time, although with minimum qualities. Also, note that the formulation in Eq. (1) allows different tile sizes and weights.

At each adaptation cycle (equals to segment size), we need to form multicast groups and set their data rates such that the average utility for all mobile terminals is maximized, Eq. (1a). Let M be the set of MCS modes. The total number of possible sessions is the number of video tiles multiplied by the number of MCS modes. However, based on users' channel conditions and viewports only a set of these sessions will be used. The decision variable $x_{i,j,k}$ determines which user should be in what multicast session. The utility of each decision is measured by function U which depends on the user's viewport and the tile's data rate, which is a function of the relative weight of the tile to the user's viewport and the corresponding MCS for the session. The constraint in Eq. (1b) ensures that users receive only one representation for each video tile. The constraint in Eq. (1c) implements the restriction on the available radio blocks for this service, denoted by Π. It uses an auxiliary variable $y_{j,k}$, defined in Eq. (1d), to inspect whether any user is using the multicast session (j, k), i.e., the session in which the video tile t_j is streamed with the MCS m_k. The variable $\alpha_{j,k}$ is

Algorithm 1: MVR (Multicast Virtual Reality)

Input :
$T \leftarrow$ Set of requested video tiles
$R \leftarrow$ Set of available data rates for video tiles
$V \leftarrow$ Set of users' viewports
$M \leftarrow$ Set of available MCSs
$B \leftarrow$ Bandwidth available for video services
Output:
$X \leftarrow$ Multicast subgroups and their bitrates

1 $X = \phi$
2 // Perform the minimum grouping stage to maximize the number of served terminals
3 **for** *each video tile t* **do**
4   m_t = minimum MCS suited for all users of this tile
5   r_t = minimum bit rate for this tile
6     $X_{i,t,m_t} = 1, \forall i \in L$, s.t i watches tile t
7 **end**
8 // Perform rate adaptation to increase viewport rate and utility
9 **while** *Bandwidth(X) <B* **do**
10     $U = \phi$
11   **for** *each video tile t* **do**
12   **for** *each MCS group m* **do**
13   // calculate the sum of utility over **all** users, if the bitrate were increased for each **MCS group** m_k (not multicast subgroup)
14             $X_{t,m}$ = GetNext(r_t, m)
15             $u_{t,m} = \sum_i$ UserUtility(t, r_t, m, ξ_i, R)/NetworkUtility($X_{t,m}$)
16             $U = U \cup u_{t,m}$
17   **end**
18   **end**
19   // Select the step which results in the maximum utility
20     $[\tilde{t}, \tilde{m}, \tilde{r_t}]$ = FindBestStep(U)
21   // Users with higher MCS can join this session if their bit rates are less than or equal to $\tilde{r_t}$
22   Update(X, $\tilde{t}$, $\tilde{m}$)
23   **if** *Bandwidth(X) >B* **then**
24   RollBack(X)
25   break
26   **end**
27 **end**

the radio resource capacity of the session (j, k) which is basically the number of bits that can be transmitted per each radio resource using the MCS mode m_k.

5.2 Proposed Algorithm

The formulation in Eq. (1) is a binary integer non-linear programming problem which is NP-hard. It can be solved by dynamic programming approaches. However, the running time of its worst cases is exponential. Therefore, we develop a heuristic algorithm, referred to as MVR (short for Multicast Virtual Reality) streaming algorithm.

A simple rate adaptation mechanism is to serve all members in a multicast group with the same data rate. In this case, to make sure that all members are able to receive the video, the data rate is determined based on the member with the worst channel condition. This approach results in a poor spectral efficiency [7].

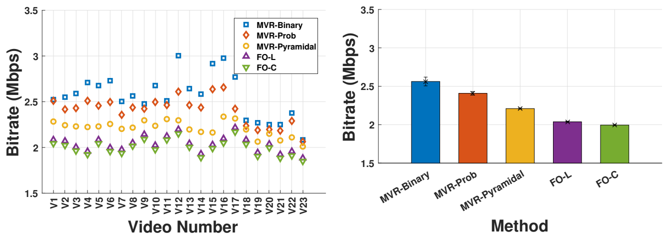
Table 1: Symbols used in this paper.

Symbol	Description
L	Set of mobile terminals.
V	Set of 360-degree videos available at the content server
T	Set of all the video tiles available at the content server. It allows different number of segments and tiles for each video.
M	Set of modulation and coding scheme (MCS) modes.
ξ_i	Viewport of user i at the beginning of the adaptation cycle
Π	Total available radio resources.
$\alpha_{j,k}$	Radio resource capacity for multicast session (j, k).
$x_{i,j,k}$	The decision variable that determines whether user i receives video tile t_j streamed with modulation and coding scheme m_k.
$y_{j,k}$	Auxiliary variable to examine whether any users has been assigned to multicast session (j, K).

To address this problem, a multicast group is divided into multiple subgroups and each subgroup is served with a transmission power and an MCS mode, with which its worst mobile terminal can decode. The subgrouping, however, is a challenging task. First, the channel conditions are not constant. Second, users' viewports are dynamically changing over the course of time, even during an adaptation cycle. Therefore, the quality of the video tiles should be set accordingly. Finally, adding more multicast sessions exhaust more resource blocks while the number of resource blocks is limited.

To tackle the aforementioned issues, we propose the MVR algorithm that divides the incoming video requests into subgroups and determines the data rate for each one. Algorithm 1 shows the pseudo code of our algorithm. Once the variables have been initialized, the algorithm starts with a minimum grouping stage, which utilizes a minimum number of multicast groups equal to the number of requested video tiles, each group is served with the minimum available bitrate, in order to maximize the number of served terminals in the system (lines 3-7). The algorithm then incrementally increases the data rate for one of the video tiles, which results in maximum total utility. The utility of each step is the sum of all user's utilities over the network utility (lines 13-14). The network utility is the average ratio of the total bitrate to the number of assigned resource blocks. This increase is only one-step, meaning that it chooses the immediate next representation. Also, note that the increase is not for all users who need that video tile. In fact, it is done only for one of the MCS groups. However, users who need that video tile and have an equal or greater MCS would also be able to receive that video tile. Therefore, once the increase with maximum utility is found, the algorithm updates the status of the subgroups through the decision variable and examines whether there are enough resources for the new subgrouping or not (lines 20-24). This procedure is repeated until the resource block limit is met.

In the ideal solution for a user, the quality of each video tile should be the maximum possible quality among the available representations multiplied by the ratio of that tile's weight to the maximum tile weight. This way, video tile(s) with maximum weight would be streamed with the maximum quality representation, while the quality of the other video tiles would be proportional to their weights. In practice, when there is not enough resources, we deviate from this ideal solution. In order to measure how much close



(a) For each video (b) Average over all videos

Figure 4: Average bitrate of viewport tiles.

Algorithm 2: User's utility calculation function

Input :
$t \leftarrow$ Current tile
$r \leftarrow$ Current tile's bit rate
$m \leftarrow$ Current MCS group
$\xi \leftarrow$ Current user's viewport
$R \leftarrow$ Set of available data rates for video tiles
Output:
$u \leftarrow$ Current user's utility

// Set next qualities to current qualities
$Q_{\text{next}} = Q_{\text{current}}$
// Check if the bit rate can be increased for current video tile t
$r_{\text{next}} = \text{nextBitRate}(r)$
if r_{next} **then**
  // If user's MCS is greater than or equal m, update tile's quality
  if $m_u \geqslant m$ **then**
    $Q_{\text{next}} = \text{getQuality}(Q_{\text{current}}, t, r_{next})$
  end
end
// User's utility increases as each tile quality increase approaching $Q_{\max}$
$S_{\text{fulfillment}} = \text{mean}(Q_{\text{next}}/Q_{\max})$
// Tile qualities should be proportionate to tile weights
$F = Q_{\text{next}}./\text{GetWeights}(\xi)$
$S_{\text{balance}} = 1 - MAE_{i<j}(F_i, F_j)$
$u = \text{mean}(S_{\text{fulfillment}}, S_{\text{balance}})$
return u

an alternative solution is to the ideal solution, we define a utility function.

Algorithm 2 shows the pseudo code of calculating this utility for a single user. This function considers two key aspects. First, the quality of video tiles should be close to the quality of video tiles in the ideal solution. Since the number of representations is finite, these values can be normalized and averaged. In the utility function, this averaged value is referred to as the fulfillment score. Second, the fulfillment score alone cannot capture the total merit of the solution, because it is unable to measure the balance of the quality distribution among tiles. In the ideal solution, the ratios of video tile qualities to their weights are virtually the same for all the tiles, meaning that their mean absolute error (MAE) is zero. Thus, the utility function normalizes these ratios and calculates their MAE. This value subtracted from one represents the balance among the video tile qualities and is referred to as the balance score in the utility function. The final utility is the average value of these two scores. The utility of each possible decision made in the main algorithm's iterations is the utility summation of all users.

6 EVALUATION

We implemented the proposed MVR algorithm and compared it against the closest algorithm in the literature [9], referred to as FO, which exploits the concept of multicast subgrouping to provide fair and optimal transmission scheduling decisions. The weighting function in the FO algorithm can be set to constant (FO-C) or linear (FO-L). We conduct trace-driven simulations. The video requests and head traces are obtained from the dataset. The channel conditions are generated as explained in Subsection 6.1. The tile sizes and weights are determined as explained in Subsections 4.2 and 4.3, respectively.

6.1 Setup

We simulate an LTE cellular network, with 20 MHz FDD physical profile, using OPNET Modeler and its LTE module [24]. We use the channel conditions obtained from OPNET as input to the algorithm. In practical scenarios, mobile operators usually install base stations in crowded areas to serve most users with strong signals. Accordingly, in our simulations, mobile users are randomly distributed within each cell such that the majority of users, about 90% of them, are densely populated within 1/3 of the cell radius and the rest are sparsely scattered around the rest of the cell area. Mobile terminals move following the random waypoint model in which mobility speed is randomly chosen between 0 and 18 km/hr. They send a channel quality indicator (CQI) report to the associated base station every 100 ms. According to the specifications, up to 60% of the radio resources can be dedicated to MBMS transitions [14]. In our simulations, we use 50% of the radio resources for this purpose.

There is a total of 23 videos in our dataset as explained in Subsection 4.1. The videos have different bitrates. We consider 12 different representations from 1 to 10 Mbps, upscaling or downscaling the videos if necessary. We generate the DASH video tiles for each video. As discussed in Section 4.2, we choose 4×8 tiling. The segments, hence tiles, are one second long. In each scenario, one of the videos is streamed by an average of 200 users. Each user randomly chooses a head trace from the existing head traces in the dataset for the video that is being streamed in the scenario, and switches his/her viewport based on that. The length of the simulations depends on the length of the videos, 30 seconds for V1-16, and about 60 seconds for V17-23. We run each scenario five times and report the average of the results.

6.2 Results

We show the performance gain achieved by our proposed algorithm (MVR) and tile weighting approach (Probabilistic), using three performance metrics: 1) average viewport bitrate, 2) the impact of viewport change during the scheduling window, and 3) the spectral efficiency.

6.2.1 Average Viewport Bitrate. 360-degree videos cover all the surroundings around the center point of the camera. However, a user only watches a portion of the entire video at a time. Therefore, the bit rate of the tiles used in the construction of the user's

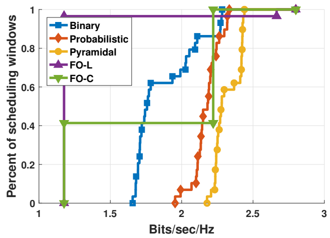
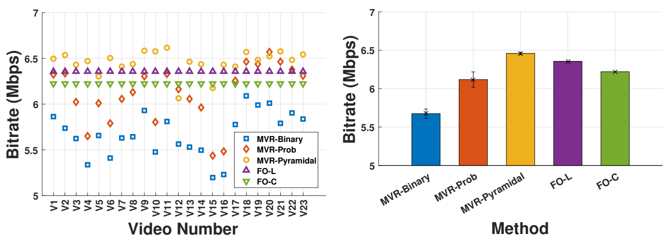
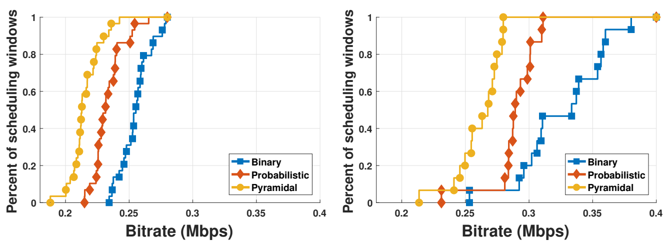


Figure 7: Cumulative distribution function of the average spectral efficiency.



(a) For each video (b) Average over all videos

Figure 5: Average bitrate of the whole video.



(a) 1-second (b) 2-second

Figure 6: Cumulative distribution function of the average bitrate change in viewport tiles during a scheduling window.

viewport has a key role on the user's perceived quality. Figure 4 shows the average viewport bitrate for each video and across all the videos. Our algorithm assigns up to 46.81% more bitrate for the tiles required to construct the viewport (Figure 4(a)). FO-C and FO-L distribute the bitrate uniformly over the video and therefore provide the least share of bitrate for the user's viewport. On average, MVR-Probabilistic provides 18.19% and 20.68% more bitrates for the tiles in viewport, compared to FO-L and FO-C, respectively. The error bars in Figure 4(b) represent the viewport bitrate variation over the time which is only 0.007 Mbps for the proposed probabilistic weighting, compared to 0.057 Mbps for the binary approach.

Figure 5 shows the average bit rate of each streamed video achieved by different algorithms for three tile weightings. It also shows the average video bitrate across all the videos. As can be seen, our algorithm when used with binary and probabilistic weightings, requires less bandwidth compared to FO-C and FO-L. For example, MVR-Probabilistic requires 3.7% less bandwdith than FO-L.

6.2.2 Interactivity Impact. In order to adaptively distribute the available bitrate among the video tiles, changes in users' viewports should be reported back to the adaptation algorithm. However, it cannot happen too frequently, otherwise its overhead cancels out the gain of the bitrate adaptation. The adaptation is carried out based on the reported viewports at the beginning of each scheduling window. However, the user's viewport changes during the scheduling window. We measure the change in the bitrates of the viewport tiles during the scheduling window, to evaluate the success of different weighting approaches in capturing the aforementioned issue. To do so, we used the interpolated view angle values in each scheduling window that have been provided in the dataset. Figure 6 shows the cumulative distribution function of the average bitrate change in viewport tiles during a scheduling window for 1-second and 2-second scheduling windows. For all the weighting approaches, the amount of change increases as the length of scheduling window increases. The performance of the probabilistic weighting stands in between the other two approaches. In Figure 6(a), in 80% of scheduling windows, the viewport bitrate changes are less than 0.22, 0.24, and 0.27 Mpbs, for the pyramidal, probabilistic, and binary approaches, respectively. Comparing Figure 4 and 6 indicates that the proposed probabilistic weighting approach is a compromise between achieving high viewport bitrate and less viewport bitrate variations.

6.2.3 Spectral Efficiency. The spectral efficiency is defined as the transmitted data rate (in bits per second) divided by the allocated bandwidth (in Hertz). According to Figure 7, the proposed algorithm outperforms FO-C and FO-L in terms of spectral efficiency for all the weighting approaches. FO-L has the worst performance. The spectral efficiency of MVR-Probabilistic is greater than FO-C for 66% of the time. For the other 34%, the difference is still very small.

7 CONCLUSIONS

Multicast is a natural choice to live stream popular events in the form of 360 videos to large-scale user communities. It is, however, a complex task due to the inherent high interactivity of VR applications and their high bandwidth requirements. To address these issues, we proposed a new data-driven probabilistic tile weighting approach and a new rate adaptation algorithm for mobile multicast environments. We prepared a comprehensive dataset of VR head traces and performed detailed analysis to investigate the viewing behavior of users and develop a new tile weighting. We evaluated the proposed solution and compared it against others in the literature using trace-driven simulations. Our results show that it assigns significantly higher video bitrates (up to 46%) for the video tiles in users' viewports, allowing them to freely change their view directions while observing much less video quality degradation. Moreover, our solution is compatible with DASH and can be employed in multicast supported mobile networks, such as LTE.

ACKNOWLEDGMENTS

This work is supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada, and by the Qatar National Research Fund (grant # [NPRP8-519-1-108]).

REFERENCES

- [1] 2016. 360-Videos Head Movements Dataset. (2016). http://dash.ipv6.enstb.fr/headMovements/.
- [2] 2016. 3GPP TS 26.346 V14.2.0; Multimedia Broadcast/Multicast Service (MBMS); Protocols and codecs. (December 2016).
- [3] 2016. Augmented/Virtual Reality revenue forecast revised to hit $120 billion by 2020. (2016). http://goo.gl/Lxf4Sy.
- [4] ISO/IEC 23009-1:2014/Amd 2:2015. 2015. Spatial relationship description, generalized URL parameters and other extensions. (July 2015).
- [5] IETF RFC 3926. 2004. FLUTE - File Delivery over Unidirectional Transport. (October 2004).
- [6] Saleh Almowuena, Md. Mahfuzur Rahman, Cheng-Hsin Hsu, Ahmad AbdAllah Hassan, and Mohamed Hefeeda. 2016. Energy-Aware and Bandwidth-Efficient Hybrid Video Streaming Over Mobile Networks. 18, 1 (January 2016), 102–115.
- [7] G. Araniti, M. Condoluci, A. Iera, A. Molinaro, J. Cosmas, and M. Behjati. 2014. A Low-Complexity Resource Allocation Algorithm for Multicast Service Delivery in OFDMA Networks. *IEEE Transactions on Broadcasting* 60, 2 (June 2014), 358–369.
- [8] Yanan Bao, Huasen Wu, Tianxiao Zhang, Albara Ah Ramli, and Xin Liu. 2016. Shooting a Moving Target: Motion-Prediction-Based Transmission for 360-Degree Videos. In *2016 IEEE International Conference on Big Data (Big Data)*. 1161–1170.
- [9] Jiasi Chen, Mung Chiang, Jeffrey Erman, Guangzhi Li, K.K. Ramakrishnan, and Rakesh K. Sinha. 2015. Fair and optimal resource allocation for LTE multicast (eMBMS): Group partitioning and dynamics. In *IEEE Conference on Computer Communications (INFOCOM)*. 1266–1274.
- [10] Vamsidhar Reddy Gaddam, Michael Riegler, Ragnhild Eg Carsten Griwodz, and Pl Halvorsen. 2016. Tiling in Interactive Panoramic Video: Approaches and Evaluation. 18, 9 (September 2016), 1819–1831.
- [11] Carsten Grijnheii, Aljoscha Smolic, and Thomas Wiegand. 2002. Efficient representation and interactive streaming of high-resolution panoramic views. In *Proceedings of International Conference on Image Processing*, Vol. 3. 209–212.
- [12] Masayuki Inoue, Hideaki Kimata, Katsuhiko Fukazawa, and Norihiko Matsuura. 2010. Interactive Panoramic Video Streaming System over Restricted Bandwidth Network. In *Proceedings of the 18th ACM International Conference on Multimedia*. NewYork, USA, 1191–1194.
- [13] Jean Le Feuvre and Cyril Concolato. 2016. Tiled-based Adaptive Streaming Using MPEG-DASH. In *Proceedings of the 7th International Conference on Multimedia Systems (MMSys '16)*. ACM, New York, NY, USA, Article 41, 3 pages.
- [14] D. Lecompte and F. Gabin. 2012. Evolved multimedia broadcast/multicast service (eMBMS) in LTE-advanced: overview and Rel-11 enhancements. *IEEE Communications Magazine* 50, 11 (November 2012), 68–74.
- [15] K. Misra, A. Segall, M. Horowitz, S. Xu, A. Fuldseth, and M. Zhou. 2013. An Overview of Tiles in HEVC. *IEEE Journal of Selected Topics in Signal Processing* 7, 6 (Dec 2013), 969–977.
- [16] Jose F. Monserrat, Jorge Calabuig, Ana Fernandez-Aguilella, and David Gomez-Barquero. 2012. Joint Delivery of Unicast and E-MBMS Services in LTE Networks. 58, 2 (June 2012), 157–167.
- [17] King-To Ng, Shing-Chow Chan, and Heung-Yeung Shum. 2005. Data compression and transmission aspects of panoramic videos. 15, 3 (January 2005), 82–95.
- [18] Omar A. Niamut, Emmanuel Thomas, Lucia D'Acunto, Cyril Concolato, Franck Denoual, and Seong Yong Lim. 2016. MPEG DASH SRD: Spatial Relationship Description. In *Proceedings of the 7th International Conference on Multimedia Systems (MMSys '16)*. ACM, New York, NY, USA, Article 5, 8 pages.
- [19] Feng Qian, Bo Han, and Vijay Gopalakrishnan. 2016. Optimizing 360 Video Delivery over Cellular Networks. In *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*. NewYork, USA, 1–6.
- [20] Thomas Stockhammer and Michael Luby. 2012. Dash in mobile networks and services. In *Visual Communications and Image Processing*. 1–6.
- [21] Y. Snchez, R. Skupin, and T. Schierl. 2015. Compressed domain video processing for tile based panoramic streaming using HEVC. In *2015 IEEE International Conference on Image Processing (ICIP)*. 2244–2248.
- [22] Hui Wang, Mun Choon Chan, and Wei Tsang Ooi. 2015. Wireless Multicast for Zoomable Video Streaming. 12, 1 (August 2015), 5:1–5:23.
- [23] Hui Wang, Vu-Thanh Nguyen, Wei Tsang Ooi, and Mun Choon Chan. 2014. Mixing Tile Resolutions in Tiled Video: A Perceptual Quality Assessment. In *Proceedings of Network and Operating System Support on Digital Audio and Video Workshop (NOSSDAV '14)*. ACM, New York, NY, USA, Article 25, 6 pages.
- [24] Yasir Zaki, Thushara Weerawardane, Carmelita Grg, and Andreas Timm-Giel. 2011. Long Term Evolution (LTE) Model Development Within OPNET Simulation Environment. In *OPNETWORK*. Bethesda, MD, USA.
- [25] Alireza Zare, Alireza Aminlou, Miska M. Hannuksela, and Moncef Gabbouj. 2016. HEVC-compliant Tile-based Streaming of Panoramic Video for Virtual Reality Applications. In *Proceedings of the 2016 ACM on Multimedia Conference (MM '16)*. ACM, New York, NY, USA, 601–605.