Effects of Adaptive Streaming Optimizations on the Perception of 360° Virtual Reality Video

Joris Heyse, Maria Torres Vega, Tim Wauters, Femke De Backere, Filip De Turck
IDLab-imec
Ghent University
9052, Ghent, Belgium
Email: {joris.heyse, maria.torresvega, tim.wauters, femke.debackere, filip.deturck}@ugent.be

***Abstract*—As the demand of Virtual Reality (VR) video streaming to mobile devices increases, novel optimization transport techniques need to be designed to cope with these ultra-high-bandwidth video services. One approach currently attracting attention is the application of adaptive tile-based streaming solutions to the VR video arena. The VR videos are encoded at different quality levels, temporally divided into segments and spatially split into tiles. During the streaming, each of these tiles can be transmitted independently according to its location within the frame (*i.e.*, within or outside of the user's field of view). These methods open a new venue for bandwidth and latency optimization for the streaming of VR videos. However, the effect of the different adaptive streaming optimizations on the end-user perception is still an open research topic. In this demo, we present a VR video platform to experience the working principles of adaptive tile-based VR video streaming services. Through different tiling approaches, bandwidth conditions and viewport algorithms, it allows the users to explore the effects that each optimization has on the perception of the service. In addition, the platform provides real-time bandwidth savings and objective Quality of Experience (QoE) measurements to provide a quantitative analysis of the optimizations effects. This demo aims to provide a common playground for researchers to benchmark and evaluate the performance of their optimization solutions.**

***Index Terms*—Virtual Reality Video Streaming, Tiling, Quality of Experience**

I. Introduction

The demand of streaming of Virtual Reality (VR) video to mobile devices is booming as Head Mounted Devices (HMDs) increase their penetration rate, with more effective and affordable solutions. Examples of such are the Google's Cardboard[1] or the Samsung Gear VR[2]. In fact, the VR video streaming traffic is expected to increase 20-fold by 2021 [1].

A VR video contains a full 360° panoramic view [2]. Thus, its streaming demands significantly higher bandwidths than traditional 2D video applications, even if only a fraction of the spherical view, namely the viewport (or Field of View-FoV), is visible at any given instant. These required ultra high bandwidths are not always available in wireless networks, and are not easy to process by lightweight mobile devices. In this situation, there is a need for designing novel intelligent techniques to cope with the network conditions and the users' requirements.

[1]Google Cardboard https://goo.gl/DSquZf
[2]Gear VR https://goo.gl/7JdQm7

In an attempt to optimize the bandwidth usage, viewport-aware schemes for VR video streaming based on the HAS (HTTP Adaptive Streaming) paradigm are currently being explored [3], [4]. HAS focuses on encoding the source content at multiple quality representations (bitrates), while each quality representation is time-segmented into small parts (*i.e.*, segments). To further optimize bandwidth usage, recent research investigations have proposed HAS variants in which quality representations are not only segmented in time but also spatially split into smaller pieces (*i.e.*, tiles) [5]. Approaches such as [6], [7], [8] bring the adaptive segment-based tiled-based streaming concept to the challenging VR arena. To this end, the VR videos are encoded at different quality levels, temporally divided in segments and spatially tiled [7]. Then, during the streaming session, only the tiles within the viewport are streamed at the highest quality, while others are maintained at lower levels or not streamed at all [8]. To be effective, these techniques rely on viewport prediction algorithms [6].

Adaptive streaming, viewport prediction, bandwidth awareness and tiling techniques lead to bandwidth optimizations but they have also the potential to severely impair the user's Quality of Experience (QoE). For example, considering that only the predicted viewport's tiles are downloaded in advance, predictions errors may lead to application level degradation (*i.e.*, video stall, quality switch) and, consequently, QoE degradation. However, the degree in which each of the optimizations might influence the QoE is still an open topic of research and discussion [2].

Herein, we present a VR video streaming platform to explore and perceive the effects of the working principles of adaptive tile-based VR video streaming services. Through different tiling approaches, bandwidth conditions and viewport algorithms, our platform allows the users to explore the effects that each optimization has on the perception of the service. To enable full control and comparison of all the conditions and reduce the computation on the client, we move the full adaptive streaming functionality to the server, leaving the client device merely responsible for VR playout and viewport position monitoring. In addition, the platform provides with a dashboard with real-time bandwidth monitoring and objective Quality of Experience (QoE) measurements. Both the subjective (users) and the objective (network and metrics) evaluations will allow us the creation of a mapping of the effects of VR streaming

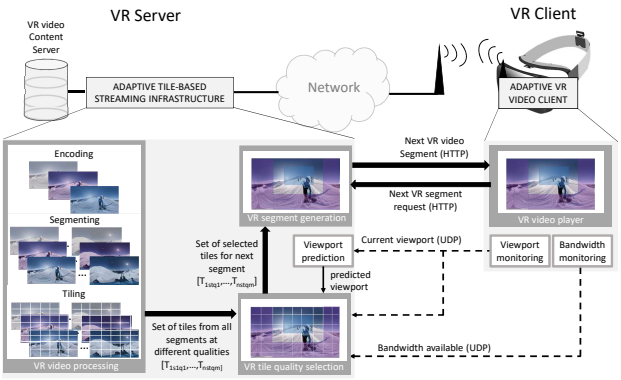

Fig. 1. Block diagram of the proposed adaptive tile-based VR end-to-end approach.

services onto QoE. The remainder of this paper discusses the working principles of the approach, and describes the demo.

II. Adaptive tile-based VR Video Streaming managed by the Server

Most of the state-of-the-art adaptive video streaming applications (for 2D, 3D or VR videos) concentrate the computation (*i.e.*, the decisions regarding the selection of next tiles and segments) in the client application. Such is the case of the standards for 2D adaptive streaming (MPEGDASH [4]) or the approaches for VR of Petrangeli et al. [6] or Quiang et al. [8]. These methods enable higher scalability. However, when dealing with segments, tiles and VR video, deploying the core computation on the client (usually lightweight smartphones) can lead to unwanted delays and freezes (the client needs to receive and merge the tiles, and display the resulting segment, while processing, predicting the location of the user and requesting the tiles for the next segment). In addition, these client-based methods fail to provide flexible solutions to assess and compare the performance and effects of the different viewport prediction algorithms and tiling/segmenting schemes during the same session and using the same conditions, which is the ultimate goal of our approach and demonstrator. Thus, we decided to move the major part of the functionality and computation to the server, leaving to the client the sole responsibility of displaying the incoming segments and of monitoring the HMD position within the 360° video and sensing the network.

Figure 1 presents our end-to-end streaming method. The VR video server stores the original VR video content (typically a set of 360° videos at the highest quality available). Prior to the streaming, each of the videos is pre-processed according to the adaptive tile-based streaming concept. First, they are encoded at different quality levels (bitrates and resolutions). Second, the compressed streams are temporally divided into segments (of duration ranging from 500 ms to 10s). Finally, each of these segments belonging to different video sources and compression is spatially split into spatial areas (tiles). As a result, a set of tiles related to multiple segments, qualities and video sources is generated.

To start the streaming, the client registers in the server, receiving a ready response when the server is prepared. To speed up the process, this registration is done using the User Datagram Protocol (UDP) which does not require acknowledgments. After the registration, the streaming starts, using the standard HAS protocol. This means that the client first requests the media presentation descriptor (MPD), and subsequently asks the segments described by it. As explained before, our client application is not required to take any decision regarding qualities of tiles or segments. Therefore, the mpd file consists of a list of as many segments as the video is composed of. During the streaming, the client monitors both the location of the HMD (by means of the available sensors in the smartphone) and the available bandwidth. Both measurements are sent back to the server using again the UDP protocol. On reception, the server generates the next segment.

Generating the segment consists of three phases: (1) viewport prediction, (2) quality assignments and, (3) tile merging. First, the current viewport information (received from the client) is used to provide the prediction of the future location of the user. Several viewport prediction algorithms have appeared in the last years, ranging from simple adaptive, non predictive approaches (in which the algorithm considers that the user will not move in the next segment), to more complex ones, such as the probabilistic approach of Xie et al. [9]. However, the results reported have been usually shown on small test sets (one video), making the validation far from generic. This is due to the fact that not only users, but also different video content types will heavily influence the accuracy of the algorithm. The only path to evaluate the performance of the different algorithms is to do so in a comparative manner using the same broad range of conditions, users and video content. The server-based adaptive streaming end-to-end system presented herein enables exactly that as we show in Section III. Second, it decides the qualities to assign to the different tiles depending on the bandwidth available and the just predicted viewport. As for viewport prediction, there is no generic solution on how to select the qualities of the different tiles. In our approach, we select the quality-zones approach presented in [2]. The tiles are distributed in zones according to their distance to the viewport, in such a way that only the tiles within the viewport are selected at the highest quality, while the edges are at the second highest quality, and so on. Finally, it merges the selected tiles (already stored in the server) and prepares the next segment that the client will request, when the current one is nearly finishing playback.

III. Experimental Demonstrator

As proof of concept demonstrator, we envision a local network in which the server and the client are connected by means of a wireless access point (Figure 2). The client consists of a Samsung GearVR[3] and a Samsung S8 cellphone [4], while the server is implemented in a standard computer

[3]http://www.samsung.com/global/galaxy/gear-vr/
[4]http://www.samsung.com/global/galaxy/galaxy-s8/

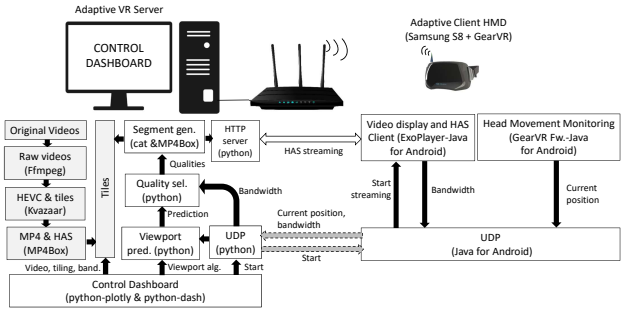

Fig. 2. Experimental demonstrator block diagram. Below the client and the server, the different software blocks are shown. The blocks in gray in the server show the preprocessing functionality.

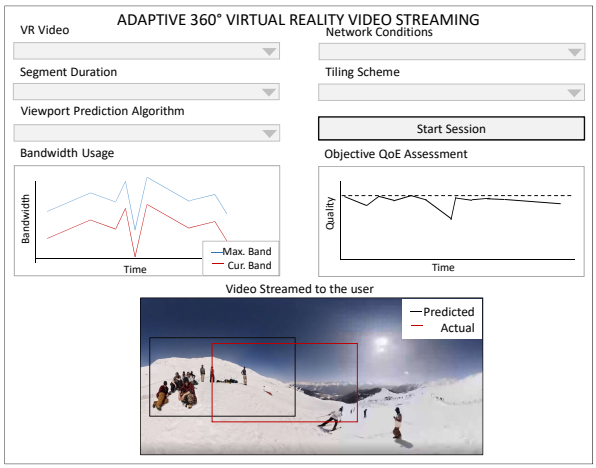

Fig. 3. Dashboard structure

(Dell Latitude 5580[5]). Also in Figure 2, the main software blocks are presented. As described in the previous Section, we define our approach as an adaptive tiling server-based method. Thus, most of the functionality is concentrated in the server. The client takes charge of monitoring bandwidth and head movement and of the HAS video display. Two type of communications are used. Bandwidth and head movement are transmitted using UDP, in order to minimize the network overhead. The video stream is transmitted using the HAS protocol, controlled by the player in the client (the Java based Exoplayer).

On the server, the videos were first pre-processed making use of the well-known video processing tools ffmpeg[6], Kvazaar[7] and MP4Box[8] (gray blocks in Figure 2). Ffmpeg extracts the raw video of the original content. The Kvazaar encoder is in charge of tiling the content and encoding to H.265/HEVC [10]. HEVC was selected due to fact that it includes tiling among its features [11]. This action is performed for each of the quality levels wanted. Finally, the MP4Box is employed to pack the HEVC streams into mp4 containers and to prepare the videos to be streamed using HAS [2]. During the streaming, the server predicts the viewport position for the next segment (based on the received position from the client, using a UDP channel) and assigns the qualities to the tiles. Once decided, it merges the different tiles, and uses the MP4Box to pack the resulting stream on an MP4 container for its transmission. Except for the video processing software packages, the server is programmed in python, due to its flexibility.

The client functionality was implemented in Java using Android Studio[9]. In order to monitor the head movement, the GearVR Framework was used[10]. It provides a position of the head within the sphere that can be mapped into the 2D projection of the video and takes care of all the VR video related characteristics such as rendering, distortions on the image to compensate for the lenses, etc. The GearVR framework was also used in order to project the 2D video back onto a sphere, which can be assigned to a mediaplayer for displaying. As media player we employed the Exoplayer[11]. The reason to choose the Exoplayer instead of the inbuilt Android's mediaplayer is two-folded. First, it allows seamless streaming of HAS-based videos, because in addition to playing the segments, it manages the HTTP requests and buffering. Second, it provides an estimation of the bandwidth required for the streaming, based on the transmission of the current segment over HTTP (retransmissions and file size).

A dashboard (Figure 3) to change the characteristics (viewport algorithm and tiling scheme), conditions (network) and content (videos) during the streaming was implemented using the dash package of plotly-python[12]. In addition, the dashboard shows the real-time performance by means of three representations. First, it provides a quantitative measure of the bandwidth optimization by plotting the employed bandwidth compared to the necessary one if the whole 360° video were streamed at the highest quality. Second, it provides a frame-to-frame objective QoE assessment applying SSIM [12] between the area of the video covered by the actual viewport and the same area of the video at the highest quality. This gives a quantitative measure of the effects of the inaccuracies of the viewport prediction algorithms and the quality selection scheme. Finally, the dashboard offers a visualization of the segment being streamed to the HMD with overlapped prediction and actual viewports (sensed at the beginning of each segment)., which provides a qualitative assessment of the effects of the viewport algorithm. In addition, it shows the strengths of the approach not only to the current user but also to other visitors.

IV. Demonstrator Scenario

The set of characteristics shown in Table I are used for the demonstration. They provide comprehensive and broad evaluation of the capabilities of our approach. Two videos, already analysed by the head movement dataset presented

[5]http://www.dell.com/dm/business/p/latitude-15-5580-laptop/pd
[6]https://www.ffmpeg.org/
[7]http://ultravideo.cs.tut.fi/#encoder
[8]https://gpac.wp.imt.fr/mp4box/
[9]https://developer.android.com/studio/
[10]http://www.gearvrf.org/
[11]https://github.com/google/ExoPlayer
[12]https://plot.ly/products/dash/

TABLE I
VR CONTENT, CHARACTERISTICS AND CONDITIONS TO BE SHOWN IN THE DEMO.

Videos	Segment	Tiles	Qual.	Viewport
1. Freestyle	5s	None	8Mbps	None
Skiing	1 s	12x4	2Mbps	Speed
2. Google	500 ms	1-4-1	500Kbps	Probability
Spotlight-HELP				

by [13], namely "Freestyle Skiing" and "Google Spotlight-HELP", are to be employed. These videos present different characteristics in terms of spatial and temporal complexity, making their analysis complementary. To show the effect of the segment's length, we have chosen three different values (5s, 1s and 500ms). The purpose of the tiling is to compare not only different types of schemes but also the no-tiling approach (*i.e.*, standard adaptive streaming). This is shown by means of three tiling options: no tiling, 12x4 and 1-4-1. The last option distributes the tiles according to the structure of the sphere. This means that the regions close to the poles are not split, while around the equator, the space is uniformly divided into four tiles. Regarding the qualities, three quality levels are shown (8Mbps, 2Mbps and 500kbps). Finally, our server-based approach allows the implementation and usage of different viewport algorithms. For this demonstrator we aim to compare three algorithms. First, we show the effects of a non-predictive approach. This algorithm assumes that the user will look exactly at the same position for the next interval. Thus, it selects the tile qualities following the previously used field of view. Counter-intuitively, this type of adaptive algorithm provides accurate results for users that travel through the sphere at very low speed or for very dynamic video content (if the content changes very fast, the user tends to focus on following the content from the same position instead of exploring the space). Our second algorithm predicts the next location based on the current position and the estimated speed at which the user will move. In order to predict the speed, the algorithm assumes that the user will have the same speed trend as in the previous two segments and based on those it extrapolates the current value. For the final viewport prediction algorithm, we take a probabilistic approach. It generates a heat-map of likely locations within the video scene where the user is most likely to be watching at next, based on previous users' locations records. This is an approach with inspirations on the Gravitational algorithm currently used by Facebook. In our case, we generated such heat-map thanks to the head movement dataset presented by Wu et al. [13].

V. CONCLUSION

In this demonstrator we present an adaptive tile-based VR platform which allows for the users to explore, experience and compare the working principles of adaptive tile-based VR video streaming. Through different tiling approaches, bandwidth conditions and viewport algorithms, the user will experience first-hand the effects that each optimization has on the perception of the adaptive tile-based VR video streaming service. In addition to the experience of the users, the platform provides real-time qualitative evaluations by means of bandwidth usage and objective Quality of Experience (QoE) measurements. These evaluations enable the creation of a mapping of the effects of VR streaming service optimizations onto video QoE. In addition, this platform allows for researchers to benchmark their optimization solutions in a common evaluation playground.

ACKNOWLEDGMENT

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).

REFERENCES

[1] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update, 20162021," Cisco Systems, Tech. Rep., 2017.

[2] R. Irajá Tavares da Costa Filho, M. Torres Vega, M. Caggiani Luizelli, J. van der Hooft, S. Petrangeli, T. Wauters, F. De Turck, and L. Paschoal Gaspary, "Predicting the Performance of Virtual Reality Video Streaming in Mobile Networks," in *Proceedings of the 2018 ACM Multimedia Systems*, 2018.

[3] G. Dimopoulos, I. Leontiadis, P. Barlet-Ros, and K. Papagiannaki, "Measuring video qoe from encrypted traffic," in *Proceedings of the 2016 Internet Measurement Conference*, ser. IMC '16. New York, NY, USA: ACM, 2016, pp. 513–526. [Online]. Available: http://doi.acm.org/10.1145/2987443.2987459

[4] I. Sodagar, "The mpeg-dash standard for multimedia streaming over the internet," *IEEE Multimedia*, vol. 18, no. 4, pp. 62–67, 2011.

[5] C. Concolato, J. L. Feuvre, F. Denoual, E. Nassor, N. Ouedraogo, and J. Taquet, "Adaptive streaming of hevc tiled videos using mpeg-dash," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. PP, no. 99, pp. 1–1, 2017.

[6] S. Petrangeli, V. Swaminathan, M. Hosseini, and F. De Turck, "An http/2-based adaptive streaming framework for 360 virtual reality videos," in *Proceedings of the 2017 ACM on Multimedia Conference*, ser. MM '17. New York, NY, USA: ACM, 2017, pp. 306–314. [Online]. Available: http://doi.acm.org/10.1145/3123266.3123453

[7] O. A. Niamut, E. Thomas, L. D'Acunto, C. Concolato, F. Denoual, and S. Y. Lim, "Mpeg dash srd: Spatial relationship description," in *Proceedings of the 7th International Conference on Multimedia Systems*, ser. MMSys '16. New York, NY, USA: ACM, 2016, pp. 5:1–5:8. [Online]. Available: http://doi.acm.org/10.1145/2910017.2910606

[8] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, ser. ATC '16. New York, NY, USA: ACM, 2016, pp. 1–6. [Online]. Available: http://doi.acm.org/10.1145/2980055.2980056

[9] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo, "360probdash: Improving qoe of 360 video streaming using tile-based http adaptive streaming," in *Proceedings of the 2017 ACM on Multimedia Conference*, ser. MM '17. New York, NY, USA: ACM, 2017, pp. 315–323. [Online]. Available: http://doi.acm.org/10.1145/3123266.3123291

[10] M. Viitanen, A. Koivula, A. Lemmetti, J. Vanne, and T. D. Hmlinen, "Kvazaar hevc encoder for efficient intra coding," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2015, pp. 1662–1665.

[11] K. Misra, A. Segall, M. Horowitz, S. Xu, A. Fuldseth, and M. Zhou, "An overview of tiles in hevc," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 6, pp. 969–977, Dec 2013.

[12] P. Orosz, T. Skopk, Z. Nagy, P. Varga, and L. Gyimthi, "A case study on correlating video qos and qoe," in *2014 IEEE Network Operations and Management Symposium (NOMS)*, May 2014, pp. 1–5.

[13] C. Wu, Z. Tan, Z. Wang, and S. Yang, "A dataset for exploring user behaviors in vr spherical video streaming," in *Proceedings of the 8th ACM on Multimedia Systems Conference*, ser. MMSys'17. New York, NY, USA: ACM, 2017, pp. 193–198. [Online]. Available: http://doi.acm.org/10.1145/3083187.3083210