Dissecting the Performance of VR Video Streaming through the VR-EXP Experimentation Platform

ROBERTO IRAJA TAVARES DA COSTA FILHO, Institute of Informatics - UFRGS, Brazil
MARCELO CAGGIANI LUIZELLI, Federal University of Pampa, Brazil
STEFANO PETRANGELI, Adobe Research
MARIA TORRES VEGA, JEROEN VAN DER HOOFT, TIM WAUTERS, and FILIP DE TURCK, Ghent University - imec, Belgium
LUCIANO PASCHOAL GASPARY, Institute of Informatics - UFRGS, Brazil

To cope with the massive bandwidth demands of Virtual Reality (VR) video streaming, both the scientific community and the industry have been proposing optimization techniques such as viewport-aware streaming and tile-based adaptive bitrate heuristics. As most of the VR video traffic is expected to be delivered through mobile networks, a major problem arises: both the network performance and VR video optimization techniques have the potential to influence the video playout performance and the Quality of Experience (QoE). However, the interplay between them is neither trivial nor has it been properly investigated. To bridge this gap, in this article, we introduce VR-EXP, an open-source platform for carrying out VR video streaming performance evaluation. Furthermore, we consolidate a set of relevant VR video streaming techniques and evaluate them under variable network conditions, contributing to an in-depth understanding of what to expect when different combinations are employed. To the best of our knowledge, this is the first work to propose a systematic approach, accompanied by a software toolkit, which allows one to compare different optimization techniques under the same circumstances. Extensive evaluations carried out using realistic datasets demonstrate that VR-EXP is instrumental in providing valuable insights regarding the interplay between network performance and VR video streaming optimization techniques.

CCS Concepts: • **Information systems** → **Multimedia streaming**; • **Human-centered computing** → **Virtual reality**; • **Networks** → *Network protocols*; *Public Internet*;

Additional Key Words and Phrases: Virtual reality, adaptive streaming, quality of experience, quality of service

This research was performed partially within the project G025615N "Optimized source coding for multiple terminals in self-organizing networks" from the fund for Scientific Research-Flanders (FWO-V). Maria Torres Vega is funded by a grant of the Research Foundation - Flanders (FWO). This work was also partially funded by CAPES, CNPq, FAPERGS, and IFSul.
Authors' addresses: R. I. T. D. C. Filho, Av. Bento Gonçalves, 9500, Campus do Vale - Bloco IV, Caixa Postal 15064, 91501-970 - Porto Alegre - Brazil; email: roberto.costa@inf.ufrgs.br; M. C. Luizelli, Av. Tiaraju, 810, 97546-550, Alegrete - RS; email: marceloluizelli@unipampa.edu.br; S. Petrangeli, 345 Park Ave, San Jose, CA 95110, United States; email: petrange@adobe.com; M. T. Vega and J. V. D. Hooft, Department of Information Technology (EA05), Ghent University – imec, Technologiepark-Zwijnaarde 15, B-9052 Gent, Belgium; emails: {maria.torresvega, jeroen.vanderhooft}@ugent.be; T. Wauters and F. D. Turck, Department of Information Technology (EA05), Ghent University – imec, Technologiepark-Zwijnaarde 15, B-9052 Gent, Belgium; emails: {tim.wauters, filip.deturck}@ugent.be; L. P. Gaspary, Av. Bento Gonçalves, 9500, Campus do Vale - Bloco IV, Caixa Postal 15064, 91501-970 - Porto Alegre- Brazil; email: paschoal@inf.ufrgs.br.

1551-6857/2019/12-ART111 $15.00
https://doi.org/10.1145/3360286

ACM Reference format:
Roberto Iraja Tavares da Costa Filho, Marcelo Caggiani Luizelli, Stefano Petrangeli, Maria Torres Vega, Jeroen van der Hooft, Tim Wauters, Filip De Turck, and Luciano Paschoal Gaspary. 2019. Dissecting the Performance of VR Video Streaming through the VR-EXP Experimentation Platform. *ACM Trans. Multimedia Comput. Commun. Appl.* 15, 4, Article 111 (December 2019), 23 pages.
https://doi.org/10.1145/3360286

1 INTRODUCTION

Virtual Reality (VR) video streaming applications are becoming increasingly popular. VR Head-Mounted Displays (HMDs) are expected to grow from 18 million in 2017 to nearly 100 million by 2022, while the associated network traffic is expected to increase 12-fold [5]. The same study points out VR video streaming as a key application, which has the potential to significantly increase the VR penetration. VR video streaming applications are challenging due to three main reasons: (*i*) they are expected to run largely over mobile networks, as mobile devices will account for 71% of the total IP Internet traffic by 2022 [5]; (*ii*) mobile networks are characterized by highly variable levels of performance [7]; and (*iii*) VR video streaming applications demand high levels of network performance to achieve a satisfactory Quality of Experience (QoE) [5]. To provide a notion of how demanding these applications are, recent studies have shown that, to provide adequate levels of QoE, current VR video applications require a network delay lower than 9 ms [8], while the bandwidth needs for the upcoming ultra high definition VR will reach 500 Mbps [5]. At this level of demand, not only will network operators struggle to provide cost-effective services, but VR video content providers and developers will also be challenged by such resource-intensive applications.

To overcome the aforementioned challenges, both the academy and the industry are investigating novel approaches for improving the efficiency of the VR video streaming ecosystem. In this direction, efficient spherical-to-plane projection schemes, which include tile-based VR video streaming, are prominent strategies for reducing the bandwidth requirements imposed by VR videos [4, 6, 12, 14, 16, 37]. These investigations extend well-established approaches to 2D video streaming, such as HTTP Adaptive Streaming (HAS) and Dynamic Adaptive Streaming over HTTP MPEG-DASH paradigms [9, 27]. In the first step, VR videos are encoded at different quality levels and representations (e.g., 720 p, 1,080 p, 4 K, 8 K). Subsequently, they are split into both spatial tiles and temporal segments. During the streaming session, the client will only request tiles corresponding to its viewport (i.e., the visible portion of the full 360-degree panoramic view). To perform selective tile requests, these schemes rely on viewport prediction heuristics [11, 15, 17, 22, 25, 26]. Other crucial building blocks in this ecosystem are Adaptive Bitrate (ABR) streaming and Buffer Management heuristics. ABR streaming benefits from both the temporal/spatial segmentation and predicted viewport to manage the playout buffer. To do so, it requests for each segment the tiles that are estimated to belong to the viewport in high quality, while the remaining tiles will either be requested at lower quality variants or not fetched at all [2, 12, 13, 19, 25].

Optimization schemes, such as the ones just mentioned, contribute to minimizing the use of network resources (mainly in terms of bandwidth). For example, when prioritizing high-resolution representations only for tiles in the viewport, a bandwidth reduction of up to 72% can be achieved [14]. However, optimization approaches can impair the performance of the VR video streaming severely, thus, degrading the user's perception of the service (i.e., QoE). For example, consider the case where the predicted viewport tiles are downloaded in advance. Errors in the viewport prediction for the user's field of view will lead to QoE degradation even though the bandwidth is high enough for the application requirements. Furthermore, the video encoding and streaming decisions (such as the spherical-to-plane projection strategy, tiling scheme, available quality representations, frame rate) and the client-side implementation aspects (e.g., playout

buffer size, rate adaptation heuristics, and tile fetching method) play an essential role in shaping the resulting VR video playout performance and, ultimately, QoE. Finally, it is worth noting that these parameters and heuristics may perform quite differently when subjected to variable network performance conditions. We deem that distinct groups can benefit from a solution to this problem: (*i*) both the research community and VR solutions designers can carry out far-reaching evaluation of their approaches when subjected to complex and realistic scenarios; and (*ii*) considering that network operators already have tools in place for measuring network performance, they can estimate VR video application performance and QoE experienced by their subscribers.

When considering both the multitude of approaches to optimize VR video streaming and the highly variable mobile network performance, it becomes a difficult challenge to understand how different (combinations of) optimization techniques perform under varying infrastructure conditions. The lack of a publicly available method and tools for systematic and reproducible evaluation exacerbate this challenge. To fill in this gap, in this article, we propose VR-EXP, an adaptive VR video streaming experimentation platform. The platform is capable of systematically evaluating different combinations of VR video streaming optimization approaches. Also, VR-EXP allows pinpointing the interplay between a set of optimization techniques and variable network performance. Composed of an evaluation method and software components, VR-EXP assumes as input tile-based VR videos, network datasets, and parameters (e.g., network performance conditions, users' head-tracking information, ABR heuristics, and tile fetching methods). Then, it emulates essential components of the VR video streaming ecosystem, measuring key VR video playout performance indicators. Finally, our platform produces, as output, detailed VR video playout performance and QoE estimation reports. Using VR-EXP, we carry out an in-depth analysis of (combinations of) state-of-the-art VR video optimization approaches under varying network conditions. It is worth mentioning that although we considered measurements from a mobile network as input, the platform is expected to be flexible enough to work with fixed network traces. However, such investigation is left as a suggestion for future work.

We summarize the contributions of this work as follows:

- We provide a platform to carry out systematic evaluations that can be executed across realistic scenarios.
- Throughout an extensive evaluation, we provide an in-depth analysis of the performance of cutting-edge optimization approaches for VR video streaming.

The remainder of this article is organized as follows: In Section 2, we present an overview of background concepts and state-of-the-art optimization approaches for the VR video ecosystem. In Section 3, we introduce VR-EXP, encompassing its main components and design choices. In Section 4, we outline the evaluation setup including the considered parameters and datasets. Then, in Section 5, we present and discuss the main results. Our conclusions along with perspectives for future work are presented in Section 6.

2 BACKGROUND AND STATE-OF-THE-ART

In this section, we provide a thorough description of state-of-the-art optimization techniques for VR video streaming. We organize these investigations into three research groups. We start by reviewing relevant projection schemes for VR video encoding. Next, we evaluate prominent investigations regarding viewport prediction. Finally, we evaluate adaptive bitrate streaming and buffer management approaches for VR videos.

2.1 Spherical-to-plane Projection

One effective strategy to reduce the huge bandwidth demands of 360-videos is delivering only the viewport in high resolution, streaming the remaining area of the video in low resolution or

not at all. To achieve this spatial segmentation of the panoramic view, several approaches explore spherical-to-plane projection techniques [4, 6, 12, 14, 16, 37]. For example, Graf et al. [12] examine the bitrate overhead and bandwidth requirements of distinct tiling schemes (i.e., 1×1, 3×2, 5×3, 6×4, and 8×5) implemented using modern video codecs (e.g., HEVC/H.265 and VP9). By applying Peak Signal-to-Noise Ratio (PSNR) within the VR video viewport, the authors assess the video quality and conclude that the 6×4 tiling scheme provides the best trade-off among viewport selection flexibility, bitrate overhead, and bandwidth requirements. In a similar direction, Zhou et al. [37] further examine this field by comparing standard spherical projection approaches to offset projection techniques. The latter are characterized by distorting the spherical surface to allow the convergence of the pixels of the VR video in a particular direction. Offset projections are significantly more complex than traditional projection techniques, because they demand a simultaneous control of bitrate and view orientation adaptations. By employing PSNR and Structural Similarity (SSIM), the authors conclude that, in general, offset projections can provide better quality than their non-offset counterparts. Despite their contributions, the conclusions of these investigations are limited, because they do not consider important variables, such as the effects of variable viewport prediction error and parallel fetching methods (such as HTTP/2) on their approaches. Also, the mentioned approaches are evaluated considering limited network performance conditions.

In another important investigation, Chen et al. [4] analyze recent advancements regarding alternative projection methods, including viewport-dependent and viewport-independent approaches. The central objective of this work is to assess both the coding efficiency and distortion introduced by each approach. Besides valuable quantitative and qualitative insights regarding a wide range of projection schemes, the authors conclude that to effectively evaluate such a wide range of projection schemes, a more sophisticated evaluation process is required. The main reason for this conclusion is that traditional PSNR computes the whole projection map, which cannot handle viewport-dependent projections. Additionally, due to the unpredictability of viewport prediction errors, the areas surrounding the viewport should also be considered in the quality evaluation, but with a reduced weight. In this investigation, the authors also review alternative metrics for video quality assessment proposed by JVET [3]. They conclude that although several flaws of conventional PSNR have been fixed, a more comprehensive method for evaluating video quality for viewport-dependent VR videos is still missing.

2.2 Viewport Prediction Algorithms

Viewport prediction heuristics benefit from the tile-based structures of the VR video to enable differentiated handling of group of tiles. Since a full VR video can easily reach 12 K video resolution [6], most video players rely on heuristic algorithms to predict near-future user's head movements. Considering the next position prediction, the VR video emulator is able to keep a small playout buffer (e.g., 2 seconds) requesting only tiles that are likely to belong to the viewport, which ultimately leads to reduced bandwidth utilization. In this direction, several recent investigations propose viewport prediction algorithms [11, 15, 17, 22, 25, 26].

To illustrate how the viewport prediction works, consider the example of a user watching a tile-based VR video using a head-mounted display. Assume a given temporal segment S_k and a respective viewport V_k, as depicted in Figure 1(a). At this moment, the video player is requesting high-resolution chunks only for tiles inside the viewport V_k. Then, based on the viewport prediction for the next segment (S_{k+1}), the video player starts requesting high-resolution tiles for the predicted viewport V_{k+1} (delimited by the blue dashed square in Figure 1(b)). However, rather than moving his/her head up, consider that the viewer actually slightly moves to the right (see Figure 1(c)). At this point, due to the viewport predictor error, the VR player requested seven tiles in high-resolution that will not actually be displayed (upper left red tiles in Figure 1(d)). Likewise, seven low-resolution tiles end up belonging to the viewport (bottom right red tiles in Figure 1(d)).

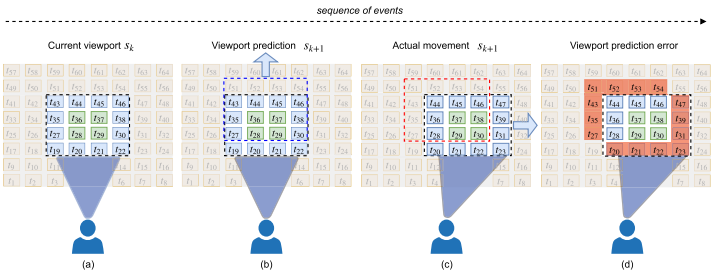


Fig. 1. Working principles of the viewport prediction and the viewport error.

As one can observe, viewport prediction is a sensitive task. Viewport prediction errors may lead to partial or full degradation of the perceived quality, even if the network performance conditions are enough to guarantee the user's QoE.

To perform the viewport prediction, most approaches follow a similar procedure, which includes processing one or more input information, applying a prediction method, and then checking the prediction accuracy. As input, prediction algorithms can rely on past users' head motion [15, 24, 26], fixation point acceleration [22], fixation point angular velocity [11, 22, 25], image saliency maps and motion maps [11], or even sound localization information [17]. In turn, to perform the viewport prediction itself, state-of-the-art approaches rely on deep learning [15], mathematical modeling [17, 22, 24–26], or neural networks [11]. Finally, the prediction accuracy is assessed by subjecting the prediction model to traces containing realistic head-tracking information (i.e., ground truth). Thus, the residual error can be evaluated. By performing such predictions, the VR video player can, according to He et al. [14], reduce bandwidth utilization by up to 72%.

As discussed, although prediction algorithms may present acceptable accuracy under certain circumstances, prediction errors are very likely to occur due to the randomness of users' behavior. Besides, prediction algorithms may considerably decrease their accuracy when the size of the playout buffer is increased. For example, the prediction accuracy can drop from 90% to approximately 60% if the prediction window is increased from 1 to 2 seconds [26]. However, an increased playout buffer may be crucial to operate in current mobile networks, which are characterized by highly variable performance conditions, even in short time frames. Considering these intricacies, an effective assessment of viewport prediction algorithms should consider (and quantify) how the error rate of a particular algorithm affects QoE when combined with other optimizations (e.g., buffer management heuristics and dynamic rate adaptation algorithm) and subjected to realistic network performance.

2.3 Adaptive Bitrate Algorithms and Buffer Management

Taking viewport prediction information as input, most approaches rely on per-tile rate adaptation algorithms. This method allows reducing the amount of information to be downloaded by keeping only the viewport's tiles in high resolution. However, this task is far from trivial, even for traditional video streaming. ABR algorithms are complex, because they must manage the available bandwidth while maximizing the quality representation and minimizing the stall probability. Although ABR algorithms for traditional 2D video streaming have been extensively explored, recent investigations [1, 28] show that it is still an open research problem. As an example, the possibility is demonstrated to significantly improve the performance of state-of-the-art ABR algorithms, namely BOLA [29] and MPC [35]. Akhtar et al. [1] demonstrate that both BOLA and MPC algorithms rely on parameters that are sensitive to variable network performance, so they may perform poorly under certain conditions. To fill this gap, the authors introduce VirtualPlayer [1], a

trace-based simulator that mimics the behavior of a traditional video streaming player. It allows, for example, to investigate ABR algorithms when subjected to real-world networks. In the same direction, Spiteri et al. [28] introduce Sabre, an open-source simulation tool that enables simulating ABR algorithms for 2D videos when subjected to realistic requirements.

When it comes to VR video, ABR becomes a much more challenging task. State-of-the-art approaches for adaptive bitrate in VR videos differ from each other mainly with respect to how they manage the balance between video quality and available bandwidth while considering the spatial segmentation. For example, Petrangeli et al. [25] consider a multi-zone VR video and propose a per-tile quality selection heuristic. The algorithm starts by selecting the highest available quality for the inner tiles (close to the fixation point), and then repeats this procedure for the outer zones until the residual bandwidth is exhausted. This approach alleviates the edge effect (transition between different quality representations). Thus, it provides superior VR video quality at the cost of increased bandwidth consumption.

He et al. [13] propose to simultaneously optimize, among other parameters, playout bitrate and buffer occupancy. Similarly to Petrangeli et al. [25], they perform bitrate adaptations depending on the position of each tile concerning the current fixation point. However, they introduce a learning strategy with the ability to avoid performance degradation for future segments by automatically adapting the buffer reservation. By using a fine-grained bitrate adaptation, these investigations were able to reduce the bandwidth utilization in 35% and 40%, respectively.

Graf et al. [12] advance a step forward in the state-of-the-art by providing a comprehensive investigation with respect to essential components of the VR ecosystem. The authors introduce three tile scheme strategies for ABR, namely Full Delivery Basic, Full Delivery Advanced, and Partial Delivery. These schemes drive the ABR algorithm regarding the bitrate adaptation for both the viewport and the remaining tiles. For example, in Full Delivery Basic scheme, all the tiles belonging to the viewport are requested in the highest available quality, while the remaining tiles are requested in the lowest quality, regardless of the available bandwidth. The Partial Delivery scheme employs an aggressive bandwidth saving strategy, requesting only the tiles within the viewport in high resolution, while the remaining tiles are not requested at all. The authors evaluate several projection schemes (as discussed in Section 2.1), combined with multiple segment sizes. By assessing the bitrate overhead, bandwidth requirements, and viewport quality, this approach achieves bandwidth savings from 40% to up to 65% when compared to state-of-the-art techniques.

Closely related to ABR algorithms, the playout buffer management plays a vital role in the VR video realm. As discussed earlier, an increased playout buffer size is an effective way to protect against stalls (i.e., empty buffer) caused by network performance fluctuations. However, a small playout buffer is necessary to keep the accuracy of viewport prediction methods within acceptable levels. Specifically on this subject, Ma et al. [19] propose a dynamic buffer size management method that is guided by a constrained optimization model. This method aims at maximizing QoE by adjusting the buffer size based on the viewport prediction error and available bandwidth. Throughout simulation experiments, the authors claim gains from 2.7% up to 6.7%, in terms of QoE, when compared to non-dynamic buffer size approaches. In another relevant investigation, Almquist et al. [2] present a data-driven study that explores the trade-off between the playout buffer size (i.e., prefetching aggressiveness) and viewport prediction errors. The prefetching aggressiveness is evaluated while considering different VR video categories (i.e., exploration, static, moving, rides, and misc.). The authors provide valuable qualitative and quantitative insights regarding how to best address the prefetching aggressiveness trade-off. As a key insight, they demonstrate that the accuracy of the prediction varies significantly among different categories. Additionally, in line with previous investigations, they emphasize that adequate levels of viewport prediction accuracy are observed only within a very small time frame.

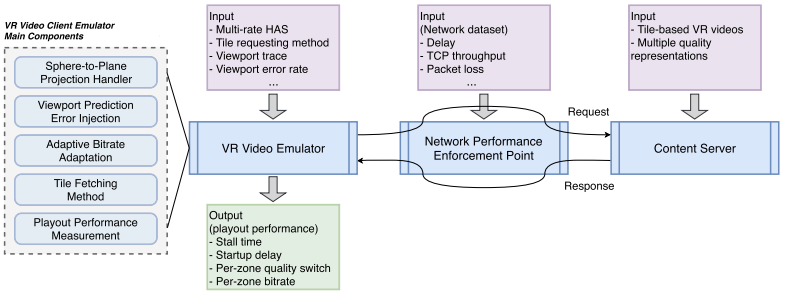


Fig. 2. VR-EXP general scheme.

In summary, during a streaming session, several components (e.g., projection scheme, viewport prediction error, buffer management approach, dynamic rate adaptation algorithm, and network performance) will have a major influence on the user's QoE. Despite several research efforts, little is known about the interplay between a set of VR video components and variable network performance conditions. Also, a solution to provide an in-depth and reproducible evaluation of the VR video streaming ecosystem is still missing. In this article, we introduce VR-EXP, an open-source publicly available platform for evaluating adaptive VR video streaming performance and QoE when subjected both to multiple VR video optimization techniques and variable network performance conditions. Different from the related work, instead of evaluating each optimization technique independently (or within a reduced set), our approach provides an extensible VR video emulator that allows for the simultaneous evaluation of multiple state-of-the-art optimization techniques. Combined with the provided network controller and realistic network performance dataset, VR-EXP contributes a step forward in the VR field by providing a reproducible method for evaluating adaptive VR video streaming optimization approaches. To the best of our knowledge, this is the first open-source method and toolkit for a comprehensive evaluation of the VR video ecosystem.

3 METHODOLOGY

In this section, we introduce the VR-EXP platform. We start by presenting the general scheme, highlighting its inputs, outputs, and main modules. Then, we introduce the VR video client emulator and its main components. Next, we discuss alternatives for enforcing network performance conditions. Finally, we present the considered QoE model, which allows evaluating the effects of multiple VR video optimization techniques on QoE.

3.1 VR-EXP General Scheme

In a nutshell, the VR-EXP platform enables evaluating the interplay between a set of adaptive tile-based VR streaming optimizations and variable network performance conditions. Figure 2 depicts the main modules of VR-EXP. The proposed method consists of systematically fetching VR videos through a controlled network environment. From a client perspective, the adaptive VR video client emulator coordinates the use of several VR video techniques upon requesting VR videos from a content server. During the streaming session, the Network Performance Enforcement Point enforces realistic network conditions on the network links between the VR Video Client Emulator and the Content Server. Once the VR video streaming session is finished, VR-EXP reports key VR video playout performance metrics.

A central component of VR-EXP is the VR video client emulator, which is responsible for processing input parameters, emulating state-of-the-art VR video optimization approaches, and measuring the playout performance. One possible path to implement this component would be to

adapt an existing VR video player. However, we decided to build a VR video client emulator from scratch. The reason for this design choice is twofold. First, we wanted configurable parameters for VR experimentation, including temporization aspects, to maximize the accuracy of the performance measurements. For example, VR-EXP allows the performance measuring interval to be configured proportionally to the segment size. Second, we wanted to stay focused on the interplay between the network performance and optimization heuristics, without the interference of external factors such as the rendering process. The rendering task is an important component of the VR video ecosystem. It can be divided into two main entities, namely the rendering process and the viewport tile scheme. The rendering process is primarily related to the HMD capabilities of each device, while the tile scheme interacts with both the rendering process and the network performance. In this work, we are interested in evaluating the influence of variable network performance on VR adaptive streaming in an isolated manner, without the interference of HMD particularities. Although VR-EXP does not provide rendering features, it allows for a configurable tile scheme, which enables capturing the influence of the tile scheme on QoE.

The source code is written in the C language using the Curl[1] and POSIX pthreads libraries to systematically fetch tile-based VR videos over the HTTP protocol. The complete list of libraries and packages used by VR-EXP can be found at the VR-EXP repository. Also, the reader interested in a jump-start to VR video experimentation may refer to VR-EXP quick start guide,[2] which provides a step-by-step procedure to carry out experiments, as well as a fully configured and ready-to-use virtual machine. To emulate a dynamic network topology as well as enforce real-world conditions, VR-EXP relies on either an SDN network controller or the Linux Traffic Controller. In the proposed platform, adaptive VR videos are provided by an HTTP server (Apache[3]) that delivers tile-based VR videos in multiple quality representations according to the HAS scheme.

The emulation of the entire VR video streaming ecosystem requires the configuration of several parameters and inputs. For flexibility, VR-EXP enables the definition of its parameters at run time. It allows building scripts for automating complex and extensive experiments. For example, it is possible to parameterize the VR video client emulator by defining behavior characteristics such as the tile requesting method, the rate adaptation heuristic, the expected viewport prediction error, and so forth. In turn, the network module is expected to be fed with a dataset containing a set of network performance metrics (e.g., delay, packet loss rate, TCP throughput). It then enforces these conditions into the emulated links connecting the VR video client emulator to the content server. Once all the input datasets and parameters are configured, the VR video client emulator starts fetching VR videos using the HTTP protocol. After processing the VR video, the emulator writes an output file containing the processed VR video performance metrics, as well as the raw performance data, as described in Section 3.2. The complete set of source code and datasets related to the VR-EXP platform are released under GNU General Public License v3.0 and are publicly available in the VR-EXP repository.

3.2 VR Video Client Emulator

We now focus on the high-level overview of the main functional components of the VR video client emulator. The VR-EXP video client emulator is an extensible and fully parameterized headless VR video client emulator. The emulator is composed of five main components (Figure 2): (*i*) sphere-to-plane projection handling, (*ii*) viewport prediction error injection, (*iii*) adaptive bitrate adaptation, (*iv*) tile fetching method, and (*v*) playout performance measurement. Next, we describe their functionality.

[1]Curl: https://curl.haxx.se/libcurl/c/.
[2]VR-EXP: https://https://github.com/rtcostaf/TOMM2019_VR-EXP/blob/master/README.md.
[3]Apache HTTP Server: https://httpd.apache.org/.

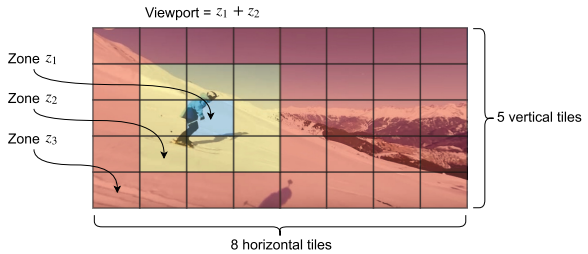


Fig. 3. Example of an 8×5 tiling scheme organized in three zones.

Sphere-to-plane Projection Handling. Several state-of-the-art approaches for VR video streaming optimization rely on tile-based projection schemes [8, 12, 15, 25]. Additionally, modern QoE estimation models employ tile clustering methods for manipulating groups of tiles in a coordinated way, depending on their spatial position [8]. To cope with these features, VR-EXP is designed to support different tiling schemes and tile clusterization into multiple zones. A multi-zone approach is in line with the notion that the spatial position in which a VR video degradation occurs is vital for estimating QoE. For example, Figure 3 depicts an 8×5 tiling scheme that is divided into three zones, where Zone 1 is defined as containing only the viewport's central tile, Zone 2 encompassing the viewport border tiles (8 tiles), and Zone 3 containing the 31 remaining tiles.

Viewport Error Injection. Once the projection handler is capable of dealing with several tiling schemes, the next step towards efficient VR streaming consists of emulating the viewport prediction. More precisely, to provide an accurate simulation of the entire VR context, the most significant information regarding any heuristic is the viewport prediction error. As discussed in Section 2.2, viewport prediction algorithms present highly variable accuracy depending on many factors. To allow for an accurate evaluation of error patterns, VR-EXP provides a controlled viewport error injection during the streaming session. We designed a flexible viewport error injection component that takes as input viewport traces (i.e., datasets describing the coordinates where the users have looked in a particular time frame). The viewport trace files contain a full record of coordinates of the VR video, captured at regular intervals (e.g., at each 20 ms). To provide a mechanism to evaluate the impact of wrong viewport predictions, VR-EXP enables injecting artificial prediction errors when processing the coordinates specified in the trace file. The error injection mechanism can be parameterized with a given error rate, as well as easily extended to support different error models. Using the viewport error injection can be very helpful in designing novel viewport prediction algorithms. Using VR-EXP, the developer can indeed measure the impact of the error on the resulting performance of the VR sessions. Also, VR-EXP allows understanding what would be the minimum error to guarantee a target performance. The error injection mechanism can be parameterized with a given error rate, as well as easily extended to support different error models. In the current version of VR-EXP, we modeled the viewport prediction error as a random variable with a uniform distribution. In other words, when an error occurs, the center tile of the viewport is moved—uniformly—to any other tile. Thus, the entire viewport will be shifted.

Dynamic Bitrate Adaptation. Taking advantage of the viewport prediction, ABR algorithms provide significant bandwidth savings by selecting appropriate quality representations for each spatial zone. In this procedure, each of the zones is assigned with the most suitable quality level according to both their distance from the center of the viewport and the available bandwidth. VR-EXP currently implements two alternative adaptive streaming heuristics. The general idea of the first heuristic procedure, named Full Delivery (FD) [25], is as follows: Once knowing the available bandwidth in the network (i.e., based on network conditions measured during the download of

previous segments), the emulator downloads tiles with the best fit regarding the available bandwidth. For each segment, the heuristic tries to first increase the bitrates on the inner zones of the viewport (Zone Z1 in Figure 3). Then, it repeats the procedure to stream tiles from the outer zones (Zones 2 and 3, respectively). Thus, when considering networks with enough available bandwidth, this heuristic will increase the quality representation for all zones. This approach provides effective protection against viewport prediction errors, at the cost of high bandwidth consumption. The second heuristic, named Full Delivery Basic (FDB) [11, 26], works similarly to the first one. However, instead of increasing the bitrate whenever possible in outer zones, this heuristic increases the bitrates only for zones within the viewport. Although FDB reduces the amount of consumed bandwidth significantly, it may entail QoE degradation in case of viewport prediction errors. Regardless of the approach, the downloaded segments are stored in a playout buffer to be eventually played. Observe that the buffer size plays a significant role in the VR video client performance—particularly regarding viewport prediction accuracy—and, therefore, can be adjusted as needed.

Tile Request Method. The combined use of tile-based VR videos, ABR heuristics, and viewport prediction have proven to be an effective approach to avoid wasting bandwidth. However, the adaptive tile-based video encoding leads to an increased number of files to be fetched from the content server. For example, consider a 10-minute tile-based VR video, split into 1-second segments, encoded with an 8×5 tiling scheme, and available on three quality representations (i.e., HD, FHD, and 4 K). For each second of video, it would be necessary to download one quality representation for each tile, which leads to 40 files per second; that is, 24 K files for a 10-minute streaming session. Considering the above, for each video segment, there is a set of tiles within pre-specified zones to be fetched from the server. VR-EXP allows fetching VR tiles according to two strategies: serial and parallel. On using the serial request method, tiles are fetched from the server, one-by-one, using multiple (non-parallel) HTTP requests, in a single connection. In turn, in the parallel method, tiles within the same zone (e.g., tiles belonging to the viewport) are fetched in parallel using a configurable number of parallel connections. Thus, when compared to a single threaded approach, multithreaded tile request methods can improve the VR video playout performance by reducing the stall time. VR-EXP allows specifying the number of threads per zone and splits the set of tiles uniformly among the available connections. It is worth mentioning that VR-EXP employs a regular HTTP server (e.g., Apache, NGINX) for hosting the tile-based VR videos, so no specific parameterization is required.

VR Video Playout Performance. To bring together the components detailed throughout this section, along with realistic input datasets, VR-EXP provides a realistic emulation of the VR video streaming ecosystem. Therefore, the next important step toward building a comprehensive VR video evaluation platform is to measure the VR video playout performance accurately. During the video streaming session, VR-EXP assesses a number of VR video playout performance metrics capable of objectively characterizing the quality of the video playout. These metrics include the number of tiles per zone/quality (e.g., number of tiles within the viewport retrieved in 4 K resolution), number of quality switches per zone (i.e., number of quality switches on a specific zone), stall time and startup time delay. These metrics were selected because they are the most influential when predicting QoE based on the video streaming playout performance [8]. It is worth mentioning that VR video applications rely on TCP/HTTP for providing reliable streaming services. Thus, network performance degradation events, such as packet loss or increased delay, will necessarily translate into either or both quality switches and video stall. Along these lines, VR-EXP focuses on evaluating how multiple VR video optimization techniques interact with variable network performance conditions. Evaluating the distortion introduced by different projection schemes and codecs is out of the scope of this work.

3.3 Network Performance Enforcing

To enforce real-world network performance conditions, it is possible to employ, at least, three different strategies: *(i)* network simulation, *(ii)* network emulation, or *(iii)* dedicated network infrastructure. The use of network simulation provides great control over the simulated elements. However, simulating the full VR video components stack, plus complex network aspects (e.g., routing, fairness between distinct TCP flavors, operating system features, and their limitations), would burden the complexity of implementation and potentially lead to inaccurate simulation results. At the other extreme, dedicated infrastructure provides a realistic environment at the cost of reduced flexibility and complex setup. In light of this, we decided to employ network emulation, as we consider this design choice a suitable balance between flexibility and accuracy.

For emulating network links, VR-EXP provides a customized SDN controller (based on Ryu[4]) which, along with Mininet,[5] enables reproducing sophisticated network scenarios. The SDN controller is the preferred option for complex network environments due to its ability to easily handle dynamic network topologies and forwarding rules. Also, this strategy allows evaluating the VR video streaming ecosystem when subjected to large topologies and high link competition through many concurrent video sessions. However, if the network scenario does not require such complexity (e.g., simulating a few links with static routes), the SDN approach could be replaced with a simpler alternative mechanism (i.e., Traffic Control[6]). Both approaches can benefit from simplified scripting to read input datasets, which describe the network performance (i.e., delay, jitter, residual bandwidth, packet loss) and enforce these network conditions on a target network.

3.4 QoE Model

VR-EXP is designed to work with any QoE model that supports VR video playout performance indicators as input. Employing a QoE model can be very insightful, as it provides a consolidated view regarding the effect of multiple VR video playout performance metrics on QoE. In consonance with state-of-the-art QoE models for traditional video streaming [20, 23, 36], we employ a QoE model [8] that is able to translate multiple VR video playout performance characteristics into an estimated QoE score.

$$\phi(Z_k) = \overbrace{\sum_{\forall t \in Z_k} \sum_{i=1}^{m} q(R(C_{t_i}))}^{\text{Quality}} - \overbrace{\mu \cdot \sum_{\forall t \in Z_k} \sum_{i=1}^{m} \left(T_d(R(C_{t_i})) - B_{t_i}\right)_+}^{\text{Stalls}} - \underbrace{\lambda \cdot \sum_{\forall t \in Z_k} \sum_{i=1}^{m-1} \left| q(R(C_{t_{i+1}})) - q(R(C_{t_i})) \right|}_{\text{Quality switches}} - \underbrace{\omega \cdot T_s}_{\text{Startup}} . \tag{1}$$

The QoE model is composed of four main terms, as shown in Equation (1). Each tile t is time-divided into m chunks $C = \{C_{t_1}, \ldots, C_{t_m}\}$. The first term uses a function $q : \mathbb{N}^+ \rightarrow \mathbb{N}^+$ that translates the measured bitrate of the chunk C_{t_m} (function $R : C_{t_m} \rightarrow \mathbb{N}^+$) into the quality perceived by the user. In this investigation, we considered the identity function $q(x) = x$. Function q is in line with the notion that different users may have a different perception regarding the bitrate of the VR video. For instance, some users may have a linear perception, which means that an increase of 50% in the video bitrate will be perceived as an increase of 50% in quality. In turn, other users may

[4]Ryu SDN Controller: https://osrg.github.io/ryu/.
[5]Mininet: https://mininet.org.
[6]TC: http://tldp.org/HOWTO/Traffic-Control-HOWTO/intro.html.

have a sub-linear quality perception, where the same increment in terms of bitrate is perceived as a marginal increment of quality [20]. The second term is used to keep track of the stall time. It considers, for each chunk C_{t_m}, that a stall event occurs when the download time T_d (defined as function $T_d : \mathbb{N}^+ \rightarrow \mathbb{N}^+$) is higher than the playout buffer (B_{t_m}). In addition, $q(R(C_{t_{m+1}})) - q(R(C_{t_m}))$ considers the quality switches between consecutive chunks, and T_s tracks the startup delay. Finally, constants μ, λ, ω are the non-negative weights used to adapt the model to different user sensitivities regarding degradation in VR video playout. For example, a higher value of μ with respect to the other weights means that the user is more susceptible to video stalls. Consequently, these events should affect the QoE indicator more severely.

Aiming to provide a more realistic assessment, the considered QoE model resorts to the concept of zones. The main idea of this approach relies on the notion that the QoE estimation must consider the spatial segmentation aspect of the VR videos. Along these lines, tiles near to the center of the viewport will greatly steer the quality perceived by the user, while bad qualities on tiles of the edge zones, or even outside the viewport, will potentially go unnoticed. For this reason, the overall video QoE ($\phi(V)$) is modeled as a weighted linear sum of the QoE measurement per zone (Equation (2)). Each weight ($\alpha_1, \alpha_2, \ldots, \alpha_k$) determines the relative importance of each zone. Note that the correspondence between tiles and zones during the rate adaptation task is independent of the one used to compute QoE. Due to the prediction error, a given tile Tk, which was initially predicted to belong to zone 3 (and thus fetched in low quality) may turn out belonging to zone 1, which ultimately contributes to QoE degradation. For the sake of completeness, we have chosen to set the weight of zone 3 to 0, since this zone will never be visualized by the user (regardless of the existence of prediction errors).

$$\phi(V) = \alpha_1 \cdot \phi(Z_1) + \alpha_2 \cdot \phi(Z_2) + \cdots + \alpha_k \cdot \phi(Z_k). \tag{2}$$

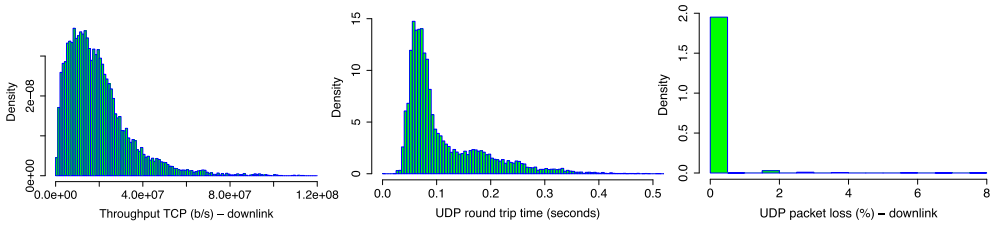
4 EVALUATION SETUP

Using VR-EXP as a basis, we carry out an extensive evaluation of state-of-the-art heuristics when subjected to variable network performance conditions. In this section, we present the experimental setup. We start by introducing the 4G/LTE performance dataset, which provides realistic network conditions to the evaluation process. Next, we describe the VR video dataset, including head track traces, which enables the evaluation of viewport-aware approaches. We end this section by outlining the experiment plan and its main procedures.

4.1 4G/LTE Performance Dataset

In this work, along with the VR-EXP method and toolkit, we provide a comprehensive dataset for 4G/LTE network performance. The dataset contains the following IP metrics: Round Trip Time (RTT), delay variation (also referred to as jitter), one-way packet loss, and one-way TCP throughput (in the scope of this work, also referred as to residual bandwidth). These metrics were gathered by means of IP active measurements, in conformance with the recommendations issued by the IETF IP Performance Metrics Working Group [21]. To obtain these indicators, we employed a scalable active measurement-based platform named Netmetric [7, 10, 30].

In Figure 4, we present a brief statistical analysis of the measurements available in the dataset regarding the three main metrics. As shown in Figure 4(a), the TCP throughput metric presents a wide range of measured values for the downlink. For example, the downlink presents a throughput varying from a minimum of 31.4 Kbps to a maximum of 113.2 Mbps, with a median of 16.5 Mbps and a mean of 19.6 Mbps. In turn, Figure 4(b) depicts the RTT metric ranging from 1 ms up to 18.5 seconds (the upper limit is not shown in the Figure 4(b) due to the long tail), with a median of 81 ms and a mean of 120 ms. Finally, the packet loss (Figure 4(c)) for the downlink ranges from 0% up to 8%.



(a) TCP throughput - downlink (b) Delay - RTT (c) Packet loss - downlink

Fig. 4. Network performance dataset: histograms for TCP throughput, delay, and packet loss.

The network dataset is composed of over 14 K measurements taken from 01/06/2017 to 31/07/2017. Each measurement considers the end-to-end path between the source node, a server located at the premises of the Federal University of Rio Grande do Sul, and a measurement device (destination). The destination of each measurement session is an Android (6.0) smartphone running the measurement agent and attached to a 4G/LTE network. Measurement devices were spread countrywide, embracing the four major mobile operators. Together, these operators are responsible for providing mobile services to over 236 million subscribers [31].

Each measurement session is composed of two bi-directional packet bursts, where the first uses UDP and the second TCP. The UDP packet burst is employed to measure RTT, loss, and jitter by injecting 400 packets of 100 bytes at 50 ms intervals. As some operators block the Network Time Protocol (NTP), we decided not to measure the One-Way Delay (OWD). Instead, the RTT metric was obtained based on a single clock (source). In turn, the TCP burst gauges the TCP throughput for the considered path by injecting 640 packets of 1,488 bytes each. For privacy reasons, sensitive information regarding the considered mobile operators (e.g., operator name, provider ID, cell ID) has been removed from the dataset.

Considering the number of measurements and the wide range of the considered metrics, the network performance dataset may be useful to support further research in several areas—especially in the field of VR video streaming, since the available metrics encompass the network performance indicators that influence video streaming performance the most (i.e., delay and residual bandwidth) [8, 36]. Additionally, the metrics' ranges allow evaluating high-resolution and tile-based VR videos, including 4K+ resolution. It is worth mentioning that the range for the TCP throughput metric is in line with similar studies conducted in other regions [32].

4.2 VR Video Dataset

In this evaluation, we use two VR videos from Wu et al.' s dataset [34], namely "Google Spotlight-HELP" and "Freestyle Skiing." Aiming at evaluating viewport-aware approaches, for each video, we also consider the available datasets that describe users' head movements while watching the VR videos. However, the original VR videos are non tile-based, so they needed to be re-encoded. To do so, the first step consisted of extracting the raw YUV files, making use of the Kvazaar encoder [33]. The resulting encoding produced two tiling schemes: 8×4 and 12×4 [18, 26]. Additionally, each tiling scheme was encoded into three quality representations, namely 720 p (1.8 Mbps), 1,080 p (2.7 Mbps), and 4 K (6 Mbps). Next, we employed the MP4Box[7] application to pack the encoded videos into MP4 containers. Then, we sliced each quality representation into 1-second segments. Finally, we used MP4Box to extract per-tile files and to generate the MPEG Dash Media Presentation Description (MPD) files considering multiple quality representations. Table 1 summarizes the main parameters regarding the VR video dataset.

[7]MP4Box https://gpac.wp.imt.fr/mp4box/.

Table 1. Adaptive Streaming Configurations

Videos	Qualities (bitrates)	Quality zones	Segment	Tiling	FPS
Google Spotlight Freestyle Skiing (Wu et al. [34])	720 p - 1.8 Mbps 1,080 p - 2.7 Mbps 4 K - 6 Mbps	Zone 1: 1 tile (central FoV) Zone 2: 8 tiles (adj. Zone 1) Zone 3: remaining tiles	1 s	12×8 8×4	30

4.3 Experiment Plan

VR-EXP was deployed on the imec iLab.t Virtual Wall emulation platform.[8] The experiments consisted of employing VR-EXP for measuring VR video performance while subjected to a broad variety of network conditions and multiple VR video optimization techniques. To capture the interplay between the considered variables (detailed in Section 3.2), we varied the experiment's parameters (e.g., network performance, VR video, tiling scheme, adaptive bitrate heuristic, playout buffer size) in a controlled manner. The experiments were organized around each key VR video optimization technique, namely the viewport prediction error, per-tile rate adaptation heuristics and tile requesting method. In a first step, we varied the parameters within each heuristic at a time, assuming default values for the remaining heuristics (according to Table 2). To capture the interplay within a set of heuristics, in the second step, we carried out a more sophisticated evaluation by varying multiple parameters and heuristics within the same experiment. To instantiate the QoE model, we consider the three-zone scheme defined by Da Costa Filho et al. [8], where Zone 1 refers to the viewport center tile, Zone 2 encompasses the eight tiles surrounding Zone 1, and Zone 3 includes all remaining tiles. We also consider the same constants and function values proposed by the authors, which are summarized as follows (refer to Equations (1) and (2)): $q = Linear$, $\mu = 4.3$, $\omega = 4.3$, $\lambda = 1$, $\alpha_1 = 0.7$, $\alpha_2 = 0.3$, and $\alpha_3 = 0$.

5 RESULTS

In this section, we present the results regarding the application of VR-EXP along with the inputs and parameters described in Section 4. We start by evaluating the effects of the Viewport Prediction Error (VPE) on VR video playout performance and QoE. Next, we extend this analysis to encompass per-tile rate adaptation heuristics and, finally, to tile requesting method. We end this section by presenting a more sophisticated scenario, where multiple parameters, heuristics, and the network performance conditions vary within the same experiment. It should be noted that, in addition to delay and bandwidth, the network dataset encompasses jitter and packet loss rate metrics. However, jitter and loss were not mentioned when characterizing network performance throughout this section. The reason is that during our experiments with VR videos, and in line with a previous study [8], such performance indicators were not shown expressive enough to model the performance of VR video and its respective QoE.

5.1 Effects of Viewport Prediction Error

When dealing with traditional 2D video streaming, we use the term video bitrate (e.g., 2 Mbps, 6 Mbps) equivalently with their respective representations of quality (e.g., 1,080 p, 4 K). Also, we can state that there is a correspondence between the average bitrate delivered to the user and the average bitrate that effectively traversed the network (i.e., bandwidth consumption). However, when it comes to tile-based VR video streaming, this relationship becomes less trivial. For example, consider the streaming of a tile-based VR video using a 12×4 tiling scheme and a viewport

[8]imec iLab.t: http://doc.ilabt.iminds.be/ilabt-documentation/virtualwallfacility.html.

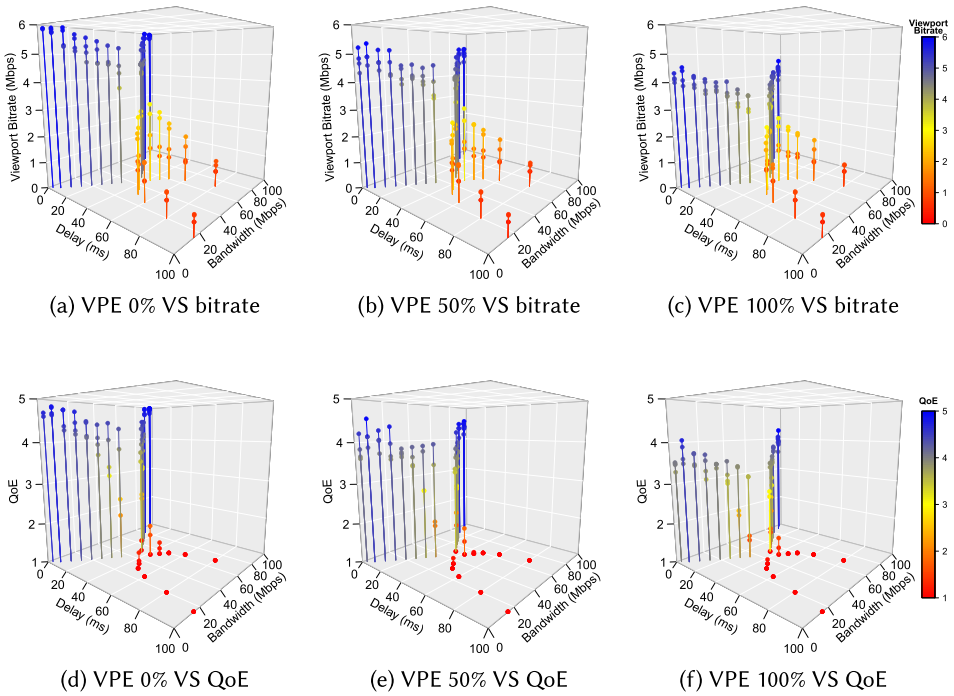
Table 2. Main VR-EXP Input Parameters

Parameter	Value/Range	Details
VR video	Google Spotlight-HELP and Freestyle Skiing	Both videos are used in all experiments
Head track traces	Google Spotlight-HELP and Freestyle Skiing	multiple users/head track traces for each video
Video format	MP4 - HEVC tile-based and HAS	Using MP4Box[9]
Video encoder	Kvazaar	Kvazaar encoder [33]
HAS	720 p (1.8 Mbps), 1,080 p (2.7 Mbps) and 4 K (6 Mbps)	Kvazaar encoder [33]
Segment size	1 second	the same for all experiments
Tiling scheme	8×4 and 12×4	Both tiling schemes are used in all experiments
Considered viewport	One central tile and eight border tiles	N/A
Viewport error rate	0% up to 100%	Default 0%
Rate adaptation heuristic	FD and BFD	Default BFD
Tile request method	Single thread, 6 threads and 8 threads	Default single thread
Client emulator monitoring interval	100 ms	Polling interval for the performance metrics
Experiment rounds for each configuration	6	Number of times each configuration is tested
Playout buffer	2 sec up to 8 sec	Default 2 sec

containing nine tiles. Assume that during most of the streaming session the viewport is displayed in 4 K resolution, while the tiles outside the viewport are fetched at 720 p. It turns out that the bitrate delivered to the user (visible portion of the VR video) is equivalent to the 4 K representation (i.e., 6 Mbps). However, when considering the FDB heuristic for adaptive bitrate, the overall bitrate of the video (i.e., equivalent to the average bandwidth demand during the streaming session) will be slightly higher than the bitrate of the 720 p representation. It happens because most of the video (not visible by the user) was fetched in low resolution. For didactic reasons, in this evaluation, we use the term *Viewport Bitrate* to denote the bitrate *perceived* by the user, while the term *Video Bitrate* refers to the total bitrate of the video (averaged over all tiles), being equivalent to the bitrate effectively demanded from the network.

As discussed in Section 2, depending on the viewport prediction algorithm and the playout buffer size, the viewport prediction accuracy can be quite erratic. In this section, we apply VR-EXP to evaluate the impact of the viewport prediction errors on both video playback performance and QoE. Figure 5 shows the performance of the video playout, regarding viewport bitrate and QoE, when subjected to variable network performance conditions and prediction error. Figure 5(a) illustrates the baseline scenario, characterized by absence of viewport prediction errors. In this scenario, a network delay below 12 ms is fundamental to provide good levels of viewport bitrate (recall that the bitrate for the 4 K representation is 6 Mbps). In such conditions, it is possible to observe viewport rates close to 6 Mbps across a wide range of available bandwidth values. Note that when considering the selected samples of our network dataset, even the lower values of residual

[9]MP4Box https://gpac.wp.imt.fr/mp4box/.



(a) VPE 0% VS bitrate (b) VPE 50% VS bitrate (c) VPE 100% VS bitrate

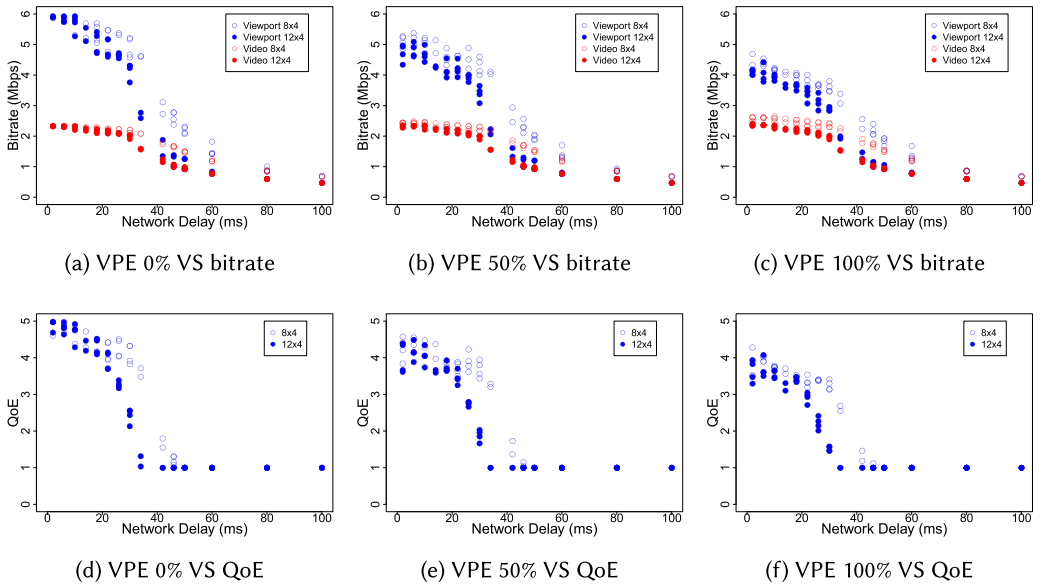
(d) VPE 0% VS QoE (e) VPE 50% VS QoE (f) VPE 100% VS QoE

Fig. 5. The effects of the viewport prediction error on VR video playout performance and QoE.

bandwidth allow to accommodate the viewport tiles in high quality and therefore achieve high viewport bitrates.

Figures 5(b) and 5(c) show how the viewport prediction error affects the viewport average bitrate. When considering a viewport prediction error rate equal to 50% (Figure 5(b)), the maximum bitrate decreases approximately by 1 Mbps, while a 100% error in the viewport (Figure 5(c)) drops the maximum bitrate to near 4 Mbps, even when considering the most favorable network condition. The viewport error does not affect the playout performance when subjected to significantly degraded levels of network performance (i.e., delay higher than 50 ms). In such cases, the rate adaptation algorithm has no room for increasing the quality representation. All tiles are requested at the lowest available quality representation and, as a direct consequence, a viewport error does not lead to additional degradation. Figures 5(d), 5(e), and 5(f) demonstrate the impact of prediction errors on QoE. One can observe that severe prediction errors (Figure 5(f)) may lead to a decrease of up to 2 points in the QoE score when compared to the baseline scenario shown in Figure 5(d).

Next, we employed VR-EXP to assess more accurately the effects of the viewport prediction error. To do so, we added the tile scheme information. Moreover, we split the rates between the bitrate observed for the tiles within the viewport and the bitrate for the entire video (including the viewport). Figure 6(a) shows the baseline case, which considers a perfect viewport prediction. To improve readability, in all plots of Figure 6, we show the network variability only in terms of delay, removing the bandwidth dimension from the analysis. The red dots represent the bitrate for the entire VR video (i.e., viewport + remaining tiles), which is equal to the network bandwidth required for streaming the VR video. When it comes to the viewport (blue dots), both tiling schemes are able to achieve the maximum bitrate when the delay is lower than 12 ms. However, the 8×4 tiling scheme presents significantly better bitrates for intermediate network conditions (delay between 12 ms and 60 ms). This gain is explained by the fact that the HTTP request/response overhead is



(a) VPE 0% VS bitrate (b) VPE 50% VS bitrate (c) VPE 100% VS bitrate

(d) VPE 0% VS QoE (e) VPE 50% VS QoE (f) VPE 100% VS QoE

Fig. 6. The effects of the viewport prediction error and tiling scheme on playout performance and QoE.

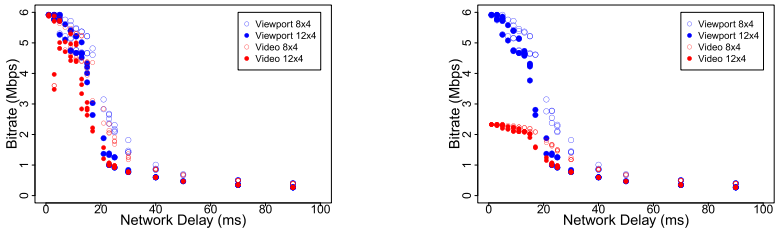
lower for the 8×4 tiling scheme (32 files per segment) against 48 files per segment for the 12×4 tiling scheme. When the delay is higher than 60 ms, the video playout is totally impaired, and neither the tiling scheme nor the VPE introduces additional degradation.

Complementing the previous analysis, in Figures 6(b) and 6(c) it is possible to observe that the tiling scheme plays an important role in the video playout performance. The viewport error leads to lower viewport bitrate for intermediate network conditions when compared to the baseline scenario. Still, for intermediate network delay, the 8×4 tiling scheme presents a viewport bitrate up to 2 Mbps higher when compared to the 12×4 tiling scheme. The obtained results indicate that the VPE influences mainly the average viewport bitrate and quality switch metrics. The remaining metrics for playout performance (i.e., startup delay and stall time) are not affected by prediction errors. Figures 6(d), 6(e), and 6(f) show that, in line with previous findings, the viewport prediction error has the potential to reduce the QoE score significantly. Nevertheless, the tiling scheme can dramatically influence the QoE score. For example, in Figure 6(d) it is possible to observe that, for a network delay of around 35 ms, the 8×4 tiling scheme outperforms the 12×4 by more than 2 points in the expected QoE score.

Main insight for viewport prediction error. Increased levels of VPE may result in reduced viewport quality and QoE. The VPE does not introduce further degradation when subjected to low-performance networks. The tiling scheme has the potential to highly affect QoE when considering intermediate levels of prediction error and network performance.

5.2 Per-tile Rate Adaptation Heuristics

As discussed in Section 2, the tile-based rate adaptation algorithm is crucial for achieving a suitable balance between playout performance and network bandwidth consumption. Although VR-EXP can be extended to encompass several strategies, in this section, we focus on two distinct approaches, namely the Full Delivery (FD) [25] and the Full Delivery Basic (FDB) [12]. Recall that both approaches request the tiles inside the viewport in the highest possible quality



(a) Full Delivery (b) Full Delivery Basic

Fig. 7. Dynamic rate adaptation heuristics: FD and FDB.

representation. The main difference between them is that, depending on the network residual bandwidth, the FD method attempts to increase the bitrate for all the tiles, including the ones outside the viewport. Conversely, the FDB approach does not increase the quality representation for tiles outside the viewport, regardless of the available bandwidth.

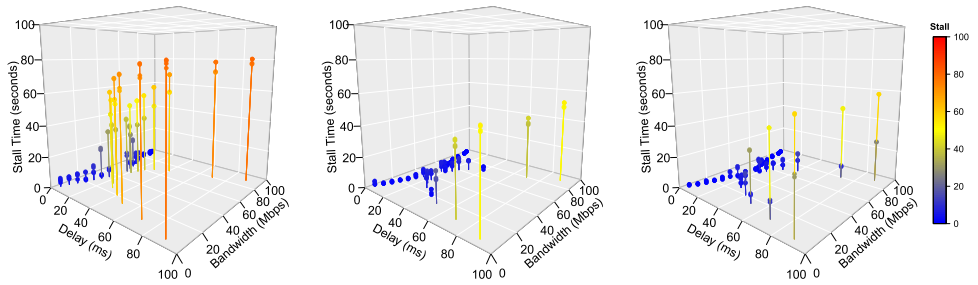
Figures 7(a) and 7(b) show the relationship between the measured viewport bitrate (blue) and the entire video bitrate (red), when subjected to variable network performance conditions. The difference between FD and FDB is more noticeable when the delay is lower than 20 ms. In this case, FD benefits from the available network performance to maximize the quality representation of the entire video. One key advantage of the FD approach is its natural protection against viewport prediction errors, at the cost of increased bandwidth consumption. However, when considering methods for viewport prediction with low error rates, the FDB method may represent a better choice, as it will maintain good levels of QoE while avoiding bandwidth waste. For intermediate network delay (between 20 and 40 ms), both methods perform similarly, because the network performance is sufficient to accommodate only the viewport in high quality. Finally, for a network delay higher than 40 ms, there is no room for increasing the quality representation at all, and both strategies present equivalent performance.

Main insight for rate adaptation heuristics. The FD heuristic provides excellent protection against viewport prediction errors at the cost of increased bandwidth consumption. If combined with low-error viewport prediction algorithms, then FDB may potentially lead to reduced bandwidth consumption.

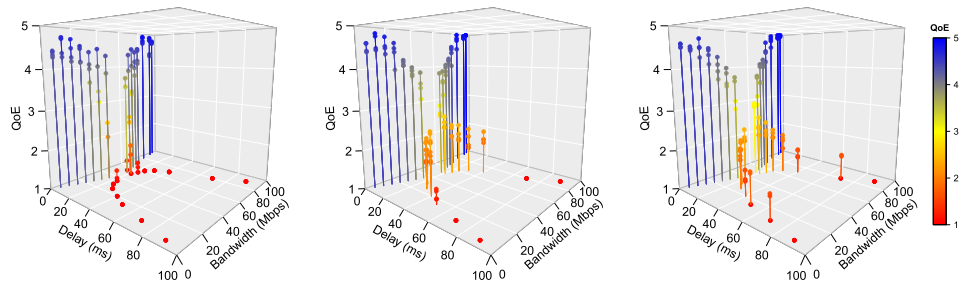
5.3 Multithreaded Tile Downloading

As discussed in Section 3, the network delay is the QoS metric that affects video playout performance the most. The reason is that high levels of network delay, when combined with both short video segments and tiling scheme overhead, limit the download throughput. As shown in Figure 8(b), when using six threads it is possible to dramatically reduce the VR video stall time. Basically, when compared to the single thread approach (Figure 8(a)), the use of six threads enables handling twice as much network delay (from 20 ms to 40 ms) while maintaining the same level of stall time. When resorting to 10 threads for tile downloading (Figure 8(c)) it was possible to slightly reduce the stalling time, especially when considering VR videos using the 8×4 tiling scheme (as discussed next).

Figures 8(e) and 8(f) depict the effects of the multithreaded approach in the QoE score. When compared to the single thread (Figure 8(d)), the multithreaded approach is able to increase the QoE score by up to 1.5 points when the delay is higher than 20 ms. However, for network delays higher than 80 ms, the QoE is completely degraded, regardless of the available bandwidth and the use of multithreaded approaches.

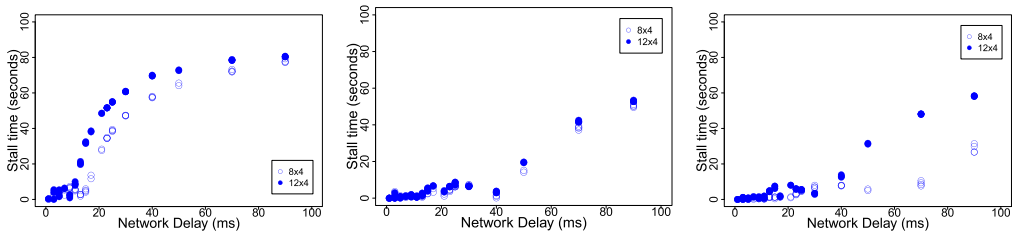


(a) Single thread - stall time (b) Six threads - stall time (c) Ten threads - stall time



(d) Single thread - QoE (e) Six threads - QoE (f) Ten threads - QoE

Fig. 8. Multi-thread effect on VR video stall time.



(a) Single thread - stall time (b) Six threads - stall time (c) Ten threads - stall time

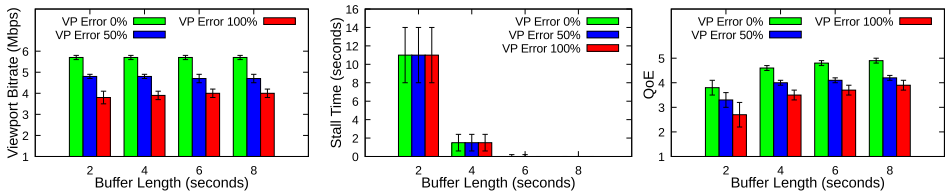
Fig. 9. Multi-thread: stall time and tiling scheme VS network delay.

Figure 9 shows the effect of the multithreaded approach on distinct tiling schemes (i.e., 8×4 and 12×4). When considering a network delay of 40 ms, the six-threads approach outperforms the single thread by reducing the stall time from 60 to 5 seconds (approximately) (Figures 8(a) and 8(b)). The experiment with six threads resulted in similar results for both tiling schemes, with a slight advantage to the 8×4 one. In turn, the ten-thread experiment variation (Figure 8(c)) led to an additional reduction of the stall time for the 8×4 scheme, but not for the 12×4, which presented roughly the same results when compared to the six-thread experiment.

Main insight for multithreaded tile downloading. Multithreaded tile fetching can dramatically reduce the stall time and increase the QoE score for intermediate levels of network performance. However, it does not provide noticeable improvements in QoE for either high or low network performance.

Table 3. Network Performance Indicators within a 60-second-long VR Video Session

Conf. ID	Delay (ms)	Bandwidth (Mbps)
1	1	74
2	4	38
3	55	31
4	2	60
5	4	54
6	95	8
7	6	22
8	1	84
9	49	19
10	87	7



(a) Viewport average bitrate (b) Stall time (c) QoE

Fig. 10. The influence of multiple VR video optimization techniques on VR video streaming playout performance and QoE.

5.4 Buffer Size and Viewport Prediction Error

The evaluation carried out earlier in this section has focused on evaluating the effects of each VR video optimization technique on VR video playout performance and QoE. Aiming to further explore the interplay among different VR video optimization techniques, in this experiment, we evaluate a set of four optimization aspects simultaneously: namely, variable viewport scheme, variable viewport prediction error, variable buffer size, and the FDB rate adaptation approach. In experiments 5.1 through 5.3, the IP performance values are fixed within each video session, allowing us to capture how QoE is affected by each network performance condition. In this experiment, instead of evaluating how the optimization techniques perform when subjected to distinct network performance conditions, we vary the network conditions within the VR video session. Table 3 shows 10 distinct combinations of network performance indicators that were randomly selected within the range for each QoS metric (as discussed in Section 4). A particular VR video session lasts for 60 seconds, where each network performance configuration lasts for 6 seconds, starting with the configuration ID 1 up to the ID 10. The main objective of this experiment is to evaluate the interplay between multiple VR video optimization approaches while subjected to highly variable network performance conditions. To provide a generalized analysis, the results presented in Figure 10 represent the averaged values when considering the entire VR video dataset. Therefore, the error bars, in this case, represent the min-max range for each histogram bin.

Figure 10(a) shows the average quality observed for the viewport when streaming VR videos subjected to variable buffer size and viewport prediction error rates. As discussed in Section 2, for most state-of-the-art viewport prediction algorithms, the bigger the buffer size, the higher the

prediction error rate. Aiming at evaluating a broad range of scenarios, the analysis presented in Figure 10(a) used a full factorial experiment design considering different values for buffer size and error rate. The obtained results indicate that the viewport prediction error greatly affects the viewport bitrate, while the buffer size itself does not have noticeable influence on it.

Figure 10(b) shows that the increased buffer size was able to dramatically reduce the stall time. For example, when considering a playout buffer dimensioned for 4 seconds of video, the stall time drops from 11 seconds to less than 2 seconds (on average). Furthermore, when increasing the buffer to 8 seconds, it was possible to completely eliminate the stall time. However, as discussed in Section 2, most state-of-the-art viewport prediction algorithms experiment sudden accuracy drop when increasing the playout buffer size. Hence, the effective analysis of the interplay between buffer size and viewport error must be done through the evaluation of the QoE indicator, since the QoE score will simultaneously consider both playout performance metrics. Figure 10(c) shows that, when using 8 seconds of playout buffer, the worst-case scenario for the QoE score (i.e., viewport prediction error of 100%) performs on par with the best-case scenario of the 2-second buffer (i.e., viewport prediction error of 0%). Furthermore, due to the human randomness, prediction algorithms may present low accuracy even when considering small buffers (e.g., 2 seconds). Therefore, using higher values for dimensioning the playout buffer (e.g., 8 seconds) will probably outperform smaller buffer setups in most cases.

Main insight for mixed buffer size and prediction error. When dealing with realistic performance levels, increasing the playout buffer size may potentially lead to a better QoE score, even considering the likely increase in the prediction error.

6 CONCLUSION

VR video streaming applications are growing fast. To cope with the huge demand for network resources, both the scientific community and the industry have proposed optimization techniques for VR videos. However, the complex interplay between VR video optimization techniques and variable network conditions challenges developers of VR video solutions, as this interaction is neither trivial nor has it been properly investigated. Additionally, a publicly available solution to provide a reproducible and in-depth evaluation of the VR video realm is still missing.

To address this problem, we proposed VR-EXP, an open-source platform for evaluating adaptive VR video streaming that encompasses various optimization techniques and allows for network performance conditions to be varied. To support realistic evaluation, we provide a 4G/LTE performance dataset composed of multiple network performance metrics. Employing VR-EXP, along with realistic datasets, we have produced an extensive assessment that examines the performance of several state-of-the-art optimization techniques when subjected to variable network conditions. The results obtained evidence that the relationship between different optimization techniques for video VR optimization is not trivial. Mainly, because certain combinations can benefit one aspect of reproduction and impair others. For example, the increased buffer size, combined with the FDB approach, may lead to increased viewport prediction error. In this case, the viewport bitrate will be degraded and the stall time will be reduced. By combining an objective assessment of VR video streaming playout performance and a comprehensive QoE model, VR-EXP allowed pinpointing the components of the VR video ecosystem that most affect the performance of VR video playout and, ultimately, QoE.

The benefits of this work are twofold. From the VR video developers' perspective, we expect to contribute a useful approach to conducting a precise and realistic performance evaluation of novel optimization techniques. In turn, from the mobile operator's perspective, we expect VR-EXP to be

a valuable tool for supporting investigations aimed at understanding and predicting how variable network conditions impact VR video performance and QoE delivered to their end-users.

REFERENCES

[1] Zahaib Akhtar, Yun Seong Nam, Ramesh Govindan, Sanjay Rao, Jessica Chen, Ethan Katz-Bassett, Bruno Ribeiro, Jibin Zhan, and Hui Zhang. 2018. Oboe: Auto-tuning video ABR algorithms to network conditions. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM'18)*. ACM, New York, NY, 44–58. DOI: https://doi.org/10.1145/3230543.3230558

[2] Mathias Almquist, Viktor Almquist, Vengatanathan Krishnamoorthi, Niklas Carlsson, and Derek Eager. 2018. The prefetch aggressiveness tradeoff in 360deg video streaming. In *Proceedings of the 9th ACM Multimedia Systems Conference (MMSys'18)*. ACM, New York, NY, 258–269. DOI: https://doi.org/10.1145/3204949.3204970

[3] Jill Boyce, Elena Alshina, Adeel Abbas, and Yan Ye. 2017. JVET common test conditions and evaluation procedures for 360 video. *Joint Video Exploration Team of ITU-T SG* 16 (2017). ITU - International Telecommunication Union. Retrieved on 21 June, 2010 from http://phenix.it sudparis.eu/jvet/doc_end_user/documents/16_Geneva/wg11/JVET-P0006-v1.zip.

[4] Zhenzhong Chen, Yiming Li, and Yingxue Zhang. 2018. Recent advances in omnidirectional video coding for virtual reality: Projection and evaluation. *Sig. Proc.* 146 (2018), 66–78. DOI: https://doi.org/10.1016/j.sigpro.2018.01.004

[5] Cisco. 2019. *Cisco Visual Networking Index: Forecast and Trends, 2017–2022*. Technical Report. Cisco Systems.

[6] X. Corbillon, G. Simon, A. Devlic, and J. Chakareski. 2017. Viewport-adaptive navigable 360-degree video delivery. In *Proceedings of the IEEE International Conference on Communications (ICC'17)*. 1–7. DOI: https://doi.org/10.1109/ICC.2017.7996611

[7] R. I. T. da Costa Filho, W. Lautenschlager, N. Kagami, V. Roesler, and L. P. Gaspary. 2016. Network fortune cookie: Using network measurements to predict video streaming performance and QoE. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM'16)*. 1–6. DOI: https://doi.org/10.1109/GLOCOM.2016.7842022

[8] Roberto Irajá Tavares da Costa Filho, Marcelo Caggiani Luizelli, Maria Torres Vega, Jeroen van der Hooft, Stefano Petrangeli, Tim Wauters, Filip De Turck, and Luciano Paschoal Gaspary. 2018. Predicting the performance of virtual reality video streaming in mobile networks. In *Proceedings of the 9th ACM Multimedia Systems Conference (MMSys'18)*. ACM, New York, NY, 270–283. DOI: https://doi.org/10.1145/3204949.3204966

[9] Giorgos Dimopoulos, Ilias Leontiadis, Pere Barlet-Ros, and Konstantina Papagiannaki. 2016. Measuring video QoE from encrypted traffic. In *Proceedings of the Internet Measurement Conference (IMC'16)*. ACM, New York, NY, 513–526. DOI: https://doi.org/10.1145/2987443.2987459

[10] Glederson Lessa dos Santos, Vinicius Tavares Guimaraes, Jorge Guedes Silveira, Alexandre T. Vieira, Jose Augusto de Oliveira Neto, R. I. T. da Costa, and Ricardo Balbinot. 2007. UAMA: A unified architecture for active measurements in IP networks; end-to-end objetive quality indicators. In *Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Network Management (IM'07)*. 246–253.

[11] Ching-Ling Fan, Jean Lee, Wen-Chih Lo, Chun-Ying Huang, Kuan-Ta Chen, and Cheng-Hsin Hsu. 2017. Fixation prediction for 360 video streaming in head-mounted virtual reality. In *Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'17)*. ACM, New York, NY, 67–72. DOI: https://doi.org/10.1145/3083165.3083180

[12] Mario Graf, Christian Timmerer, and Christopher Mueller. 2017. Towards bandwidth-efficient adaptive streaming of omnidirectional video over HTTP: Design, implementation, and evaluation. In *Proceedings of the 8th ACM Conference on Multimedia Systems Conference (MMSys'17)*. ACM, New York, NY, 261–271. DOI: https://doi.org/10.1145/3083187.3084016

[13] Jian He, Mubashir Adnan Qureshi, Lili Qiu, Jin Li, Feng Li, and Lei Han. 2018. Favor: Fine-grained video rate adaptation. In *Proceedings of the 9th ACM Multimedia Systems Conference (MMSys'18)*. ACM, New York, NY, 64–75. DOI: https://doi.org/10.1145/3204949.3204957

[14] M. Hosseini and V. Swaminathan. 2016. Adaptive 360 VR video streaming: Divide and conquer. In *Proceedings of the IEEE International Symposium on Multimedia (ISM'16)*. 107–110. DOI: https://doi.org/10.1109/ISM.2016.0028

[15] Xueshi Hou, Sujit Dey, Jianzhong Zhang, and Madhukar Budagavi. 2018. Predictive view generation to enable mobile 360-degree and VR experiences. In *Proceedings of the Morning Workshop on Virtual Reality and Augmented Reality Network (VR/AR Network'18)*. ACM, New York, NY, 20–26. DOI: https://doi.org/10.1145/3229625.3229629

[16] H. Hristova, X. Corbillon, G. Simon, V. Swaminathan, and A. Devlic. 2018. Heterogeneous spatial quality for omnidirectional video. In *Proceedings of the IEEE 20th International Workshop on Multimedia Signal Processing (MMSP'18)*. 1–6. DOI: https://doi.org/10.1109/MMSP.2018.8547114

[17] E. Jeong, D. You, C. Hyun, B. Seo, N. Kim, D. H. Kim, and Y. H. Lee. 2018. Viewport prediction method of 360 VR video using sound localization information. In *Proceedings of the 10th International Conference on Ubiquitous and Future Networks (ICUFN'18)*. 679–681. DOI: https://doi.org/10.1109/ICUFN.2018.8436981

[18] Ron Kohavi et al. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'95)*, Vol. 14. 1137–1145.

[19] L. Ma, Y. Xu, J. Sun, W. Huang, S. Xie, Y. Li, and N. Liu. 2018. Buffer control in VR video transmission over MMT system. In *Proceedings of the IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB'18)*. 1–5. DOI: https://doi.org/10.1109/BMSB.2018.8436817

[20] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. 2017. Neural adaptive video streaming with Pensieve. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM'17)*. ACM, New York, NY, 197–210. DOI: https://doi.org/10.1145/3098822.3098843

[21] A. Morton. 2016. *RFC 7799 - Active and Passive Metrics and Methods (with Hybrid Types In-Between)*. Technical Report. IETF.

[22] T. C. Nguyen and J. Yun. 2018. Predictive tile selection for 360-degree VR video streaming in bandwidth-limited networks. *IEEE Commun. Lett.* 22, 9 (Sept. 2018), 1858–1861. DOI: https://doi.org/10.1109/LCOMM.2018.2848915

[23] Stefano Petrangeli, Jeroen Famaey, Maxim Claeys, Steven Latré, and Filip De Turck. 2015. QoE-driven rate adaptation heuristic for fair adaptive video streaming. *ACM Trans. Multimedia Comput. Commun. Appl.* 12, 2, Article 28 (Oct. 2015), 24 pages. DOI: https://doi.org/10.1145/2818361

[24] S. Petrangeli, G. Simon, and V. Swaminathan. 2018. Trajectory-based viewport prediction for 360-degree virtual reality videos. In *Proceedings of the IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR'18)*. 157–160. DOI: https://doi.org/10.1109/AIVR.2018.00033

[25] Stefano Petrangeli, Viswanathan Swaminathan, Mohammad Hosseini, and Filip De Turck. 2017. An HTTP/2-Based adaptive streaming framework for 360 virtual reality videos. In *Proceedings of the ACM on Multimedia Conference (MM'17)*. ACM, New York, NY, 306–314. DOI: https://doi.org/10.1145/3123266.3123453

[26] Feng Qian, Lusheng Ji, 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 (ATC'16)*. ACM, New York, NY, 1–6. DOI: https://doi.org/10.1145/2980055.2980056

[27] Iraj Sodagar. 2011. The MPEG-DASH standard for multimedia streaming over the internet. *IEEE Multimedia* 18, 4 (2011), 62–67.

[28] Kevin Spiteri, Ramesh Sitaraman, and Daniel Sparacio. 2018. From theory to practice: Improving bitrate adaptation in the DASH reference player. In *Proceedings of the 9th ACM Multimedia Systems Conference (MMSys'18)*. ACM, New York, NY, 123–137. DOI: https://doi.org/10.1145/3204949.3204953

[29] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman. 2016. BOLA: Near-optimal bitrate adaptation for online videos. In *Proceedings of the 35th IEEE International Conference on Computer Communications (INFOCOM'16)*. 1–9. DOI: https://doi.org/10.1109/INFOCOM.2016.7524428

[30] K. Stangherlin, R. C. Filho, W. Lautenschläger, V. Guadagnin, L. Balbinot, R. Balbinot, and V. Roesler. 2011. One-way delay measurement in wired and wireless mobile full-mesh networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference*. 1044–1049. DOI: https://doi.org/10.1109/WCNC.2011.5779279

[31] TELCO. 2018. Mobile Operators Market Share in Brazil. Retrieved from: http://www.teleco.com.br/en/en_mshare.asp.

[32] J. van der Hooft, S. Petrangeli, T. Wauters, R. Huysegems, P. R. Alface, T. Bostoen, and F. De Turck. 2016. HTTP/2-Based adaptive streaming of HEVC video over 4G/LTE networks. *IEEE Commun. Lett.* 20, 11 (2016), 2177–2180.

[33] M. Viitanen, A. Koivula, A. Lemmetti, J. Vanne, and T. D. Hämäläinen. 2015. Kvazaar HEVC encoder for efficient intra coding. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'15)*. 1662–1665. DOI: https://doi.org/10.1109/ISCAS.2015.7168970

[34] Chenglei Wu, Zhihao Tan, Zhi Wang, and Shiqiang Yang. 2017. A dataset for exploring user behaviors in VR spherical video streaming. In *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17)*. ACM, New York, NY, 193–198. DOI: https://doi.org/10.1145/3083187.3083210

[35] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. 2015. A control-theoretic approach for dynamic adaptive video streaming over HTTP. In *Proceedings of the ACM Conference on Special Interest Group on Data Communication (SIGCOMM'15)*. ACM, New York, NY, 325–338. DOI: https://doi.org/10.1145/2785956.2787486

[36] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. 2015. A control-theoretic approach for dynamic adaptive video streaming over HTTP. *SIGCOMM Comput. Commun. Rev.* 45, 4 (Aug. 2015), 325–338. DOI: https://doi.org/10.1145/2829988.2787486

[37] Chao Zhou, Zhenhua Li, Joe Osgood, and Yao Liu. 2018. On the effectiveness of offset projections for 360-degree video streaming. *ACM Trans. Multimedia Comput. Commun. Appl.* 14, 3s, Article 62 (June 2018), 24 pages. DOI: https://doi.org/10.1145/3209660

Received February 2017; revised July 2019; accepted September 2019