VRCast: Mobile Streaming of Live 360-Degree Videos

by

Omar Eltobgy

B.Sc. Computer Engineering, Alexandria University, Egypt, 2016

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
School of Computing Science
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY
Fall 2018

Approval

Name:	**Omar Eltobgy**
Degree:	**Master of Science (Computing Science)**
Title:	**VRCast: Mobile Streaming of Live 360-Degree Videos**
Examining Committee:	**Chair:** Steven Bergner University Research Associate

Mohamed Hefeeda
Senior Supervisor
Professor

Jiangchuan Liu
Supervisor
Professor

Ryan Shea
Internal Examiner
Assistant Professor

Date Defended: **December 10, 2018**

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 smart phones, tablets, and head-mounted displays. Unicast streaming of immersive content on cellular networks 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, which include handling user interactivity, ensuring smooth quality, supporting user mobility, conserving the energy of mobile receivers, and ensuring fairness among users. We propose a comprehensive solution for the problem of live streaming of 360-degree videos to mobile users, which we refer to as VRCast. VRCast is designed for cellular networks that support multicast, such as LTE. It divides the 360-degree video into tiles and then solves the complex live streaming problem in two steps to maximize the viewport quality of users and ensure a smooth quality within the same viewport while saving the energy of mobile devices and achieving fairness across users. Extensive trace driven simulation and real LTE testbed results show 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 enhances the median frame quality by up to 22% and reduces the variation in the spatial quality by up to 53% and improves the energy saving for mobile devices by up to 250%.

Keywords: Mobile Multimedia; Video Streaming; 360-Degree Videos; Multicast

Dedication

To my beloved parents and fiancè.

Acknowledgements

First of all, I would like to express my special thanks and sincere gratitude to my senior supervisor, Professor Mohamed Hefeeda, for guiding me in the last two years with a lot of patience, care, and support. I would also like to extend my thanks to my supervisor, Professor Jiangchuan Liu, for his time and support. In addition, I want to express my sincere thanks to Dr. Ryan Shea, my thesis examiner, for his efforts to review this thesis carefully and improve it with their thoughtful and constructive comments. Furthermore, many thanks to Dr. Steven Bergner for taking the time to chair my thesis defense.

Also, I would like to thank all my colleagues at the Network Systems Lab for their support and help, especially Omar Arafa who was my collaborator in this research. I would also like to thank my friends for their encouragement and constant support. I am indebted to my family for their love, encouragement, and endless support. I owe my deepest gratitude to my parents. I owe my loving thanks to my fiance, Bassant. Without their encouragement and understanding, it would be impossible for me to finish this thesis.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 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 [16]. 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 *live* video sessions. 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. Multicast offers an efficient and scalable approach to stream live videos to many users. The current generation (4G) of cellular networks already have support for multicast services. For example, the evolved Multimedia Broadcast Multicast Service (eMBMS) is part of the LTE standard [38]. And because of recent enhancements to eMBMS [1] that addressed limitations observed in early deployments, many cellular operators have started deploying or experimenting with multicast services. For instance, Verizon, Korea Telecom, and China Unicom have launched multicast services using eMBMS [52, 20]. Telstra, AT&T, and Globe have committed to launching eMBMS, and 31 other operators have been testing the technology [29]. Furthermore, Google has introduced eMBMS support in the developer preview of Android 8.1 for its Nexus and Pixel phones [26].

Prior works have proposed various optimizations for mobile multicast services in terms of bandwidth, video quality, and mobile energy consumption [8]. Most of these works, however, are designed for traditional, single-view videos. In addition, despite the significant practical interest in mobile multicast as mentioned above, current commercial base stations do not yet efficiently support streaming immersive multimedia content [29]. We consider mobile multicasting of 360-degree videos, which is a more complex problem than multicasting single-view videos. This is not only because 360-degree videos require multiple times the amount of bandwidth of single-view videos, but also because 360-degree videos offer unprecedented interactivities between users and the content. Specifically, users watching a 360-degree video can dynamically change/select their viewing direction, resulting in an immersive and engaging experience, but creating challenges for the network

that needs to support this interactivity in addition to handling user mobility and varying channel conditions.

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 [23, 27]. And 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 many mobile phones and tablets. Furthermore, mobile data traffic carrying immersive content is expected to grow 11 folds by 2021 relative to 2016 [16]. Therefore, the problem of efficiently delivering immersive multimedia content is of practical importance.

In this thesis, we present a comprehensive solution (called VRCast) for the problem of *live* streaming of 360-degree videos to mobile users. Our solution supports user interactivity (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, achieves high spectral efficiency of the expensive wireless link, and runs in real time.

We evaluate VRCast using trace-driven simulations and implementation in a real LTE testbed. Our results 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% and reduces the variation in the spatial quality by up to 53% and increases the energy saving for mobile devices by up to 250%. In addition, we develop an LTE testbed and show that our system adapts to different user activity patterns, scales well with large number of users and does not cause rebuffering events at user side.

1.2 Problem Statement

The problem addressed in this thesis is to optimally multicast a live 360-degree video of a popular event to a large number of mobile users with dynamic channel conditions watching different viewports. The 360-degree video to stream is in the form of a grid of $N \times M$ tiles, so that we can control the quality of each part of the video individually. Our objective is to maximize the average viewport quality for all users given a limited number of radio resource blocks (RBs) in the cellular network. This is a fairly complex problem, since other metrics should also be considered as they affect the user experience, e.g., temporal/spatial smoothness of rendered viewport tiles, energy of mobile devices, fairness among mobile users, and spectral efficiency.

We propose decomposing the complex problem of mobile multicast streaming of 360-degree videos into two sub-problems. In the first sub-problem, we divide users into multicast groups to maximize the assigned bitrate for each group. Then, these bitrates are used as a budget in the second sub-problem where we divide them among tiles to optimize video quality. This decomposition is

intuitive and allows us to carefully model and solve each sub-problem optimally. In addition, it allows us to solve each sub-problem at a different time scale, which is an important advantage for our approach as it offers more flexibility. Notice that the first sub-problem deals with optimizing network parameters and the second optimizes the 360-degree video streaming parameters. Both clearly can vary at different time scales.

1.3 Thesis Contributions

The contributions of the thesis can be summarized as follows:

1. A comprehensive solution for the problem of *live* streaming of 360-degree videos to mobile users over LTE networks.
2. Extensive trace-driven simulations showing that VRCast outperforms the state-of-the-art algorithms across multiple performance metrics.
3. An LTE testbed that supports *multicast* streaming of 360-degree videos. Using the testbed, we show the practicality of VRCast.

1.4 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 provides background information needed to understand the concepts discussed in this thesis and summarizes the related work in the literature. Chapter 3 describes the considered system model and states the addressed problem. Chapter 4 presents the mathematical formulation of the problem and our proposed solution. Chapter 5 presents our trace-driven simulations and comparisons against other works in the literature and describes our LTE implementation and analyzes various aspects of VRCast. Chapter 6 concludes the thesis.

Chapter 2

Background and Related Work

In this chapter, we present a brief background on 360-degree videos, adaptive streaming, and LTE networks. Then, we summarize the related work in the literature.

2.1 Background

2.1.1 360-degree Videos

In 360-degree videos, a view in every direction is recorded at the same time. These videos are shot using omnidirectional cameras or a collection of cameras, and then the output videos are stitched together to get the final spherical view of the 360-degree video. During playback, the viewer has control of his viewport using his mobile phone or a head mounted display (HMD) such as GearVR and Oculus Rift. The controls can be simplified using Euler angles (pitch, yaw, roll) which corresponds to rotations around the (x, y, z) axes, respectively. The user viewport can be defined as their head rotation angles, and the Field of View (FOV) of the HMD or the mobile phone. This introduces one of the main challenges in 360-degree video streaming which is wasting bandwidth on video parts that will not be viewed by the user. In addition, the bandwidth requirements for streaming 360-degree videos are more than the normal 2D videos (usually 3 or 4 times more).

There are multiple projections for the 360-degree videos in order to transform their spherical nature into 2D planar format in order to use the usual encoding techniques applied on 2D videos. Multiple sphere-to-plane mappings have been proposed in the literature, including equirectangular [12], cubemap [44], pyramid [36], offset-cubemap [34], rhombic dodecahedron [24], and barrel projections [35].

Our solution can be applied to any of the projections used for 360-degree videos as long as we have a set of non-overlapping regions that covers the whole video frame. Also, there should be a way to get the area of the overlap between any viewport and each region. In our work, we use the simplest projection, i.e, equirectangular projection which is the most commonly used mapping. It can be described as unwrapping a sphere on a 2D rectangular plane with the dimensions $(2\pi r, \pi r)$, where r is the radius of the sphere. Equirectangular projection is widely supported and easily viewable even with no special players. On the other hand, one of its main drawbacks is the amount of redundant

pixels, specially around the top parts of the sphere, which will waste the user's limited bandwidth in a streaming scenario. Other projections try to address this drawback [44, 36, 35].

We chose equirectangular projection as defining the non-overlapping regions covering the frame is straightforward and also supported by HEVC encoder. The frame is divided into a $N \times M$ grid of rectangular tiles. The motion-constrained tile set feature of HEVC [41] can be directly used to produce these tiles without any synchronization problems between tiles. Also, a single decode can be used to output the final video that consists of multiple tiles.

2.1.2 Adaptive Streaming

Most current multimedia streaming services have adopted a pull-based approach for delivering media content over the Internet using the widely popular Hyper-Text Transfer Protocol (HTTP). Adaptive streaming over HTTP is introduced to support instant streaming and bitrate adaptation. Video content is divided temporally into a number of non-overlapping segments, each corresponds to a few seconds of the media. Each segment is encoded at multiple bitrates and/or resolutions, called representations. A manifest file is used to describe the different representations of segments and the location and duration of each segment. This manifest file is shared with users interested in streaming the video. Then, users continuously monitor the available resources, such as available bandwidth and the buffer capacity, and dynamically select the right encoding bitrate for the next segment. After that, users generate requests to download these segments. There are different bitrate adaptation algorithm on the client side to choose the suitable segment bitrate such as: Festive [31], BBA [30], Pensieve [40], and MPC [58].

There are various implementations of adaptive streaming over HTTP, including Adobe HTTP Dynamic Streaming (HDS), Microsoft Live Smooth Streaming, and Apple HTTP Live Streaming (HLS). An international standard for this streaming class is called Dynamic Adaptive Streaming over HTTP (DASH) [50]. In DASH, the manifest file is known as a media presentation description (MPD) file and it describes the properties and URLs of the content and its segments.

MPEG DASH standard [42] proposes two methods for low latency streaming, which is crucial for live streaming. The first method is based on very short segment durations, which is the most commonly used approach to reduce streaming latency. DASH clients define their buffer capacities according to the number of segments, ignoring the segment duration. Therefore, clients start playback when a certain number of segments are received. Accordingly, decreasing the segment size also reduces the buffer duration before playback starts.

The second method is based on dividing each segment into multiple fragments. Each fragment can be encoded and packaged before the whole segment is available. On the client side, HTTP chunked delivery is used to request individual fragments. Essaili et al [19] presented a prototype for a low latency live streaming system based on DASH using multi-fragment segments and HTTP chunked delivery.

In our work, we use DASH to stream the 360-degree videos from the server to the base station. Then the base station forwards the data to the mobile users using FLUTE [51, 46], which is the

standard protocol used for multicast in LTE networks. Note that our system is independent of the different streaming protocols as long as the video is available in the form of independent tiled segments with different qualities.

2.1.3 Streaming 360-degree videos

The traditional method of streaming 360-degree videos is to use the 2D streaming adaptative bitrate algorithms built on DASH. The whole video is encoded uniformly with the same target bitrate. This leads to a big waste in bandwidth because the user only views about 30% of the whole frame. Two approaches have been proposed in the literature to improve 360-degree video streaming: region-of-interest (ROI) streaming and tiling.

In ROI-based streaming, the 360-degree video is projected onto a geometric structure such as a pyramid [36] or cube map [34] such that more bits are allocated to the viewport. Multiple versions of the video are created for each possible viewport. During streaming, users choose the version that overlaps the most with their current viewport.

In tile-based streaming, a 360-degree video is divided into a grid of $N \times M$ tiles. A higher bitrate is given to tiles in the viewport while a lower bitrate is given to other tiles. MPEG-DASH Spatial Relationship Description (SRD) [45] supports tiled streaming, because SRD can describe a video as a spatial collection of synchronized videos. Using SRD, we can stream to users the area they are currently viewing with the highest quality possible based on their available bandwidth. In addition, the motion-constrained tile set (MCTS) feature of HEVC [41] has been investigated in [37, 49, 61] to reduce the transport complexity and reduce synchronization problems between tiles such that a single decoder can be used.

In our work, we tile the videos and encode each tile in different bitrates using HEVC encoding. We describe the available representations for each tile in the MPD using SRD. Then, the server decides what representation to send for each tile.

Wang et al. [54] did subjective experiments to study the impact of having different qualities for tiles in the viewport [54]. They conclude that in most cases, by mixing tiles from HD $1920 \times 1080p$ stream and $1600 \times 900p$ stream, 14%-20% bandwidth can be saved without any noticeable perceptual quality loss.

2.1.4 Cellular Networks

Our system is independent of the cellular network technology, provided that we can schedule physical network resources with different capacities to users with various channel conditions. In our work, we use LTE as a sample cellular network. We can extend our work to 5G because both of them use Orthogonal frequency-division multiplexing (OFDM). The high-level network architecture of LTE is comprised of three main components: User Equipment (UE), Evolved UMTS Terrestrial Radio Access Network (E-UTRAN), and Evolved Packet Core (EPC). EPC communicates with packet data networks in the outside world such as the Internet. E-UTRAN handles the radio communica-

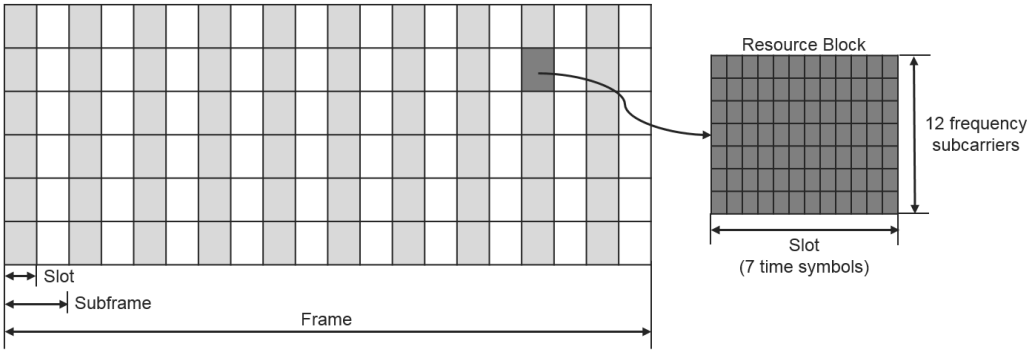


Figure 2.1: LTE frame.

tions between UEs and EPC and has one component, the evolved base stations, called eNodeB or eNB.

As shown in Figure 2.1, the LTE downlink channel is divided into fixed time frames, each spanning 10 ms. Each frame is further divided into 10 sub-frames, each spanning 1 ms and containing two slots (0.5 ms each). The base station (eNB) transmits each subframe using OFDM, which divides available radio resources into grids in both time and frequency domain. Each grid cell (spanning 15 kHz $\times$ 66.7 μs) is called a resource element (RE). The eNB schedules radio resources at the granularity of a physical resource block (RB) comprising multiple resource elements. Each RB spans half a subframe (0.5 ms) in the time domain and 12 OFDM subcarriers (180 kHz) in the frequency domain. UEs are dynamically allocated non-overlapping sets of RBs depending on their channel conditions.

To accommodate the time-varying radio channel conditions of UEs, LTE uses a link adaptation method known as adaptive modulation and coding (AMC) which adapts the modulation scheme and code rate based on the channel's signal-to-noise ratio (SNR). The quality of a channel is periodically measured at the UE and sent to the eNB in the form of so-called channel quality indicators (CQIs). The modulation and coding scheme (MCS) used for the resource blocks assigned to a UE is then chosen based on the reported CQI value such that the block error rate is less than a threshold. Higher MCS modes require good channel quality and lead to higher number of bits per resource block.

In order to improve energy efficiency of mobile users, LTE specifies the Discontinuous Reception Mechanism (DRX) [5]. Enabling DRX allows UEs to wake up and sleep in a periodical manner. The sleeping/waking period is multiple of subframes (1ms). Scheduling RBs has to consider the DRX parameters to ensure that data is sent when the receiver is awake. The energy consumption in the sleep time is much less than wake up time.

2.1.5 Evolved Multimedia Broadcast Multicast Services (eMBMS)

The 3GPP standard specifies the Evolved Multimedia Broadcast Multicast Service (eMBMS) for multicast over LTE networks [2]. eMBMS enables LTE cellular networks to deliver video streams

over multicast, which allows a streaming server to substantially reduce the wireless network load by serving mobile devices interested in the same video stream using a single multicast session.

A multimedia multicast service consists of four network elements: Broadcast/Multicast Service Center (BMSC), Gateway (GW), Mobile Management Entity (MME), and Multi-cell/multicast Coordination Entity (MCE).

First, the service center (BMSC), which is located within the core network, is responsible for authenticating and authorizing the content providers, managing the charging process, and controlling the overall configuration of data flow through the core network. Second, the eMBMS gateway is considered as a logical node that helps in multicasting any IP packet generated from BMSC to all base stations located in a certain area. Moreover, the gateway handles further session control signaling via the mobile management entity (MME). MME plays an important role in performing a number of controlling procedures such as user tracking, paging, and bearer activating. Finally, MCE ensures the full functionality by performing the time synchronization as well as coordinating the usage of the same radio resources and transmission parameters across all cells belonging to a particular area.

Multicast services are offered on a time-shared basis with unicast connections. The frame structure in an LTE network is subdivided into 10 equal sub-frames whose lengths are equal to 1 millisecond. Some of these sub-frames (numbered 0, 4, 5, and 9) are reserved for unicast connections and cell specific information. Any or all of the remaining six sub-frames may be allocated to the multicast service. A mobile terminal is informed about which sub-frames are assigned to its multicast session via a broadcast channel and this allocation can be changed dynamically at specified intervals.

2.2 Related Work

2.2.1 Mobile Video Streaming

Mobile video streaming of traditional single-view video streams has been extensively studied in the literature, for both unicast and multicast. For example, Yu et al. [60] utilize scalable video coding (SVC) to optimize the energy consumption of mobile devices, whereas Almowuena et al. [8] divide users into groups based on their channel conditions to achieve the same goal. Chen et al. [14] discuss fairness among users in mutlicast groups and present a resource allocation method for user grouping. Chen et al. [15] introduce a unicast framework for adaptive streaming of single-view videos over LTE networks. None of these works, however, addresses the characteristics of 360-degree videos, such as the large size of the streams and the user interactivity with the content.

Xie et al. [57] present piStream, a DASH-compatible adaptive video streaming framework that exposes LTE's physical layer information to facilitate video rate adaptation. piStream's PHY-informed design enables a more accurate bandwidth estimation and agile video rate adaptation.

2.2.2 360-degree video streaming

Few recent works addressed various aspects of 360-degree video streaming, especially in the unicast model, e.g., [62, 47, 28, 56, 17]. Two common themes can be identified in previous works: tiling and region-of-interest (ROI) streaming.

In tile-based streaming, a 360-degree video is projected on an equirectangular map [59] and divided into an $N \times M$ 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. One of the earliest solutions for adaptive 360-degree video streaming is proposed in [7]. The authors formulate and solve a bandwidth-optimal system for selecting the quality of tiles that maximizes the quality of viewport depending on available bandwidth. Another tile-based method is proposed in [48] that examines three prediction approaches to predict the user's future viewport, and selects the optimal sequence of tiles which minimizes bandwidth consumption. The presented results show that viewport can be predicted rather accurately for the next second. In addition, the recent High Efficiency Video Coding (HEVC) standard supports tiling [61], which further helps in adopting tile-based streaming.

Graf et al. [28] discuss streaming of 360-degree videos over HTTP by exploring different tiling patterns. We use the analysis in [28] to select tile sizes in our work. Petrangeli et al. [47] construct a 360-degree streaming framework over HTTP/2, which aims to reduce the high bandwidth requirements and storage costs of current streaming solutions for 360-degree videos. Nasrabadi et al. [43] investigate a buffer-efficient approach using scalable video coding in 360-degree video streaming and provide a comprehensive buffer analysis. Xie et al. [56] propose a probabilistic tile-based adaptive streaming system. They apply a target-bufer-based control algorithm to ensure continuous playback within a small buffer and construct a probabilistic model to cope with the viewport prediction error. Xiao et al. [55] compute the optimal tiling to minimize storage and bandwidth. They formulate these storage and bandwidth concerns as an integer linear programming problem. Their experiments show that non-uniform tiling solutions can significantly outperform existing tiling schemes.

In ROI-based streaming, the 360-degree video is projected onto a geometric structure such as a pyramid [17] or cube map [59], where regions of the structure represent different user viewports. For each viewport, a video version is created where more bits (quality) are allocated to parts of the 360-degree video in the viewport and less bits are allocated to the other parts. The versions are stored on the server, and during streaming, the version that aligns the most with the user's current viewport is served to that user. Zhou el al. [62] analyze the ROI-based streaming approach used by Facebook and its impact on user experience and bandwidth consumption. Corbillon et al. [17] investigate the effect of various projections and quality arrangements on the video quality displayed to the user.

Multicast of 360-degree videos has received much less attention in the literature thus far. We are only aware of one recent algorithm that considers adaptive streaming of VR content over LTE networks which is referred to as MVR (Multicast of Virtual Reality content) [6]. MVR proposes a heuristic algorithm that divides users into subgroups based on their channel conditions and tile

weights, and determines the bitrate for each tile in each subgroup. Unlike MVR, VRCast divides the problem into two subproblems, one is related to the mobile channel conditions and the other considers viewports of users. This division allows us to solve each problem optimally and efficiently in real time. However, MVR considers both aspects together and this leads to a suboptimal solution that can not run in real time.

In our experiments, We compare VRCast against MVR as well as the closest work for single-view video streaming [14] after adding some extensions to support 360-degree videos. The authors of [14] propose a multicast grouping that maximizes the proportional fairness. They provide a mathematical formulation for their problem and solve it optimally using a dynamic programming approach.

Chapter 3

System Model and Problem Definition

This chapter explains the considered system model and its components. It also defines the problem addressed in this thesis and its challenges.

3.1 System Model

We consider live streaming of popular 360-degree videos over 4G/5G cellular networks that support multicast services, such as eMBMS in LTE networks [38]. As shown in Figure 3.1, our scenario is similar to regular mobile multimedia streaming, where a content server provides a video stream, through high-speed wired links, to a Broadcast Multicast Service Center (BM-SC) which announces streaming services to multiple users connected to one or more cellular base stations. Each base station serves users within its range. Our work manages the wireless channel between each base station and its users to optimize the delivery of 360-degree videos. The radio resources of this wireless channel are divided across time and frequency. 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 [4].

The channel conditions of each mobile user fluctuate over time due to mobility and channel impairments such as shadowing, interference, and multi-path fading. Mobile users periodically report their channel quality indicator (CQI) to the associated base station. Each CQI value is mapped to a modulation and coding scheme (MCS), which determines the bitrate per RB for each user. Higher MCS modes require good channel qualities and lead to higher resource block capacity (bits/RB). On the other hand, lower MCS modes are more robust and usable for diverse (both strong and weak) channel qualities. The MCS mode must be chosen to accommodate all users in a multicast group, which means that the MCS mode must be chosen based on the user with the worst channel conditions. Thus, putting all users within a cell in a single multicast group may not yield the best performance. The first aspect of our problem is how to divide mobile users into multiple multicast groups to maximize the average bitrate received while maintaining fairness.

To save the energy of mobile devices, the base station transmits the video data in bursts, which allows mobile devices to wake up the reception component of the device to receive the data during

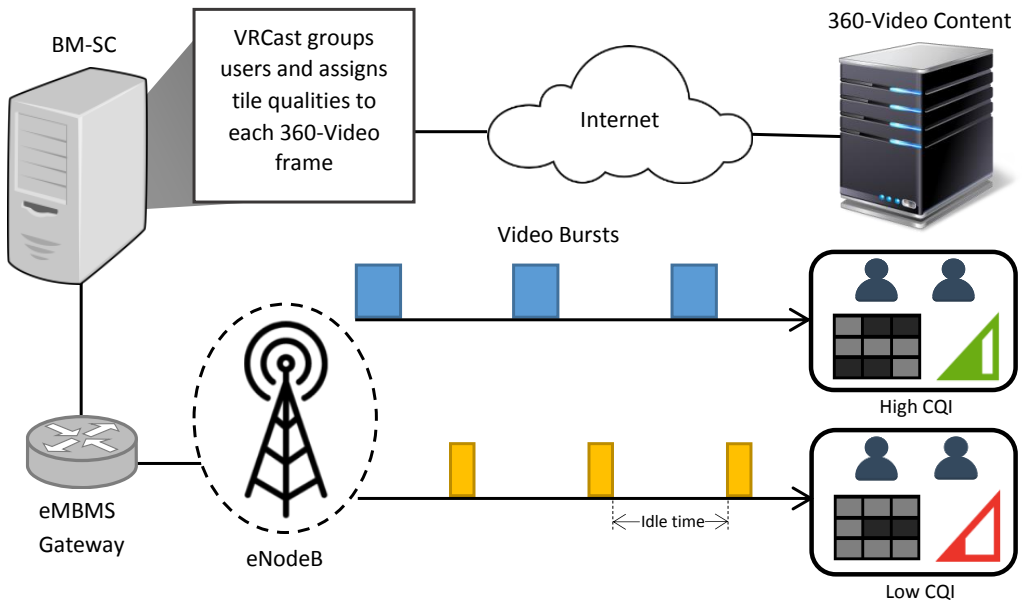


Figure 3.1: A high-level illustration of the considered model.

the burst and put it in low-energy (idle/sleep) mode when there is no burst. The second aspect of our problem is how to construct bursts and compute sleep durations to maximize the energy saving of mobile users. In LTE, the Discontinuous Reception Mechanism (DRX) is used to implement the computed bursts to save energy [5].

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 can 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. Users can be watching different viewports at the same time. The viewports of users are periodically sampled and sent back to the base station. The frequency of sending this data is low for each user, once every few seconds or more according to the number of users. On the server side, a prediction module is used to estimate future viewports.

Each 360-degree video is divided into a grid of $N \times M$ tiles. The video is also temporally divided into segments, each is in the order of one to a few seconds. Each tile in each segment is encoded at multiple quality representations. Information about segments and their bitrates are stored in Media Presentation Description (MPD) files, according to the Spatial Representation Description (SRD) feature of the widely-used DASH protocol [45]. The final aspect of our problem is to determine which tiles to transmit and the DASH representation for each tile to maximize the quality for all users and maintain spatial smoothness across tiles.

3.2 Problem Definition

The problem addressed in this thesis is to optimally multicast a live 360-degree video of a popular event to a large number of mobile users with dynamic channel conditions watching different viewports such that the average viewport quality for all users is maximized given a limited number of resource blocks (RBs) in the cellular network. This is a fairly complex problem, as a solution for it not only needs to maximize the utilization of the wireless resources, but it also should optimize the perceived quality for users in terms of viewport bitrates and temporal/spatial smoothness of the rendered viewport tiles. Furthermore, the solution needs to save the energy of mobile devices, and to support the user interactivity with 360-degree videos.

To illustrate the complexity of this problem, we discuss various trade offs. First, consider a simple approach that puts all users in the same multicast group. In this case, users with poor channel conditions impose a restriction on the modulation and coding scheme 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. Thus, we need to find an optimal number of multicast groups to achieve the best possible quality given the available resources.

Second, consider streaming different viewports of 360-degree videos using a given bitrate budget. Unlike traditional single-view videos, only a fraction of each 360-degree video frame can be watched (viewport) by each user. Thus, the quality of experience is heavily impacted by the quality of the viewport. In addition, some viewports are more popular (watched by more users) than others. Therefore, it is important to carefully allocate the available bitrate to tiles to achieve high quality within the viewport and across successive viewports.

Third, once the bitrate is decided for each tile, the bits need to be transmitted over the wireless channel using the resource blocks. Resource blocks span two dimensions: frequency and time. Naively choosing resource blocks for tiles may spread them over longer time than necessary, and thus reduces the opportunity for energy saving for mobile devices (as the reception component needs to be active during the entire transmission time of the resource blocks). Therefore, it is important to carefully construct bursts of data for different parts of the 360-degree video to save energy.

Chapter 4

Proposed Solution

This chapter presents our proposed solution (VRCast) for 360-degree video multicast streaming over LTE networks. It further explains the steps of our solution by an illustrative example.

4.1 VRCast Solution Overview

We propose decomposing the complex problem of mobile multicast streaming of 360-degree videos into two sub-problems. In the first sub-problem, we divide users into multicast groups to maximize the assigned bitrate for each group. Then, these bitrates are used as a budget in the second sub-problem where we divide them among tiles to optimize video quality. This decomposition is intuitive and allows us to carefully model and solve each sub-problem optimally. In addition, it allows us to solve each sub-problem at a different time scale, which is an important advantage for our approach as it offers more flexibility. Notice that the first sub-problem deals with optimizing network parameters and the second optimizes the 360-degree video streaming parameters. Both clearly can vary at different time scales.

We note that our solution manages the last hop (wireless link between base stations and mobile users). Thus, it does not introduce any additional delay from the source server to the base station. The content is assumed to be divided into tiles and encoded in multiple representations using standard DASH; this happens at the server and typical for current live streaming services. Our solution at the base station divides users into groups and selects the quality representations for different tiles. This occurs in real time for each scheduling window Δ (in the order of 1–2 sec). Meaning, while users are consuming data for window at time t_1, the decisions for window $t_1 + \Delta$ are computed. Since our solution is efficient and takes much less time ($\approx$ milliseconds) than the window size to compute, we do not introduce any additional delay into the live streaming session, which is crucial for such sessions.

It is also important to note that each multicast group receives *all* tiles, but tiles have different qualities depending on their importance. This is indicated in Figure 3.1 by different shades of the tiles in the two multicast groups. The availability of all tiles supports fast switching, especially for unusual random viewport changes occasionally made by some users.

Symbol	Description
C	Number of MCS modes
K	Number of multicast groups
M	Number of mobile users
Q	Number of quality representations
R	Total 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
c_i	MCS for user i (bits/RB)
$\hat{c}_k$	Minimum MCS for multicast group k (bits/RB)
$\mathbb{G}_k$	Set of users in multicast group k
$w_{t,i}$	Weight of tile t according to the viewport of user i
$W_{k,t}$	Average weight of tile t for multicast group k
x_k	RBs assigned to multicast group k
$y_{k,t,q}$	Indicator variable showing if quality q is assigned to tile t for multicast group k

Table 4.1: Symbols used in the thesis.

The details of each step are explained in the following subsections, followed by an illustrative exampling showing all steps of VRCast. Symbols used in this thesis are listed in Table 4.1.

4.1.1 Grouping Users

Given a budget of resource blocks R and a number of mobile users M with different channel conditions (and hence MCS modes), we would like to optimally partition users into multicast groups and assign resource blocks to each group, such that the average bitrate received by users is maximized while maintaining fairness among users in different groups.

The mathematical formulation of this problem is given in Eq. (4.1), which is a two-level nested optimization problem. The outer optimization computes the optimal grouping of users to maximize the average bitrate as shown in Eq. (4.1a). The inner optimization computes the optimal allocation of resource blocks to multicast groups that maximizes the proportional fairness, which is defined as the sum of the log of bitrates in Eq. (4.1e). The inner optimization is similar to the ones presented in [14], and it has an analytic solution which distributes the resource blocks among groups proportional

to the number of users in each group. That is, the optimal solution for the inner optimization problem is given by $x_k^* = |\mathbb{G}_k|/M \times R$.

$$\max_{\mathbb{G}_k, x_k} \quad \sum_{k=1}^{K} \frac{|\mathbb{G}_k|}{M} \times \frac{\hat{c}_k \times x_k}{S} \tag{4.1a}$$

$$\text{subject to} \quad \hat{c}_k = \min_{i \in \mathbb{G}_k} c_i \qquad \forall k \leq K \tag{4.1b}$$

$$|\mathbb{G}_1 \cup \mathbb{G}_2 \cup \cdots \cup \mathbb{G}_K| = M \tag{4.1c}$$

$$\mathbb{G}_k \cap \mathbb{G}_l = \phi \qquad \forall k, l \leq K, k \neq l \tag{4.1d}$$

$$x_k = \arg\max_{x_k} \left\{ \sum_{k=1}^{K} |\mathbb{G}_k| \log(\frac{\hat{c}_k \times x_k}{S}) : \sum_{k=1}^{K} x_k \leqslant R \right\} \tag{4.1e}$$

$$\text{variables} \quad K, x_k, \mathbb{G}_k \tag{4.1f}$$

The objective of the outer optimization problem, Eq. (4.1a), is to maximize the average bitrate for all users. 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 user's group $\hat{c}_k$ and the number of RBs allocated to this group according to the inner optimization problem x_k^*. The constraint in Eq. (4.1b) 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. (4.1c) and Eq. (4.1d) ensure that each user belongs to only one multicast group.

We design an efficient algorithm using dynamic programming to compute the optimal solution, which is shown in Algorithm 1. The basic intuition used in this algorithm is that users with similar MCS values are more likely to be in the same group than users with quite different MCS values. Therefore, The algorithm starts by sorting users based on their MCS values. It then forms MCS groups consisting of users having the same MCS value. Then, we can think of the problem as placing partitions between users, and we want to decide the optimal number and positions of these partitions. The best solution for k groups can be found by considering the optimal solution for $k-1$ groups plus the utility of the new group. The dynamic programming array used $U(K, l, i)$ is the total utility for the first i MCS groups forming l partitions, when there are a total of K multicast groups such that $U(K, l, i) = \max_{\forall j<i} U(K, l-1, j) + utility(K, \mathbb{G}_l)$, where j is the number of MCS groups forming $l-1$ partitions.

After finding the optimal grouping and allocation of RBs, we schedule these RBs in the time-frequency grid to save the energy of mobile users. Since there are no RBs shared between any two multicast groups, the optimal solution is to arrange the RBs of each multicast group contiguously, i.e., no RBs from a multicast group is allocated in the middle of the RBs of another group. Then, we form data bursts based on the grouped RBs. A burst is specified by the start and end times of the

Algorithm 1: Group users and assign RBs.

Function *GroupUsers*
  Input : c_i sorted, N
  Output: $K^*, \mathbb{G}_l^*, x_l^*$
  for $K = 1 : N$ **do**
    // Partition the 1^{st} multicast group $\mathbb{G}_1$
    for $i = 1 : N - (K - 1)$ **do**
      $U(K, 1, i) = GetGroupUtility(K, \{1, ..., i\})$
    end
    // Partition the l^{th} group $\mathbb{G}_l$
    for $l = 2 : K$ **do**
      // Partition users till MCS i
      for $i = l : M - (K - l)$ **do**
        // Partition users from MCS greater than j
        for $j = l - 1 : i - 1$ **do**
          $\mathbb{G}_l = \{j + 1, \ldots, i\}$
          $u = U(K, l - 1, j) + GetGroupUtility(K, \mathbb{G}_l)$
          $U(K, l, i) = max(U(K, l, i), u)$
        end
      end
    end
  end
  // Backtrack to find optimal $\mathbb{G}_l^*$ and x_l^*
  $K^*, \mathbb{G}_l^*, x_l^* = constructSolution(U)$
Function *GetGroupUtility (K, $\mathbb{G}_l$)*
  $x_l^* = \dfrac{|\mathbb{G}_l|}{M} \times R$ // Optimal RBs allocation
  $\hat{c}_l = \min_{i \in \mathbb{G}_l} c_i$
  return $u_l = \dfrac{|\mathbb{G}_l|}{M} \times \dfrac{\hat{c}_l \times x_l^*}{S}$

grouped RBs. In LTE networks, this is implemented by setting the DRX parameter, which instructs the reception component of mobile devices to wake up only during the transmission period of their group's RBs and sleep otherwise.

Time Complexity and Notes. It is straightforward to show that the time complexity of the proposed user grouping algorithm is $O(C^4)$, where C is a *constant* that represents the maximum number of MCS modes, and it *does not* depend on the number of mobile users in the system. In current LTE networks, the maximum value of C is 30 [38]. Thus, the proposed algorithm can easily run in real time.

We note that we group users based on their channel conditions only. This is to maximize the utilization of the available wireless resources. We did try to further classify users based on their current viewports in addition to channel conditions. This, however, resulted in worse performance in terms of the achieved video quality and energy saving, because users with diverse channel conditions but sharing similar viewports may end up in the same group, which forces the whole group to use lower MCS mode (i.e., lower bitrate) and hence reduces the quality for all users. Our user grouping method does not mean that users will be receiving data that they are not interested in, since all tiles are sent to each group to support fast interactivity. The importance of viewports is considered in assigning different qualities to tiles.

4.1.2 Assigning Quality Representations to Tiles

The second step of VRCast is to distribute the computed resource blocks in step 1 among tiles in the different multicast groups. That is, for each multicast group $\mathbb{G}_k$ with MCS mode $\hat{c}_k$ and x_k^* RBs assigned to it, we need to assign a quality representation q to each tile 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. The mathematical formulation of this problem is given in Eq. (4.2).

The formulation in Eq. (4.2) computes the optimal quality representation for each tile. Two alternative utilities could have been adopted in step 2. The first one is the weighted sum of tiles bitrates (WSTB). WSTB maximizes the average viewport bitrate. However, it does not consider the spatial smoothness of the tiles in the viewport. In order to consider spatial smoothness, We use the weighted product of tiles bitrates (WPTB) as our utility. WPTB is defined as $\prod_t b_{t,q}^{W_{k,t}}$, which is equivalent to $\sum_t W_{k,t} \times \log(b_{t,q})$. WPTB improves spatial smoothness because maximizing the product of tiles bitrates leads to variance minimization among them. The constraint in Eq. (4.2b) restricts the available number of resource blocks for multicast group $\mathbb{G}_k$ to x_k^*. The constraint in Eq. (4.2c) calculates the weight of each tile as the aggregate of the weights of tiles across sampled users in the group. The tile weight for each user is defined as the percentage of overlap between the tile and the viewport of the user. The constraints in Eq. (4.2d) and Eq. (4.2e) ensure that only one

quality representation is assigned to each tile. Note that $y_{k,t,q}$ is a decision variable that determines whether the quality q is assigned to tile t for multicast group k.

$$\max_{y_{k,t,q}} \quad \sum_{t=1}^{T}\sum_{q=1}^{Q} W_{k,t} \times \log(b_{t,q}) \times y_{k,t,q} \tag{4.2a}$$

$$\text{subject to:} \quad \sum_{t=1}^{T}\sum_{q=1}^{Q} \left\lceil \frac{b_{t,q}}{\hat{c}_k} \right\rceil \times y_{k,t,q} \leqslant x_k^* \tag{4.2b}$$

$$W_{k,t} = \sum_{i \in \mathbb{G}_k^*} w_{i,t} \tag{4.2c}$$

$$\sum_{q=1}^{Q} y_{k,t,q} = 1, \forall t \tag{4.2d}$$

$$y_{k,t,q} \in \{0,1\}, \forall t, q \tag{4.2e}$$

$$\text{variables} \quad y_{k,t,q} \tag{4.2f}$$

It is straightforward to show that the problem in Eq. (4.2) falls in the category of Multi-choice Knapsack problems [33], which is NP-Complete. Each tile is equivalent to a class and the available quality representations for each tile are equivalent to the items in each class. Constraints (4.2d) and (4.2e) ensures that only one quality is picked from every tile. Each quality has a utility ($W_t log(b_{t,q})$) defined in the objective function (4.2a). The cost of each quality is the number of RBs required to stream the chosen quality, which equals to $\lceil \frac{b_{t,q}}{\hat{c}_k} \rceil$ defined in (4.2b).

Algorithm 2 shows our solution for the problem in Eq. (4.2). We design an algorithm to solve the optimization problem in Eq. (4.2). The algorithm first assigns a minimum quality representation for each tile, 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 by considering 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 $R(t,q)$ is the number of RBs needed to stream tile t with quality q.

Time Complexity. The proposed algorithm for assigning qualities to tiles terminates in $O(T \times Q \times x_k^*)$, where T is the number of tiles, Q is the number of quality representations, and x_k^* is the number of RBs.

To shed some light on this time complexity, we mention the practical ranges of the three parameters T, Q, and x_k^*. The number of tiles T is in the order of few tens. For example, recent works that adopt tiling, e.g., [47, 28, 56], set T around 50. The number of DASH quality representations Q is typically less than 10 in practice. For x_k^*, the maximum total number of resource blocks for *all* multicast groups in one second (the typical scheduling window) is less than 60,000 in LTE [38, 3].

Algorithm 2: Assign quality representation to each tile.

Function *AssignTileQuality*
  Output: $b_{k,t} \leftarrow$ Bitrate for each tile in group k.
  $\forall t, b_{k,t} = b_{t,1}$ // Assign min bitrates to tiles
  $\forall \tau, V(0, \tau) = 0$
  for $t = 1 : T$ **do**
    for $\tau = \frac{b_{t,1}}{\hat{c_k}} : x_k^*$ **do**
      for $q = 1 : |b_t|$ **do**
        if $\tau - \frac{b_{t,q}}{\hat{c_k}} \geq 0$ **then**
          $u_{k,t,q} = W_{k,t} \times log(b_{t,q})$
          $u = V(t-1, \tau - \frac{b_{t,q}}{\hat{c_k}}) + u_{k,t,q}$
          $V(t, \tau) = max(V(t, \tau), u)$
        end
      end
    end
  end
  // Backtrack to find optimal $b_{k,t}$
  $b_{k,t} = constructSolution(V)$

Thus, for each multicast group, the number of resource blocks x_k^* is in the order of a few tens of thousands. Putting all numbers together indicate that our algorithm can easily run in real time.

4.2 Illustrative Example

For illustration, we explain the steps of our solution using the following simple scenario. A budget of 54 RBs is available in a time-frequency grid of 6 time slots and each slot has 9 RBs. 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 per second respectively. The channel condition (MCS value) and viewport tiles of each user is shown in Table 4.2. 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 bits capacity of each RB (bits/RB). However, for simplicity, we define MCS values as number of bits per RB.

Applying step 1, 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 4.3. 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 (row 4) assigned to each group 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

User Index	MCS	Viewport tiles
User 1	1 bit/RB	1, 2
User 2	1 bit/RB	3
User 3	2 bits/RB	2
User 4	2 bits/RB	1, 3
User 5	2 bits/RB	1, 2
User 6	2 bits/RB	2, 3
User 7	4 bits/RB	1, 2
User 8	4 bits/RB	2
User 9	4 bits/RB	2, 3

Table 4.2: MCS values and viewport tiles of users in the example.

	G1	G2		G3		G4		
1. M/C groups	[1,2,4]	[1,2]	[4]	[1]	[2,4]	[1]	[2]	[4]
2. No. of users	9	6	3	2	7	2	4	3
3. Min. MCS	1	1	4	1	2	1	2	4
4. No. of RBs	54	36	18	12	42	12	24	18
5. No. of bits	54	36	72	12	84	12	48	72
6. Bitrate	9	6	12	2	14	2	8	12
7. Avg. bitrate	9	8		**11.33**		8		

Table 4.3: Possible groupings in the example.

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

Table 4.4: RBs required to send each quality representation.

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

Table 4.5: Feasible quality assignments for group 2.

bitrates according to the number of users per group (row 2). As shown in 4.3, grouping 3 is optimal and maximizes the utility.

In step 2, we have 3 tiles and 3 quality representations for each tile of bitrates 4 (L), 20 (M), and 32 (H) bits. Table 4.4 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 4.5 shows the feasible quality assignments with highest utilities for group 2.

Chapter 5

Evaluation

This chapter presents an extensive evaluation using trace-driven simulation and an LTE testbed to assess the performance of VRCast compared to the closest works in the literature.

5.1 Simulation Setup

5.1.1 360-degree Videos and User Interactivity Traces

We used and combined two datasets of 360-degree videos in our experiments to cover a wide variety of content and user viewing behavior. The first dataset [11] contains 16 videos at 4K 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 recorded every 0.1 second, in the form of Euler angles.

The second dataset [18] contains 7 videos at 4K 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 are recorded in the form of the Hamilton quaternions representation. 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 recorded every 0.1 second. To exercise different network conditions and user mobility with realistic speeds, we generate 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 [6], and encoded each tile at five quality representations using Kvazaar, [53], an open source HEVC 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 [45] to segment the video into one-second segments and generate the MPD DASH manifest.

During simulation, each user randomly selects one of our traces and starts interacting with the video (i.e., changing viewports) according to the trace. The decision for a segment at time t is based on a sample of 10% of the viewports at $t - 0.5$, after applying the Auto Regressive Moving Average (ARMA) prediction module proposed in [25] to predict their future viewport at time t.

5.1.2 LTE Network Configuration and User Mobility

We simulate a mobile network using the LTE module in the NS3 simulator. Our scenario contains one base station (eNodeB) with a cell radius of 10x10 km and 300 mobile users registered with the base station. We use a configurable ratio of the available resource blocks to live stream 360-degree videos to all 300 users. VRCast operates periodically each second to allocate resource blocks according to the channel conditions and viewports of mobile users. Users move according to the Self-similar Least Action Walk (SLAW) mobility model [39]. This model 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, [10], 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 the case where some users are walking and others are riding buses or cars. 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 bitrate they can receive.

In practical scenarios, mobile operators usually install base stations in crowded areas to serve most users with strong signals. Accordingly, in our simulations, users are distributed such that 2/3 of them are close to the base station (within 1/3 of the cell radius), while other users are scattered further away from the base station. Moreover, mobile users can move with different speeds depending on whether they are standing, walking, or riding a bus. We simulate users with moving speed of 1 m/sec (walking) and others with speed up to 10 m/s (riding a bus).

5.2 Algorithms Compared Against

We compare our algorithm against the closest, most recent, algorithm in the literature, which is MVR (Multicast of Virtual Reality content) [6]. MVR solves the problem of 360-degree mobile video multicast. It is a heuristic algorithm that divides users into subgroups based on their channel conditions and tile weights, and determines the bitrate for each tile in each subgroup. The authors of MVR showed that it outperforms the previous algorithms in the literature.

In addition, we compare against another state-of-the-art grouping algorithm, which is called Proportional Fairness (PF) [14]. 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.

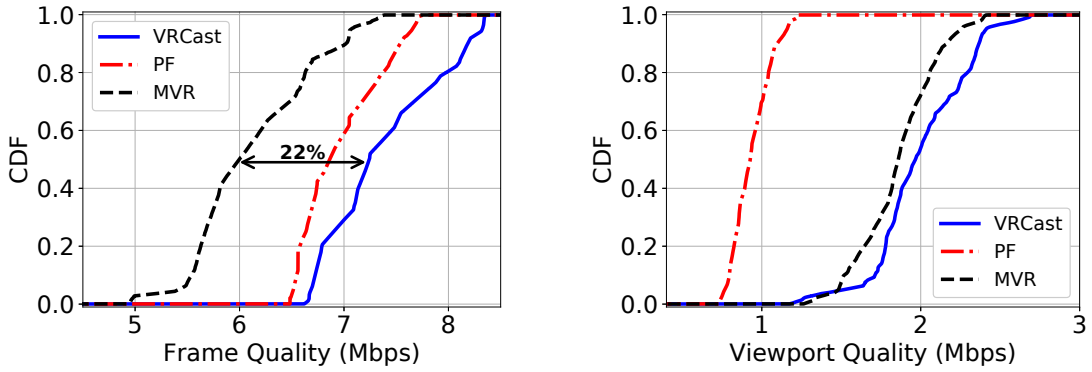
We consider the following performance metrics, which were used in similar previous works, e.g., [6, 8, 28, 56].

- **Frame Quality:** is the bitrate of the video frame sent to a multicast group containing all tiles.
- **Viewport Quality:** is the weighted sum of the bitrate assigned to tiles in the viewport. The weight of each tile is the percentage of its overlap with the viewport.
- **Spatial Variance:** is the variance in the quality of tiles in the viewport. A recent work [56] showed that the spatial variance has a direct effect on QoE.
- **Energy Saving:** is the average fraction of time that the reception component of a mobile device is closed to the total time. Users in the same group only wake up while the group's video frame is being sent and sleep otherwise according to the DRX parameters [5]. We note that the networking module consumes about 30% of the total energy of the mobile device during video streaming [32]. The energy saving that we measure, is only related to the networking component while other modules in the mobile device are not affected by our solution.
- **Fairness:** captures the relative quality observed by each user compared to others in the same multicast group. It is defined as the Jain's index of the ratio between each user viewport bitrate and the total frame bitrate.
- **Spectral Efficiency:** is the total transmitted data rate (in bits per second) divided by the channel bandwidth (in Hertz). This metric shows the efficiency of the streaming algorithm in using the cellular network resources, which is an important aspect for network operators.

In the following sections, we first present the comparisons of our algorithm against previous works. Then, we analyze the performance of our algorithm from different perspectives. In both cases, we repeat each experiment 30 times and plot and analyze the average results across all repetitions.

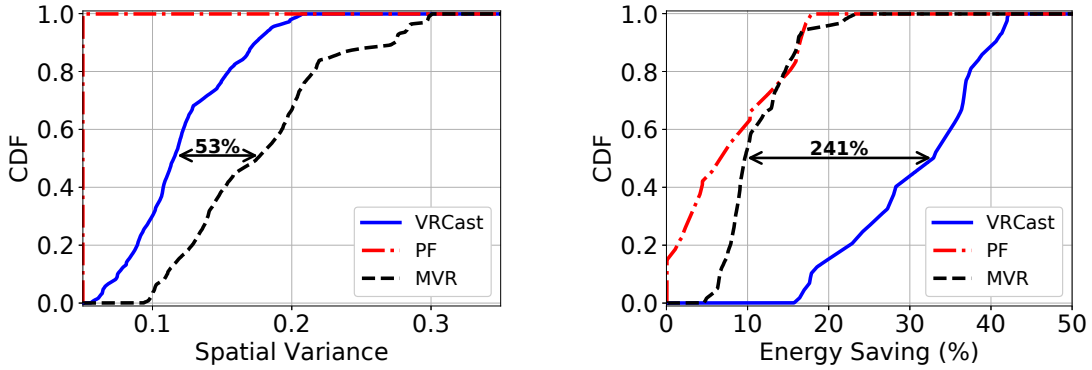
5.3 Performance of VRCast versus others

We compute and plot the cumulative distribution function (CDF) for each performance metric across time for all users and all video traces. Figures 5.1, 5.2, and 5.3 summarize the comparison results for all metrics. The figures show that our algorithm (VRCast) substantially outperforms the other algorithms (MVR and PF). Specifically, Figure 5.1(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 VRCast is about 7.25 Mbps while it is 5.97 Mbps for MVR, which is an improvement of 22%.



(a) Frame Quality (b) Viewport Quality

Figure 5.1: Performance of VRCast against other algorithms.

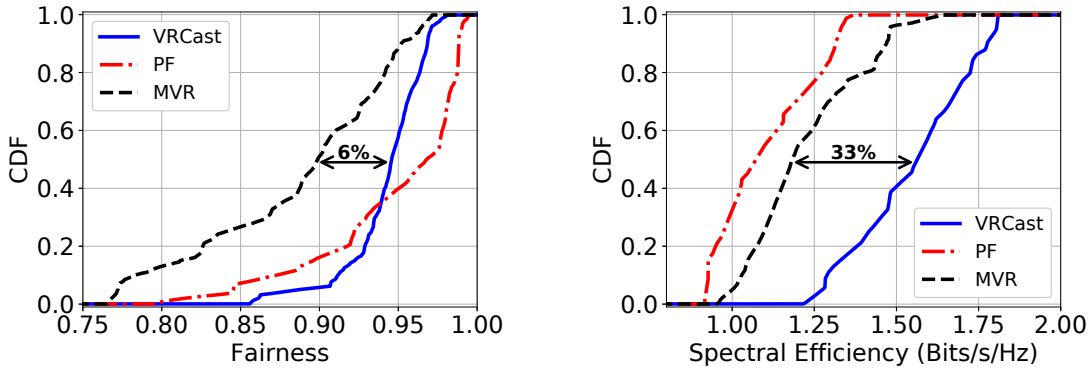


(a) Spatial Variance (b) Energy Saving

Figure 5.2: Performance of VRCast against other algorithms (continued 1).

In addition, as shown in Figures 5.1(b) and 5.2(a), 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. This, however, yields poor quality for the viewports (Figure 5.1(b)).

Mobile devices have limited battery lifetimes. Receiving and rendering 360-degree videos consume substantial energy. The proposed VRCast algorithm reduces the energy needed to receive 360-degree videos by carefully transmitting the video data in bursts. As shown in Figure 5.2(b), VRCast achieves substantial improvements in the 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 VR-



(a) Fairness

(b) Spectral Efficiency

Figure 5.3: Performance of VRCast against other algorithms (continued 2).

Cast, more than 80% of the users achieved at least 20% energy saving. The figure also shows that VRCast increases the median of the energy saving by more than 241% compared to MVR and PF.

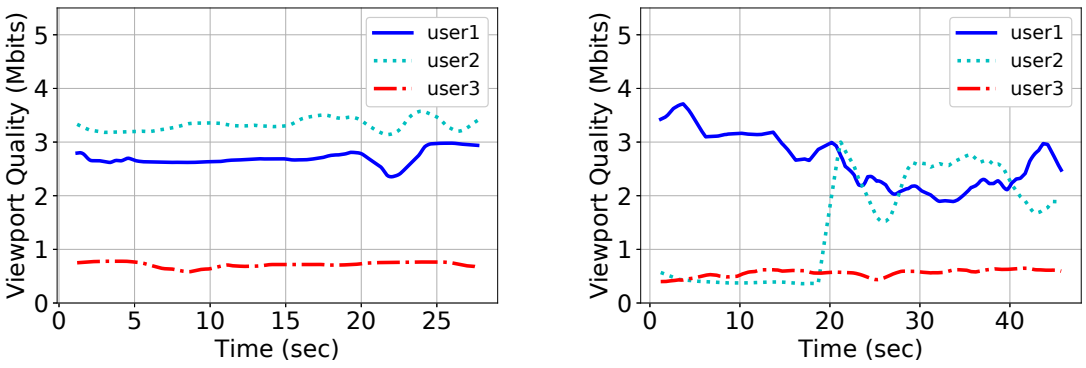
Maintaining fairness among different users is another desirable feature of the proposed VRCast. As illustrated in Figure 5.3(a), VRCast does not sacrifice the fairness among users to achieve the quality improvements shown in Figure 5.1. The fairness index of VRCast is fairly high and close to 1. For example, Figure 5.3(a) shows that 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.

Finally, 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. Figure 5.3(b) 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.

5.4 Analysis of VRCast

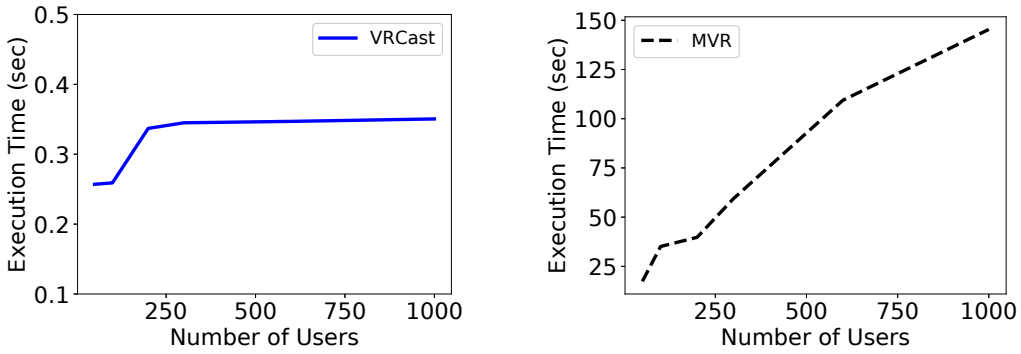
In this section, we take a close look at the performance of the proposed VRCast algorithm. First, We analyze the viewport quality of users across time. We choose 3 users with high (user 1), medium (user 2), and poor (user 3) channel conditions. We select two videos: boxing match and diving scene. We 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.

Figure 5.4(a) shows the viewport bitrate across time for the first 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



(a) Boxing Match (b) Diving

Figure 5.4: Sample results of VRCast with different videos.



(a) VRCast (b) MVR

Figure 5.5: Running time of VRCast and MVR as the number of users increases.

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 their viewport to less popular tiles (the audience) and receives less viewport bitrate.

Figure 5.4(b) shows the viewport bitrate across time for the second 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 low channel conditions 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.

Next, we analyze the scalability of VRCast by measuring and analyzing its average execution time with different number of users. The results in Figure 5.5 show that: (i) the running time of

VRCast does not depend on the number of users, and (ii) VRCast terminates in a few hundreds of milliseconds on a commodity PC. In contrast, MVR's running time grows linearly with the number of users and it is multiple orders of magnitudes greater than that of VRCast.

5.5 Testbed Implementation

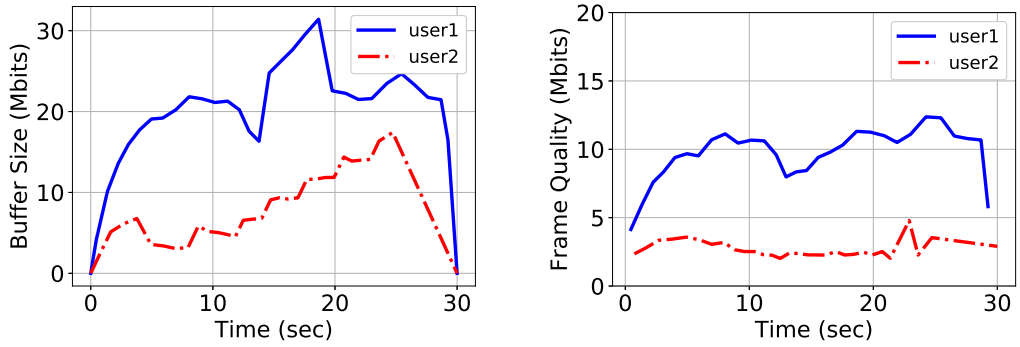
We developed an LTE testbed to validate our proposed solution in real systems. Building LTE testbeds for *multicast* services is quite challenging as there are many hardware and software components that need to be developed and integrated correctly to get the testbed running. We summarize our experience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

The testbed is similar to Figure 3.1 and it has base station, mobile devices, and BM-SC server. The hardware of the base station is implemented using the Ettus USRP B210 [21] software defined radio to transmit/receive data to/from mobile devices. The software of the base station is the Amarisoft LTE stack [9], which implements the functions of the eNB, 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 listens to a specific IP address.

We use Bittium mobile phones [13]. We chose these phones because they support the Expway eMBMS middleware [22], which allows the phones to discover multicast services and connect to them. The testbed has two phones in two different channel conditions. This is by placing one phone close to the base station and the other few meters away (the range of the base station is small as we did not use power amplifiers). The MCS mode for the phone close to the base station is 23 and for the far away phone is 12. Each phone represents a user. We make each phone run one of the user interactivity traces (Section 5.1).

In addition, we integrate simulated users in the testbed in order to test a realistic live streaming scenario in which multiple users, connected to the eNB, are streaming the video. Specifically, in addition to the two real users, we generate 298 simulated mobile users with different channel conditions and user interactivities. The information about all users (real and simulated) is periodically given to our algorithm (VRCast) to divide them into groups and decide on the quality for individual tiles in real time. 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.

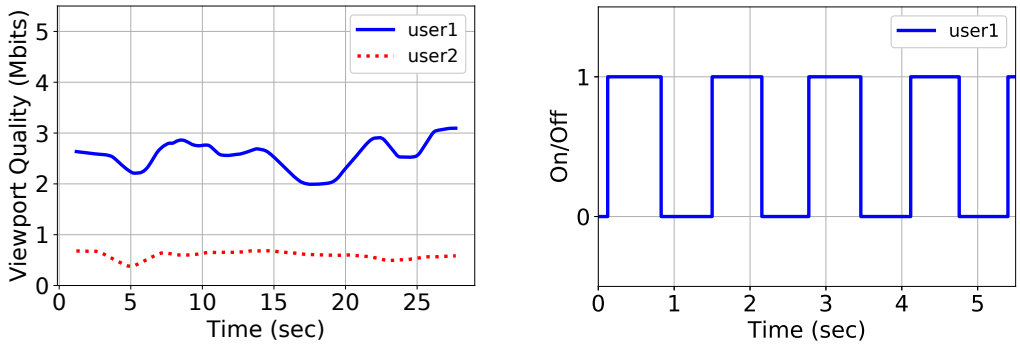
The last main part of the testbed is the BM-SC server, which runs VRCast. VRCast groups users, chooses quality representation for each tile, and concatenates tiles to generate a video file for each multicast group. It also computes the start and end times of data bursts to be transmitted to the mobile users. The decisions from VRCast are then mapped to various configuration parameters of the Amarisoft LTE software as, the number of allocated subframes for each service and minimum MCS for each PMCH.



(a) Buffer Size (Mbits)

(b) Frame Quality (Mbits)

Figure 5.6: Performance of VRCast in the LTE testbed.



(a) Viewport Quality (Mbits)

(b) Energy User 1

Figure 5.7: Performance of VRCast in the LTE testbed (continued).

5.6 Empirical Results from Testbed

The testbed is a proof-of-concept to show the practicality and correctness of VRCast. To this end, we analyze the buffer level at the two real phones as the streaming session progresses. The maximum buffer size was set at 50 Mbits. Figure 5.6(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 max value.

Next, we measure the actual frame quality and viewport quality received on the phones in Figures 5.6(b) and 5.7(a). User 1 has good channel conditions so s/he experiences a better frame and viewport qualities while user 2 has a poor channel condition but still gets decent qualities.

Finally, we analyze the burst transmission behavior of VRCast which provides energy saving for mobile receivers. At each mobile user, we record the start and end of the data reception (burst). We plot the results for user 1 in Figure 5.7(b) for 5-sec period. The figure clearly 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 Figures 5.6(b) and 5.7(a).

Chapter 6

Conclusions and Future Work

6.1 Conclusions

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 large 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 comprehensive solution for the problem of 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, achieves high spectral efficiency of the expensive wireless link, and runs in real time.

VRCast divides each 360-degree video into tiles where each tile is encoded in multiple DASH representations. It optimally divides mobile users into multiple multicast groups and assigns a quality representation to each tile to maximize the user-perceived video quality. Finally, VRCast transmits data in bursts to save energy for mobile receivers.

We evaluated VRCast using trace-based simulations. We used interactivity traces representing 1,335 sessions from 50 users watching 23 diverse 360-degree videos. We used a realistic mobility model to produce channel conditions for users. Our results showed that VRCast significantly outperforms the closest algorithms in the literature across multiple performance metrics. For example, compared to the state-of-the-art (MVR), the median frame quality is improved by up to 22% and the variance in the spatial quality is reduced by up to 53% and the energy saving for mobile devices is enhanced by up to 250%. Also, VRCast still achieves fairness among users and achieves a fairly high fairness index while utilizing LTE netowrk resources efficiently and providing high frame bitrates.

Furthermore, we developed a complete LTE testbed and evaluated VRCast in it. Our empirical results show that VRCast achieves smooth quality (no stalls or buffer overflows) and it effectively utilizes standard LTE features to transmit data in bursts to save energy of mobile devices.

6.2 Future Work

The work in this thesis can be extended in multiple directions. First of all, the tiling scheme can be further investigated to figure out what is the most suitable tiling for multicast streaming of a specific video. We can have a non-uniform tiling in which tile sizes change according to the regions of interest in the 360-degree video. Another direction is enhancing the prediction of users viewports to be more interactive to the changes in head movements during the transmission time and throughout the segment duration. Therefore, if the user suddenly moves his head to watch another part of the video, the quality of experience does not change significantly which further reduces temporal quality variance.

In addition, some outlier users can be watching less popular tiles and receiving very low quality. Scalable video coding (SVC) can be useful in this situation because of the idea of base layer and enhancement layers. We can have unicast sessions for enhancement layers to increase the quality of the viewport tiles of these users. Furthermore, we can find a more representative way of assigning weights to tiles. For example, sampling viewports of users in a more representative way instead of random sampling so that we have a better idea about the summary of the aggregate of the viewports of users. Also, the history of the viewports can be used to enhance tile weights. Finally, we can further investigate cases of mobile users handover, in which a UE is transferred from one base station to another without disconnecting the session. In this case, users might face a degradation in viewport quality. We can predict these cases and improve our adaptation algorithm to take care of these scenarios.

Bibliography

[1] 3GPP. Enhanced Television Services over 3GPP eMBMS, October 2017. http://www.3gpp.org/news-events/3gpp-news/1905-embms_r14.

[2] 3GPP. TS 26.346, Multimedia Broadcast/Multicast Service, Protocols and Codecs, 2017. http://www.3gpp.org/DynaReport/26346.htm.

[3] 3GPP. TS 36.104, E-UTRA, Base Station (BS) Radio Transmission and Reception, 2017. http://www.3gpp.org/DynaReport/36104.htm.

[4] 3GPP. TS 36.211, E-UTRA, Physical Channels and Modulation, 2017. www.3gpp.org/dynareport/36211.htm.

[5] 3GPP. TS 36.331, E-UTRA; Radio Resource Control (RRC) Protocol Specification, 2017. http://www.3gpp.org/DynaReport/36331.htm.

[6] Hamed Ahmadi, Omar Eltobgy, and Mohamed Hefeeda. Adaptive Multicast Streaming of Virtual Reality Content to Mobile Users. In *Proc. of ACM MM'17*, pages 601–605, Mountain View, CA, October 2017.

[7] Patrice Rondao Alface, Jean-François Macq, and Nico Verzijp. Interactive omnidirectional video delivery: A bandwidth-effective approach. *Bell Labs Technical Journal*, 16(4):135–147, 2012.

[8] Saleh Almowuena, Md Mahfuzur Rahman, ChengHsin Hsu, Ahmad AbdAllah Hassan, and Mohamed Hefeeda. Energy-Aware and Bandwidth-Efficient Hybrid Video Streaming Over Mobile Networks. *IEEE Trans. on Multimedia*, 18(1):102–115, 2016.

[9] Amarisoft. Amarisoft full LTE software suite solution, 2018. https://www.amarisoft.com/software-enb-epc-ue-simulator/.

[10] Nils Aschenbruck, Raphael Ernst, Elmar Gerhards-Padilla, and Matthias Schwamborn. Bonn-Motion: A Mobility Scenario Generation and Analysis Tool. In *Proc. of ACM SIMUTools'10*, page 51, Malaga, Spain, March 2010.

[11] Yanan Bao, Huasen Wu, Tianxiao Zhang, Albara Ah Ramli, and Xin Liu. Shooting a Moving Target: Motion-Prediction-Based Transmission for 360-Degree Videos. In *Proc. of IEEE Big Data'16*, pages 1161–1170, Washington, DC, December 2016.

[12] Ingo Bauermann, Matthias Mielke, and Eckehard Steinbach. H. 264 based coding of omnidirectional video. In *Computer Vision and Graphics*, pages 209–215. Springer, 2006.

[13] Bittium. Bittium Tough Mobile, 2018. https://www.bittium.com/ BittiumToughMobile.

[14] Jiasi Chen, Mung Chiang, Jeffrey Erman, Guangzhi Li, KK Ramakrishnan, and Rakesh K Sinha. Fair and Optimal Resource Allocation for LTE Multicast (eMBMS): Group Partitioning and Dynamics. In *Proc. of IEEE INFOCOM'15*, pages 1266–1274, Hong Kong, April 2015.

[15] Jiasi Chen, Rajesh Mahindra, Mohammad Amir Khojastepour, Sampath Rangarajan, and Mung Chiang. A Scheduling Framework for Adaptive Video Delivery over Cellular Networks. In *Proc. of ACM Mobicom'13*, pages 389–400, Miami, FL, October 2013.

[16] VNI Cisco. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016 - 2021 White Paper, March 2017.

[17] X. Corbillon, G. Simon, A. Devlic, and J. Chakareski. Viewport-Adaptive Navigable 360-degree Video Delivery. In *Proc. of IEEE ICC'17*, pages 1–7, May 2017.

[18] Xavier Corbillon, Francesca De Simone, and Gwendal Simon. 360-Degree Video Head Movement Dataset. In *Proc. of ACM MMSys'17*, pages 199–204, Taipei, Taiwan, June 2017.

[19] Ali El Essaili, Thorsten Lohmar, and Mohamed Ibrahim. Realization and evaluation of an end-to-end low latency live dash system. In *2018 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, pages 1–5. IEEE, 2018.

[20] Ericsson. Gigabit Network Launched in China, August 2017. https: //www.ericsson.com/en/press-releases/2017/8/gigabit-network-launched-in-china.

[21] Ettus. Ettus Research USRP B210, 2018. https://www.ettus.com/ product/details/UB210-KIT.

[22] Expway. Expway Broadcast Multicast Service Center (Expway BMSC), January 2017. http://www.expway.com/embms/.

[23] Facebook. Enhancing High-Resolution 360 Streaming With View Prediction, April 2017. https://code.facebook.com/posts/ 118926451990297/enhancing-high-resolution-360-streaming-with-view-prediction/.

[24] Chi-Wing Fu, Liang Wan, Tien-Tsin Wong, and Chi-Sing Leung. The rhombic dodecahedron map: An efficient scheme for encoding panoramic video. *IEEE Transactions on Multimedia*, 11(4):634–644, 2009.

[25] V. R. Gaddam, M. Riegler, R. Eg, C. Griwodz, and P. Halvorsen. Tiling in Interactive Panoramic Video: Approaches and Evaluation. *IEEE Trans. on Multimedia*, 18(9):1819–1831, Sept 2016.

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

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

[28] Mario Graf, Christian Timmerer, and Christopher Mueller. Towards Bandwidth Efficient Adaptive Streaming of Omnidirectional Video over HTTP: Design, Implementation, and Evaluation. In *Proc. of ACM MMSys'17*, pages 261–271, Taipei, Taiwan, June 2017.

[29] GSA. LTE Broadcast (eMBMS) Market Update, 2016 - 2021 White Paper, November 2017.

[30] Te-Yuan Huang, Ramesh Johari, Nick McKeown, Matthew Trunnell, and Mark Watson. A buffer-based approach to rate adaptation: Evidence from a large video streaming service. *ACM SIGCOMM Computer Communication Review*, 44(4):187–198, 2015.

[31] Junchen Jiang, Vyas Sekar, and Hui Zhang. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. *IEEE/ACM Transactions on Networking (TON)*, 22(1):326–340, 2014.

[32] Nan Jiang, Viswanathan Swaminathan, and Sheng Wei. Power Evaluation of 360 VR Video Streaming on Head Mounted Display Devices. In *Proc. of ACM NOSSDAV'17*, pages 55–60, Taipei, Taiwan, June 2017.

[33] Hans Kellerer, Ulrich Pferschy, and David Pisinger. Introduction to NP-Completeness of Knapsack Problems. In *Knapsack problems*, pages 483–493. Springer, 2004.

[34] Evgeny Kuzyakov. End-to-end optimizations for dynamic streaming. *Facebook Code, Feb*, 22, 2017.

[35] Evgeny Kuzyakov, Shannon Chen, and Renbin Peng. Enhancing high-resolution 360 streaming with view prediction, 2017.

[36] Evgeny Kuzyakov and David Pio. Next-generation video encoding techniques for 360 video and vr, 2016.

[37] Jean Le Feuvre and Cyril Concolato. Tiled-Based Adaptive Streaming using MPEG-DASH. In *Proc. of ACM MMSys'16*, pages 41–43, Klagenfurt, Austria, May 2016.

[38] David Lecompte and Frédéric Gabin. Evolved Multimedia Broadcast/Multicast Service (eMBMS) in LTE-Advanced: Overview and Rel-11 Enhancements. *IEEE Communications Magazine*, 50(11), 2012.

[39] Kyunghan Lee, Seongik Hong, Seong Joon Kim, Injong Rhee, and Song Chong. SLAW: A New Mobility Model for Human Walks. In *Proc. of IEEE INFOCOM'09*, pages 855–863, Rio de Janeiro, Brazil, April 2009.

[40] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. Neural adaptive video streaming with pensieve. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 197–210. ACM, 2017.

[41] Kiran Misra, Andrew Segall, Michael Horowitz, Shilin Xu, Arild Fuldseth, and Minhua Zhou. An Overview of Tiles in HEVC. *IEEE Journal of Selected Topics in Signal Processing*, 7(6):969–977, 2013.

[42] I Mpeg. Information technology-dynamic adaptive streaming over http (dash)-part 1: Media presentation description and segment formats. *ISO/IEC MPEG, Tech. Rep*, 2012.

[43] Afshin Taghavi Nasrabadi, Anahita Mahzari, Joseph D. Beshay, and Ravi Prakash. Adaptive 360-Degree Video Streaming using Scalable Video Coding. In *Proc. of ACM MM'17*, pages 1689–1697, Mountain View, CA, October 2017.

[44] King-To Ng, Shing-Chow Chan, and Heung-Yeung Shum. Data compression and transmission aspects of panoramic videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(1):82–95, 2005.

[45] Omar A Niamut, Emmanuel Thomas, Lucia D'Acunto, Cyril Concolato, Franck Denoual, and Seong Yong Lim. MPEG DASH SRD: Spatial Relationship Description. In *Proc. of ACM MMSys'16*, pages 1–8, Klagenfurt, Austria, May 2016.

[46] Toni Paila, Rod Walsh, Michael Luby, Vincent Roca, and Rami Lehtonen. FLUTE-file Delivery Over Unidirectional Transport. Technical report, 2012.

[47] Stefano Petrangeli, Viswanathan Swaminathan, Mohammad Hosseini, and Filip De Turck. An HTTP/2-Based Adaptive Streaming Framework for 360 Virtual Reality Videos. In *Proc. of ACM MM'17*, pages 306–314, Mountain View, CA, October 2017.

[48] Feng Qian, Lusheng Ji, Bo Han, and Vijay Gopalakrishnan. Optimizing 360 video delivery over cellular networks. In *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, pages 1–6. ACM, 2016.

[49] Yago Sánchez, Robert Skupin, and Thomas Schierl. Compressed Domain Video Processing for Tile Based Panoramic Streaming Using HEVC. In *Proc. of IEEE ICIP'15*, pages 2244–2248, Quebec City, QC, September 2015.

[50] Thomas Stockhammer. Dynamic adaptive streaming over http–: standards and design principles. In *Proceedings of the second annual ACM conference on Multimedia systems*, pages 133–144. ACM, 2011.

[51] Thomas Stockhammer and Michael G Luby. Dash in Mobile Networks and Services. In *Proc. of VCIP'12*, pages 1–6, San Diego, CA, November 2012.

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

[53] Marko Viitanen, Ari Koivula, Ari Lemmetti, Jarno Vanne, and Timo D Hämäläinen. Kvazaar HEVC Encoder for Efficient Intra-Coding. In *Proc. of IEEE ISCAS'15*, pages 1662–1665, Lisbon, Portugal, May 2015.

[54] Hui Wang, Vu-Thanh Nguyen, Wei Tsang Ooi, and Mun Choon Chan. Mixing Tile Resolutions in Tiled Video: A Perceptual Quality Assessment. In *Proc. of ACM NOSSDAV'14*, page 25, Singapore, Singapore, March 2014.

[55] Mengbai Xiao, Chao Zhou, Yao Liu, and Songqing Chen. OpTile: Toward Optimal Tiling in 360-degree Video Streaming. In *Proc. of ACM MM'17*, pages 708–716, Mountain View, CA, October 2017.

[56] Lan Xie, Zhimin Xu, Yixuan Ban, Xinggong Zhang, and Zongming Guo. 360ProbDASH: Improving QoE of 360 Video Streaming Using Tile-based HTTP Adaptive Streaming. In *Proc. of ACM MM'17*, pages 315–323, Mountain View, CA, October 2017.

[57] Xiufeng Xie, Xinyu Zhang, Swarun Kumar, and Li Erran Li. pistream: Physical layer informed adaptive video streaming over lte. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pages 413–425. ACM, 2015.

[58] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. A control-theoretic approach for dynamic adaptive video streaming over http. In *ACM SIGCOMM Computer Communication Review*, volume 45, pages 325–338. ACM, 2015.

[59] Matt Yu, Haricharan Lakshman, and Bernd Girod. A Framework to Evaluate Omnidirectional Video Coding Schemes. In *Proc. of IEEE ISMAR'15*, pages 31–36, Fukuoka, Japan, October 2015.

[60] Ya-Ju Yu, Pi-Cheng Hsiu, and Ai-Chun Pang. Energy-Efficient Video Multicast in 4G Wireless Systems. *IEEE Trans. on Mobile Computing*, 11(10):1508–1522, 2012.

[61] Alireza Zare, Alireza Aminlou, Miska M Hannuksela, and Moncef Gabbouj. HEVC-Compliant Tile-Based Streaming of Panoramic Video for Virtual Reality Applications. In *Proc. of ACM MM'16*, pages 170–178, Amsterdam, Netherlands, October 2016.

[62] Chao Zhou, Zhenhua Li, and Yao Liu. A Measurement Study of Oculus 360 Degree Video Streaming. In *Proc. of ACM MMSys'17*, pages 27–37, Taipei, Taiwan, 2017.