Exploring New York in 8K - An Adaptive Tile-based Virtual Reality Video Streaming Experience

Maria Torres Vega, Jeroen van der Hooft, Joris Heyse, Femke De Backere, Tim Wauters, Filip De Turck
Ghent University - imec
Ghent, Belgium
name.surname@ugent.be

Stefano Petrangeli
Adobe Research
San Jose, USA
spetrange@adobe.com

ABSTRACT

Adapting and tiling the streaming of virtual reality (VR) video content has the potential to reduce the ultra-high bandwidth requirements of this type of multimedia services. Towards that goal, the optimization of a number of aspects is currently actively being researched. Novel rate adaptation heuristics, sophisticated viewport prediction algorithms and streaming protocol optimizations have proven their value to improve certain aspect of the VR streaming chain. However, the interplay between all these different optimizations as well as their tradeoff has not yet been explored in an experimental playground. The purpose of this demonstrator is to provide a full end-to-end adaptive tile-based VR video streaming system where each of the optimization aspects can be tuned with and their effect illustrated on-site.

CCS CONCEPTS

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

KEYWORDS

Virtual Reality, Adaptive Video Streaming, MPEG-DASH, Viewport Prediction

1 INTRODUCTION

Adaptive tile-based streaming provides the solution to deal with the ultra-high bandwidth requirements of the streaming of virtual reality (VR) video content. This could potentially facilitate the usage of lightweight clients and to cope with the ever-changing conditions of wireless networks [**?**]. Towards this goal, optimizations for a plethora of aspects of the streaming chain are currently under research and development.

Novel sophisticated viewport prediction algorithms for prefetching of the content that the user will be observing next [**? ?**], bandwidth and viewport aware rate adaptation algorithms (such as the approaches of Hosseini et al. [**?**] or van der Hooft et al. [**?**]), or the application of HTTP/2 to speed up the request process (Petrangeli et al. [**?**]) are already facilitating the streaming of VR videos over the network. However, while current methods provide suited solutions for one or two of the transmission aspects, the question of the interplay among all the components and how they affect one another in real streaming environments remains unanswered.

In this work, we present an end-to-end adaptive tile-based VR video streaming platform in which the interplay of the different optimization components is evaluated on-site. In addition to implementing optimizations both from the state-of-the-art and from our own previous work, this platform provides a proactive quality reassignment feature. It allows for real-time changes on the priorities and qualities of the content according to the most up-to-date predicted position, while the streaming is taking place. The approach will be showcased on an omnidirectional high motion video of the city of New York in an 8K resolution[1]. In the reminder of this paper, the different optimizations of which the end-to-end approach is composed are first presented. Then, implementation details of the testbed as well as video characteristics are provided. It finalizes with some conclusions.

2 APPROACH

Figure **??** presents the block diagram of the end-to-end adaptive VR video streaming approach. The 360° content is encoded at different quality levels, segmented and tiled at the server side. During the streaming, the client requests the tiles from the corresponding video segment at a decided quality. The quality decision is taken by the client's rate adaptation heuristic according to the video's bitrate, the predicted viewport location, the video's bitrate and the buffer size. During the streaming, the quality reassignment block allows for real-time reordering of the remaining tiles to stream according to the most up-to-date predicted user's position. In the remainder of this Section, each of the chosen tasks and optimizations are described.

2.1 VR Video Tiling

Tile-based encoding for VR video is often achieved through the HEVC standard, because it is one of the few encoders that allows tiling[2]. Others are for example, VP9. Even if there are currently

MMSys '19, June 18–21, 2019, Amherst, MA, USA

ACM ISBN 978-1-4503-6297-9/19/06.
https://doi.org/10.1145/3304109.3323831

[1] https://www.youtube.com/watch?v=2Lq86MKesG4
[2] https://www.divx.com/en/software/technologies/hevc/

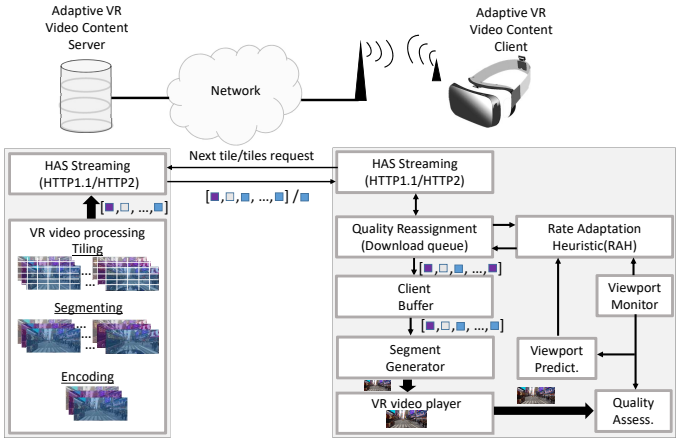

Figure 1: End-to-end block diagram for Virtual Reality Video streaming.

several mapping strategies to map the sphere onto a 2D projection (such as cubic, pyramid and dodecahedron mappings [**?**]), the equirectangular mapping is most commonly used, in which the sphere is mapped onto a rectangle of $m \times n$ tiles. This approach allows for more fine-grained decision-making by the client, using the available throughput where it is needed. However, as previously mentioned, tile-based encoding introduces an encoding overhead. Using different tiling schemes, we will evaluate this overhead and the resulting video quality under similar network conditions. Similar to related work, the Constant Rate Factor (CRF) rate control will be used to differ between visual quality, and thus resulting bitrates for different quality representations.

2.2 HAS Streaming Protocol: HTTP/1.1 vs HTTP/2

Most HTTP Adaptive Streaming (HAS) solutions use HTTP/1.1 with request-response transactions to retrieve the required resources, buffering fetched video segments and playing them out in linear order. In the most common case, one GET request is needed in order to retrieve the next temporal part of the video; therefore, this approach is feasible as long as the duration of the video segments is of a higher order than the latency within the network. However, the introduction of the additional tiling dimension requires multiple resources to be retrieved, in order to play out one temporal video segment (each tile requires its own GET request). This could potentially impede the overall throughput between client and server, resulting in a lower video quality.

There are two possible solutions. First, an approach based on multiple persistent TCP connections can be used. Most browsers today (including Chrome, Firefox and Safari) use up to six parallel TCP connections in order to reduce the page load time, fetching required resources in parallel. Second, the need for multiple GET connections can be eliminated by using the server push feature of the HTTP/2 protocol [**?**]. To enable this feature for tile-based VR environments, a custom request handler can be employed at server-side, allowing the client to define a list of quality levels for each of the tiles to retrieve. However, it can lead to inaccurate quality distributions in cases where the viewport prediction error is too high. This demonstrator will explore the tradeoffs among these three streaming variants, namely HTTP/1.1, HTTP/1.1 with parallel connections, and HTTP/2.

2.3 Viewport Prediction Algorithms

Viewport prediction is one of the most important aspects of tile-based HAS. If the future position is accurately predicted, the client can compensate for the user's movements and download relevant parts of the video at higher quality. In the state-of-the-art, a distinction is made between approaches that require knowledge on the content to predict the next position of the user (content-aware solutions) and the ones that make predictions based only on previous positions or trajectories (content-agnostic solutions). Most content-aware approaches require preprocessing of the video content and thus, are currently too computationally intensive for real-time predictions and unfit for VR video streaming of live content, or demonstrations (*e.g.*, Pano2vid [**?**]). Content-agnostic solutions, on the other hand, can be more lightweight as they only require the Head-Mounted Display (HMD) sensor data and are better suited for real-life demonstrations. Thus, for this work we will focus on content-agnostic approaches.

The current trend in content agnostic viewport prediction, is to make predictions based on the previous trajectory of the user. This is the case of two-dimensional linear extrapolation of the user's movement proposed by Petrangeli et al. [**?**]. In [**?**], we proposed a three-dimensional extension of this approach. We modeled the user movement as a trajectory on the surface of the unit sphere, rather than a 2D trajectory on the equirectangular projection. In addition, as the user's movement is largely volatile, our proposal was to predict the next viewport not by completing the full trajectory, but rather a fraction thereof. This approach aims at reducing the impact of users' movement, thus resulting in a lower prediction error. In addition to volatility of movement, in a lot of cases, the user moves only during certain sections of the video. This greatly difficults the use of machine learning techniques, because the collected user data to use for training the algorithms is biased towards lack of movement. To cope with this, in [**?**] we proposed to a machine learning algorithm which first predicts if the user will move at all to then provide a prediction of the direction. We based our algorithm on reinforcement learning methods, in particular Contextual Bandit Learning. This demonstration work will put to test these three approaches. For the remainder of this paper, the approach of Petrangeli et al. will be dubbed as 2D-Full, the van der Hooft et al. as 3D-Partial and the algorithm of Heyse et al. as CBL-Weighted.

2.4 Rate Adaptation Heuristic

Adding a spatial dimension increases the complexity of the rate adaptation heuristic. In this demonstrator, we present and test two rate adaptation heuristics. The first one, dubbed Uniform Viewport Quality (UVQ) strongly focuses on the (predicted) viewport location, uniformly increasing the quality of each tile whose center is within the viewport. The second heuristic uses a Center Tile First (CTF) approach, which increases the quality of the tiles to the highest quality representation, in the order of decreasing distance between the center of the tile and the viewport location.

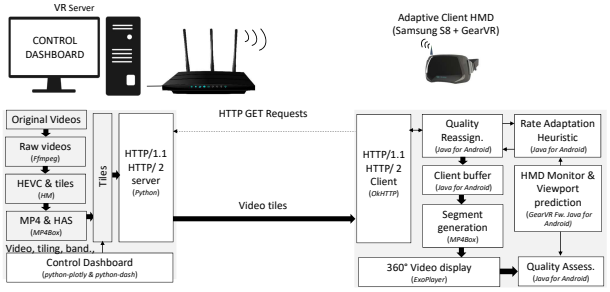

Figure 2: Software block diagram of the adaptive VR video streaming demo. For each of the blocks, the specific library used can be found in between parenthesis.

2.5 Quality Reassignment

The rate adaptation heuristic makes per-segment decisions based on the predictions of the viewport, the network conditions at the time of the calculation, the buffer status, etc. Even if this approach has significant advantages, it is extremely sensitive to changes on the initial conditions (user's viewport position, network conditions, etc.), especially for larger segment durations. For example, the user could change its trajectory, or the download of a tile might take longer than expected due to congestion in the network. In addition, the shorter the time between the viewport prediction and the playout of the segment, the more accurate the prediction will be.

To cope with the real-time changes on conditions, this demonstrator incorporates a feedback loop. As soon as a segment is removed from the buffer and playout starts, the client starts buffering the next segment. Therefore, the viewport prediction algorithm predicts the expected viewport coordinates, based on the client's location history. Subsequently, this prediction is fed to the rate adaptation heuristic, which makes a decision on the quality of each of the tiles based on the viewport coordinates, the observed throughput and the available quality representations. Its calculations and decisions are forwarded to the download queue, which will start retrieving the tiles that are furthest away from the currently predicted viewport location. This allows the client to receive the tiles which are expected to have the least impact on the final viewport. Furthermore, it reduces the chances of freezing (all tiles need to be downloaded before playout of the next segments can start). While the downloading takes place, the viewport coordinates of the HMD are regularly updated and new predictions are made. Using the new viewport location, along with information of the download queue, the heuristic reassigns the bandwidth budget among all the tiles which have not yet been downloaded, and forces an update of the data in the download queue. Thus, initial decisions can be overruled by the client, in favor of tiles which are expected to be of higher importance than previously estimated.

2.6 Quality Assessment

Related work on quality assessment of tile-based VR streaming has focused on the quality of the viewport center only. This allows to measure how long the user spent on each quality layer. However, this does not take into account the fact that the users' eyes also move within the viewport. To account for this eye movement, in [**?**], we introduced a quality metric based on a probabilistic approach.

Based on the experimental users' gaze data presented by Rai et al. in [**?**], we modeled the gaze as a probability distribution which weights the effect of each of the tiles within the viewport according to the influence on the overall quality. In [**?**] we presented the evaluation of our metric relative to objective metrics such as PSNR and SSIM. In this demonstrator, we will evaluate the quality with this metric.

3 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 **??**). The client consists of a Samsung GearVR[3] and a Samsung S8 cellphone[4], while the server is implemented in a standard computer (Dell Latitude 5580[5]). Also in Figure **??**, the main software blocks are presented.

On the server side, the videos were first preprocessed making use of the well-known video processing tools ffmpeg[6], HM[7] and MP4Box[8] (gray blocks in Figure **??**). Ffmpeg extracts the raw video of the original content. The HM encoder is in charge of tiling the content and encoding to H.265/HEVC [**?**], which allows for tiling the content [**?**]. The encoding was performed for each quality representation. Finally, MP4Box packs the HEVC streams into mp4 containers and prepares the video tiles to be streamed using HAS. The HTTP server is implemented in Python using the Hyper library [9], which allows to stream using both HTTP/1.1 and HTTP/2.

The client functionality was implemented in Java using Android Studio[10]. In order to monitor the head movement, the GearVR Framework was used[11]. 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[12] due to its demonstrated performance to display omnidirectional video [**?**]. However, given the fact that the current Exoplayer version cannot deal with tiled content, the tiles need to be merged into a segment before displaying it. For this, we again turned to MP4Box, which is also compatible with Android environments. Finally, the client side of the HTTP protocol was implemented using the OkHTTP library[13], which allows to stream both in HTTP/1.1 and HTTP/2.

4 DEMONSTRATION SCENARIO

In order to show the tradeoffs, advantages and disadvantages of the optimizations imposed by our adaptive tile based streaming

[3] http://www.samsung.com/global/galaxy/gear-vr/
[4] http://www.samsung.com/global/galaxy/galaxy-s8/
[5] http://www.dell.com/dm/business/p/latitude-15-5580-laptop/pd
[6] https://www.ffmpeg.org/
[7] https://hevc.hhi.fraunhofer.de/HM-doc/
[8] https://gpac.wp.imt.fr/mp4box/
[9] https://hyper.readthedocs.io/en/latest/
[10] https://developer.android.com/studio/
[11] http://www.gearvrf.org/
[12] https://github.com/google/ExoPlayer
[13] https://square.github.io/okhttp/

Table 1: Obtained bitrates [Mb/s] for the New York video at the different tiling schemes.

CRF	1 × 1	4 × 4	6 × 4
25	43.38 ± 13.07	43.34 ± 13.15	43.64 ± 13.53
35	9.32 ± 2.57	9.39 ± 2.72	9, 56 ± 2.61
45	2.98 ± 0.86	3.13 ± 0.88	3, 18 ± 0.89

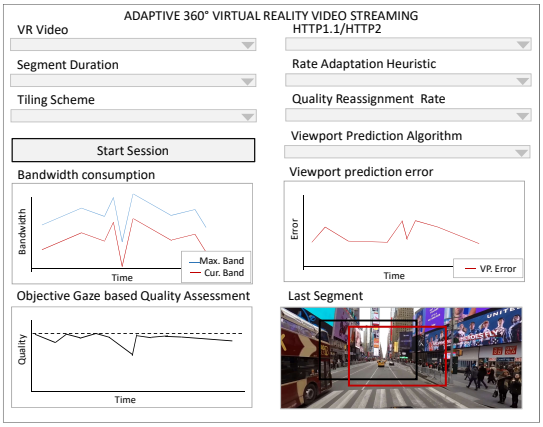

Figure 3: VR server Dashboard.

approach we selected the 8K omnidirectional video "New York tour" (available online on You tube). As explained in the demonstrator Section, the video needs to be encoded, segmented and tiled to make it available to the adaptive VR streaming client. In order to explore the strengths of our optimizations, we selected three different encoding CRFs (25, 35, 45) using a Group of Pictures (GOP) of 32 frames. These CRFs results in bitrates of around 43Mb/s for a CRF of 25, 10Mb/s for a CRF of 35 and 2.5Mb/s for a CRF of 45. Subsequently, the video streams were split into 1, 065s segments. This duration corresponds to 1 Group Of Pictures (GOP). Further, these segments were subjected to three different tiling schemes, namely 1×1 (or no tiling), 4×4 and 6×4. During the demonstration at the conference, the users will be able to explore New York from a 360° perspective, while benefitting from the different optimizations of the streaming.

A dashboard (Figure **??**) 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[14]. In addition, the dashboard shows the real-time performance by means of four 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 shows a view of the error in the prediction in radians as predicted by the client. This view provides the chance to observe the comparative performance of each of the presented viewport prediction algorithms. Third, it provides a QoE assessment using our own probability based assessment metric [**?**]. 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 just streamed to the HMD with overlapping 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.

[14]https://plot.ly/products/dash/

5 CONCLUSION

Optimizations on the prediction of the field of view, on how to adapt the quality of the stream content and to improve the transmission protocols are making adaptive tile-based VR video streaming a well-suited solution to cope with the bandwidth requirements of omnidirectional content. Understanding the interplay of these optimizations on the overall performance of the system and the perceived quality of the user is thus fundamental. This demo presents an end-to-end adaptive tile-based virtual reality video streaming system in which all elements of the chain can be adapted, different algorithms input and their characteristics changed.

ACKNOWLEDGMENTS

Maria Torres Vega is funded by grant of the Research Foundation - Flanders (FWO). Jeroen van der Hooft is funded by the Agency for Innovation by Science and Technology in Flanders (VLAIO). 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).