Film Editing: New Levers to Improve VR Streaming

Savino Dambra, Giuseppe Samela, Lucile Sassatelli, Romaric Pighetti, Ramon Aparicio-Pardo and Anne-Marie Pinna-Déry*
Université Côte d'Azur, CNRS, I3S
Sophia Antipolis, France

ABSTRACT

Streaming Virtual Reality (VR), even under the mere form of 360° videos, is much more complex than for regular videos because to lower the required rates, the transmission decisions must take the user's head position into account. The way the user exploits her/his freedom is therefore crucial for the network load. In turn, the way the user moves depends on the video content itself. VR is however a whole new medium, for which the film-making language does not exist yet, its "grammar" only being invented. We present a strongly inter-disciplinary approach to improve the streaming of 360° videos: designing high-level content manipulations (film editing) to limit and even control the user's motion in order to consume less bandwidth while maintaining the user's experience. We build an MPEG DASH-SRD player for Android and the Samsung Gear VR, featuring FoV-based quality decision and a replacement strategy to allow the tiles' buffers to build up while keeping their state up-to-date with the current FoV as much as bandwidth allows. The editing strategies we design have been integrated within the player, and the streaming module has been extended to benefit from the editing. Two sets of user experiments enabled to show that editing indeed impacts head velocity (reduction of up to 30%), consumed bandwidth (reduction of up to 25%) and subjective assessment. User's attention driving tools from other communities can hence be designed in order to improve streaming. We believe this innovative work opens up the path to a whole new field of possibilities in defining degrees of freedom to be wielded for VR streaming optimization.

CCS CONCEPTS

• **Human-centered computing** → **User studies**; *Virtual reality*;
• **Networks** → *Network experimentation*;

KEYWORDS

360 video, streaming, film editing, tiling, head motion

*S. Dambra, G. Samela and R. Pighetti are now with Eurecom, Politecnico di Bari and Francelabs, respectively. Corresponding author: sassatelli@i3s.unice.fr

MMSys'18, June 12–15, 2018, Amsterdam, Netherlands

ACM ISBN 978-1-4503-5192-8/18/06…$15.00
https://doi.org/10.1145/3204949.3204962

1 INTRODUCTION

Virtual Reality (VR) has taken off in the last two years thanks to the democratization of affordable Head-Mounted Displays (HMDs). According to [11], 12.4 million more HMD sales are forecast by the end of 2018, steadily increasing to a yearly 40-million in 2022, and VR/AR (Augmented Reality) traffic will increase 20-fold by 2021 [8]. In this article, the type of VR we focus on are 360° videos (shot with an omnidirectional camera). Two main hurdles are however currently in the way of the full rise of 360° videos. The first one is the ability to stream these videos, the second is the design and creation of these videos for best user's experience.

The streaming delivery over the Internet is a true challenge because the bit rates entailed by 360° videos (even H.265-compressed) are much higher than for conventional videos (about 28Mbps [3, 19, 35] with limited quality, up to 5.2 Gbps for an artifact-free VR experience with sight only [4]). Such required bandwidth lead to offer the download option to avoid interruptions and low definitions. To cope with the discrepancy between the required video rate for best quality and the available network bandwidth, a simple principle is to send the non-visible part of the sphere with lower quality [10, 33] (see Fig. 1). The question is then how to allocate the bandwidth resource by choosing which quality to send for each region of the sphere. The quality decisions must strive to maintain a high-quality in the user's Field of View (FoV). However, a main component which has allowed streaming of good-quality videos over the best-effort Internet has been the playback buffer at the client. This buffer allows to absorb the bandwidth variations and prevents playback interruptions if the network drop is not too long. For 360° videos as well, this feature is crucial for streaming. An important problem is then how to reconcile the contradictory strategies of allowing the client to buffer several seconds of video to absorb the network instability, while at the same time keeping the qualities of the buffered segments for each region up-to-date with the moving user's FoV. Existing strategies to tackle this problem are (i) keeping the buffers short (e.g., 1 second [10]), with or without (ii) gaze prediction [21, 23, 25], while (iii) allowing buffers to build up requires to enable replacements to maintain freshness w.r.t. the FoV [47]. In the 2017 Facebook technical conference [21], the combination of gaze prediction and segment replacements has been briefly presented in slides, and is planned to be introduced in the FB360 delivery service by 2018. While 360° videos are heavier to give the user the ability to move, the amount of replacements directly depends on the quantity of motion, and incurs bandwidth overhead. The way the user exploits her/his freedom is therefore crucial for the network load.

In turn, the way the user moves depends on the content itself, and varies widely. As shared by the former film director of the Arte TV channel in [40], "Two different watching behaviors seem to

emerge: either a spectator will be focused on one particular thing [...] or he/she will fall victim to the FOMO (fear of missing out) effect, and that's very big in VR, looking everywhere and focusing on nothing." VR is a whole new medium whose immersive capacity provides an unprecedented feeling of presence in the scene. VR therefore makes for a powerful instrument in a broad range of goals and genres (documentaries, storytelling, journalism, etc.). VR filmmaking is still a much unexplored domain, wide open for innovation as presented in [6]. The cinematic language for VR does not exist yet, "its grammar only being invented" [6, 45]. In the present work, we introduce high-level manipulations of the content, and specifically focus on editing which is, e.g., transition from one sequence-shot scene to the next. Editing is important for the user's experience, not to feel thrown off balance when entering a new scene (environment) [6], triggering discomfort and fast head motion.

In this article, we take a strongly inter-disciplinary approach to devise editing strategies for 360° videos, and investigate their impact on streaming performance both for network-level metrics and user-level metrics. Our approach is therefore at the crossroads of networking, Human Computer Interfaces (HCI) and cinema. Our proof-of-concept proves that high-level content manipulations indeed impact the streaming performance.

Contributions:

• We create a complete testbed implementing an advanced streaming strategy composed of MPEG DASH-SRD tiled format with buffering, replacements and FoV-based quality decision for the Samsung Gear VR on Android.

• We design editing strategies aimed at reducing the user's motion, hence the consumed bandwidth, and potentially increase their Quality of Experience (QoE). We incorporate the editing within our streaming application, so as to make the streaming process benefit from the editing (in particular, from the knowledge of future head position).

• We carry out two rounds of User Experiments (UX) to compare the streaming with different editings in terms of head motion, bandwidth and subjective users' assessment. We perform hypothesis testing and show that simple editing strategies can reduce head motion (mean velocity) by 30%, consumed bandwidth by 25% while maintaining high quality in the FoV.

Our testbed, obtained dataset and contents are made available and described in App. A.

Our approach considers neither (FoV-based) encoding nor motion prediction, but instead high-level content manipulations to limit or control the user's head motion. It is therefore entirely complementary and compatible with the former.

Thanks to an inter-disciplinary approach, we uncover a whole new field of possibilities in defining degrees of freedom to be wielded in optimizations of VR streaming. We believe the proof-of-concept presented in this paper can open up the path to important improvements in the online distribution of VR.

Sec. 2 positions our work with respect to existing relevant works. Sec. 3 presents the design of the editing strategies. Sec. 4 details the testbed we produced. Sec. 5 details the design of the UX, while Sec. 6 shows and explains the results. Sec. 7 provides a discussion on the impact of the approach, and Sec. 8 concludes the article.

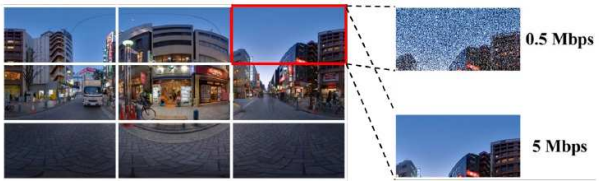

Figure 1: A tiled 360° scene (equirectangular projection) and a tile encoded into two different qualities.

2 RELATED WORKS

We structure this section as follows. First, we present the general trends to lower the required bandwidth, by making transmission decisions based on the FoV (tiling, viewport-based encoding). As the efficacy of these solutions rely on the knowledge of the future FoV, we then overview current proposals for FoV prediction. Our approach to consider motion is different and complementary: instead of predicting, we seek to control the motion, to limit the randomness and exploit the a priori knowledge for the streaming decisions. Finally we review approaches to drive user's attention.

Two approaches allow to consider spatially-heterogeneous encoding, whose goal is to save bandwidth by sending in higher quality regions in the current FoV, and those outside the FoV with degraded quality. The Spatial Relationship Description (SRD) extension to the MPEG DASH standard considers the 360° video as made of independent objects over time (segments) and space (tiles) [33]. It splits the sphere into pre-defined tiles and encodes them into predefined qualities (see Fig. 1). Tiling however reduces compression efficiency, which has been addressed in [46]. It also entails numerous HTTP requests, addressed in [36, 37] by considering HTTP/2. To keep fresh w.r.t. to the current FoV, the buffers are usually kept shallow (e.g., [10]). This is addressed in [44] where layered video coding is employed to improve quality in the FoV when bandwidth allows. This is the approach we have considered with replacement instead, in order to keep H.264/AVC encoding. Alternatively to tiling, regional projection-based representations are designed in [10, 20, 47]. The sphere is split into regions projected onto cubes (or pyramids), the central view is optimally sampled and the surrounding pixels are sub-sampled. The resulting spherical content is therefore sent as a single entity in each segment. For example Oculus uses the Offset cubic projection, considering 22 offset cube orientations and 4 quality levels, hence generating 88 versions of each video, with 1s-long segments [47]. It is also shown that a better quality than equirectangular projection can be achieved with less than 50% of the pixels, but only if the discrepancy between the actual FoV center and the selected orientation is maintained under 40° by the adaptation algorithm. Oculus makes its own MPEG DASH extension to support this content shaping. The adaptation logic is conservative to avoid stalls, and fetches low quality segments upon FoV change. It is also emphasized that user head movements affect the number of wasted segments, and heavily depend on the video being watched. The high storage costs incurred by this

approach have been tackled in [9]. To remedy the storage problem, transcoding can be used [2, 39]. In [2] in particular, full transcoding of the requested viewports for each user is designed to improve the perceived quality. Tiles in 4K and 1080p resolutions are combined so that the transcoding is real-time. To experiment editing, we needed a sound yet simple enough testbed. For this reason, we opted for MPEG DASH-SRD with H.264/AVC. Also, the vast majority of above works have been published after we made the testbed, in the first semester of 2017.

All the above approaches require to know the future FoV position to work at their best. Saliency maps are computer vision tools to predict the probability that each region attracts the human gaze (see, e.g., [12]). Leveraging this concept to predict the future FoV position, some recent works have in particular employed deep learning approaches. In [23], Long Short Term Memory (LSTM) networks are used to predict the future orientation and tiles in the FoV. In [20], based on a map extracted from the Facebook users' data, a so-called gravitational model is assumed to predict the next position from the current trajectory, speed, and attractors known from the heat map.

On the other hand, the human attention is to be at the center of the media experience. For example, it has been highlighted in [30] that the influence of binocular depth cue on visual attention in 3D-TV is highly dependent on the content itself and not only on the presence or strength of the stereoscopic disparity. It is also a crucial aspect for the VR experience, which is starting to be investigated. In [18], a vibrotactile HMD is introduced. Electromechanical tactors on the fronthead are instrumented to allow a natural, precise and rapid localization of the target in an immersive VR setup. The autopilot feature of the 360fly 4K 360° camera allows to automatically pan to the area of footage that has the highest degree of motion [1], but is hence destined to viewing outside a headset.

In [7], a plan of future work lists two methods to shift the user's gaze to the required region, for collaborative narrative tasks in VR. One technique envisioned is the introduction of a firefly in the FoV flying in the direction of the pre-set target region, until the user moves his head. The second is rotating the sphere to reposition the user, inspiring from film techniques but expected to be disorienting for the user. This is hence the idea the closest to ours, but not aimed at streaming, nor carried out yet. As detailed in the next section, we choose other types of editing for a cinematic setup, which were devised from both recent knowledge gained in VR gaming, and film editing.

Film editing consists in selecting shots from the raw footage and assembling them (see Fig. 2). As film director Preston Sturges stated, "There is a law of natural cutting and [...] this replicates what an audience in a legitimate theater does for itself", that is make appear on the screen what the viewer would watch in the real-world, so that the cut is not perceived by the viewer [43]. Editing is by construction a main tool to control the user's attention. Several types of editing exist [5]. Specifically, Match on action consists in interrupting and resuming a motion, and has inspired Match on attention for VR presented in [6], which we consider as our first strategy whose impact on streaming is tested. From circa 2006, editing has entered the so-called "post-classical" phase, where the Average Shot Length (ASL) has drastically decreased, dropping from 8-11s in the 1930's to 3-4s today [13]. It is in particular aimed at changing scene fast (for example from a room to a car) without showing the intermediate steps and letting the brain fill in. In this article, this is what we leverage to control motion while avoiding sickness. Fast cutting was first aimed at young generations (also called "MTV" editing) and first used in major action movies. It then got generalized and is now widely used (TV shows, series, etc.) so that the general population is used to get their attention led and maintained by fast cuts. Avoiding cuts in VR cinema (see, e.g., [38]) may therefore entail boredom and trigger scene exploration in search of action, hence more motion. Sec. 3 details how we leverage film editing to design simple yet new editing strategies to prove that editing impacts streaming performance.

3 DESIGN OF VR EDITING TO EASE STREAMING

We first present the background on content semantics for editing. We then introduce the concept of editing and the tool used. Finally we present the editing designs we make for our proof-of-concept.

3.1 The concept of region of interest

An image is made of different zones which can be weak or strong attractors for human attention. The latter are called Regions of Interest (RoI), or salient points (see, e.g., Fig. 3). High-saliency regions are characterized by both (i) low-level features (related to the primary human visual system) and (ii) high-level features (related to the semantics) (see, e.g., [12]). Examples of (i) include high contrast, motion, small depth (in the foreground), and (ii) can be other human faces, talking faces compared with non-talking faces, cars, animals. The user's motion being heavily driven by the so-defined salient regions, it is therefore crucial to take them into account in our editing design so as to limit the head motion.

3.2 The concept of editing

The editing material in legacy video and VR is sequence-shots. As depicted in Fig. 2, editing in legacy videos is cropping and sorting linearly the shots over time. The concept of editing in VR has been introduced by the Google principal filmmaker in May 2016 [6]. In VR, a 360° sequence-shot can be represented as a circular color stripe; the radial axis is the time. Editing consists not only in sorting the stripes (shots) in time, but also in rotating them around each other to control where the viewer arrives in the next scene, depending on her point of view in the previous scene (see Fig. 3). Controlling such transitions between the scenes means to control how the user feels teleported from one world to the other, and is hence central for QoE and motion limitation.

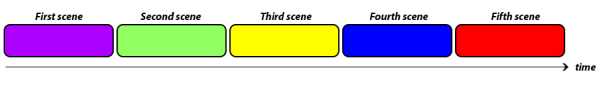

Figure 2: Editing legacy videos: arranging scenes over time.

To implement the editing strategies presented next, we use Adobe Premiere Pro, a timeline-based video editing software. It supports the editing of 360° contents. We employ features such as scene cut,

Figure 3: Left: Region of Interest (RoI). Right: 360° scenes over time, black (white) dot is RoI at the beginning (end) of the scene.

arrangement, and rotation. The *offset* filter tool enables in particular to change the rotation of the scene around the user. Example of the effect of different filter values are provided in Fig. 4.

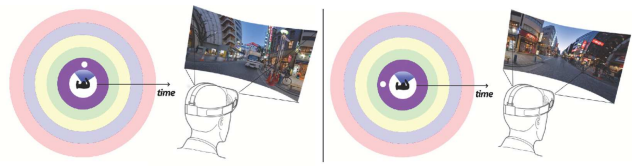

Figure 4: Effect of different values for the offset filter.

3.3 Static editing

The first strategy we introduce is dubbed "Match on attention" in [6] where it has been sketched. We start from the knowledge that, when presented with a new image, the human gaze goes through an exploratory phase scanning fast the environment before settling on a RoI [12]. The hypothesis we make (which turns out reasonable, see Sec. 6.1) is that the same phenomenon occurs when entering a new scene in 360, where rapid head motions are likely if the user had to move to find another RoI. To prevent that, our strategy is to align, as much as possible, the RoIs between the successive scenes. The principle is depicted in Fig. 5. Doing so, if a user is facing a RoI at the end of a scene (which we consider likely because it is a salient region by definition), then he will face another RoI in the next scene, and hence will be less likely to move fast to reposition in the initial gaze exploratory phase. The adjective static reflects the fact that the editing is made at video creation time (the video file needs to be regenerated if one wants to change the editing).

We emphasize that our goal is to prove the concept that editing can impact head motion and consumed bandwidth. To do so, we implement instances of our approach on two specific contents. The problems of how to define more precisely RoI matching (same depth, type of sound, etc.), how to prioritize which RoI to pair and match (when there is more than one RoI in a scene), how to automatize the matching, are out of the scope of this article.

3.4 Dynamic editing

As aforementioned, fast-cutting consists in assembling shots of short durations (e.g., 3.5s on average [13]) one after another to

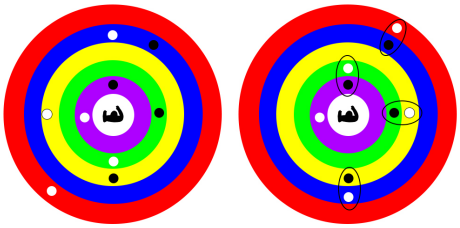

Figure 5: Left: RoI positions are not systematically aligned. Right: RoI are aligned (static editing).

maintain the user's attention. So far in 360° video creation, an entire scene (or even episode [38]) can be made of a single shot without any cut, in order not to disturb the user, who is much more sensitive to video manipulation in VR. Other contents (e.g., [32]) have only slow cuts corresponding to new camera positions. Researches in Human Computer Interfaces (HCI), and specifically virtual HCI, are identifying a number of guidelines in creation of applications for HMDs [41]. Moving the 360° camera is indeed tricky, and a priori constrains main cinematographic techniques such as traveling [45]. However, it has been shown that linear and constant-speed motion is tolerated by the ear-vision system, as well as so-called snap-changes [22], which are fast-cuts in a 360° scene to allow moving in the scene while skipping any non-linear motion that would create sickness when the user does not move, and letting the brain just "fill in the blanks" without the vestibular system being involved.

We leverage the above components to drive the user's attention (e.g., making him focus on chosen parts of a commercial or assisting in following a story, as decided by the creator) while improving user's perception by feeling less motion sickness, having to move less, and feeling more immersed by not missing major events in the 360° scene. We hence introduce the second editing strategy that we dub "dynamic editing" because it consists in repositioning, at runtime and when needed, the user in front of a certain region of interest, by rotating the sphere in a snap. On one hand, the user will undergo these repositionings, but we posit that, if done based on the scene content, they can go mostly unnoticed. On the other hand, by taking some freedom off of the user, we remove the gaze uncertainty by the same amount, the decision of which quality to fetch based on future gaze position are hence made exact, thereby lowering the amount of required replacements. The streaming application therefore needs to be upgraded to consider the presence of the forthcoming snap-changes and hence exploit the advantages of editing.

Implementation of snap-changes. First, the time positions of the snap-changes are chosen along the video (by hand for our proof-of-concept). For each snap-change, the angular position of the desired video sector to be shown in front of the user is found by superimposing a *millimetric paper* onto the projection of the 360° video in Adobe Premiere. Fig. 6 provides an example: the chosen sector is centered around the white truck whose angular position is −90°. For each snap-change, the angular value specifies by how many degrees the sphere should be rotated in order to have the desired part of the video in the FoV. The last information needed

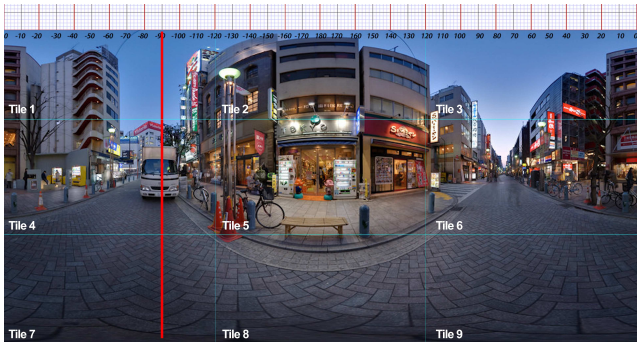

Figure 6: Identification of the targeted RoI angular position for snap-change construction.

for a snap-change are the indexes of the tiles overlapping the desired video sector, so as to use this knowledge in the fetching process (described in Sec. 4.2.5). These three pieces of information are gathered into an XML tag (example from Fig. 6 continued):

```
<?xml version="1.0"?>
<snapchange>
        <milliseconds>25000</milliseconds>
        <roiDegrees>-90</roiDegrees>
        <foVTile>1,2,4,5</foVTile>
</snapchange>
```

All the tags are chronologically sorted into an XML file which is used by the client application to dynamically edit the content. Following the "30 Degree Rule" which states that the shift in camera angle between two shots of the same subject must exceed 30° [31], we check 100ms before the time of the snap-change if the difference between the current position and the snap-change angle is lower than 30°. If so, the snap-change is not triggered. Otherwise, it is and the sphere is rotated by the angular difference in a snap, to make the user face the desired video sector.

4 VR STREAMING TESTBED: TOUCAN-VR

We present the main components of our testbed implementing an advanced streaming strategy composed of MPEG DASH-SRD tiled format with buffering, replacements and FoV-based quality decision for the Samsung Gear VR on Android. Source codes and usage guidelines are provided in App. A.

4.1 Preliminary components

Formatting the VR content for DASH-SRD. We start from a standard MP4 file containing an equirectangular projection of the camera's surrounding environment. Content must be split in space with tiling and in time to obtain segments. First, the *transcoding* phase creates several versions of the video which differ in resolution and/or bit rate. Then *tiling* is performed on each version. FFmpeg [24] is used for both these phases. Finally MP4Box [27], a tool from the GPAC library, is used for the *segmentation* phase, which splits each tile into segments, and produces the final DASH-SRD described content (with proper manifest file). The creation workflow is available as a Java script.

Virtual network. : The network is made of a Virtual Machine (VM) acting as an Apache HTTP web server, a VM dedicated to control the delay and bandwidth using the *tc* command. The latter serves as a gateway to the WiFi hotspot active on the host machine, so that the smartphone client connects to the virtual network using its WiFi interface. The DASH SRD-described contents are stored on the HTTP web server.

Parametrizer application. : The client consists of two distinct applications: *Parametrizer* lets tune the network parameters and preferences of the *TOUCAN-VR*, which is in charge of all the main VR functionalities. For the present work, we must be able to tune these parameters without modifying and recompiling each time the source code. The *Parametrizer* app (shown in Fig. 7) allows the following choices: video to be played; maximum buffer size (maximum video duration the video player can attempt to buffer); minimum buffer size (minimum video duration under which the video player starts again to buffer); minimum buffer size to launch playback; minimum buffer size to resume playback after a stall. We specify that these parameters are applied to each tile's buffer. The application also allows to enable or disable each one of the logging processes (see Sec. 4.2.4) and provides information about the tiling scheme adopted for the content.

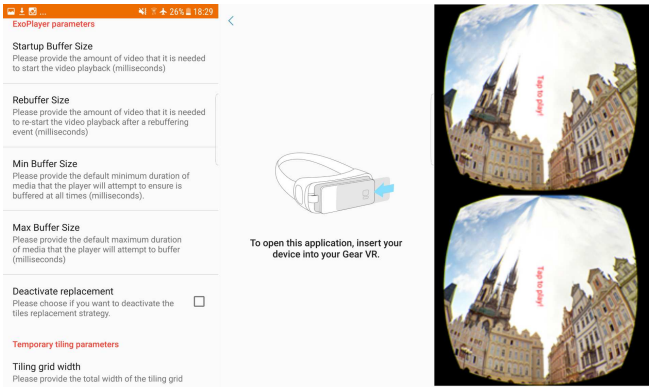

Figure 7: The Parametrizer application and the welcome screen.

4.2 TOUCAN-VR application

Video playback, streaming strategy, and logging system are under the control of the *TOUCAN-VR* application. It is an Android application that uses the Gear VR framework [42] provided by Samsung to build and show VR scenes. We use a Samsung Galaxy S7 Edge coupled with a Samsung Gear VR headset.

4.2.1 DASH-SRD media player. DASH-SRD-ready video players are not common yet: most of them are web players written in Javascript, and are not suitable for Android and the Gear VR framework. GPAC-Osmo4 [28] is a highly configurable multimedia player, implementing most of the existing delivery protocols, including DASH-SRD, and is multi-platform. In particular, Android is supported. However, Osmo4 presented some insurmountable problems: the playback of tiles is not well synchronized (even in MS Windows with a 3×3 tiled video), the light documentation of the C source code did not permit to correct the issue in a limited period

of time, and the player was not readily compliant with the Android version of the phone (7.0). That is why we instead considered ExoPlayer [26] which is an open-source media framework developed by Google for Android. It has a modular architecture and natively supports streaming using DASH.

We have extended ExoPlayer to introduce DASH-SRD support. First the MPD file parser has been extended so that the *supplemental property* tag containing details about tile locations could be properly taken into account. Second, the tiles fetched independently must be stitched back together in order to rebuild the 360° video. Leveraging the information about tile positions obtained at the previous step, each portion of content has been rendered in its correct position. Third, we achieve tile temporal synchronization by exploiting ExoPlayer's synchronization of video, audio and subtitle streams. Specifically, a *rendered* object is in charge of a single stream and all objects are synchronized by a player's internal module. We therefore add as many *renderers* as tiles, which guarantees Exoplayer plays them synchronously. The resulting *TOUCAN-VR* application is able to play locally-stored DASH-SRD described 360° videos. We then decribe ExoPlayer's processes for buffering, replacements and FoV-based quality decision.

4.2.2 Buffering and quality selection. Our proof-of-concept requires to have each tile's segment available in different qualities. The goal is however to understand the impact of editing on streaming performances. We therefore choose the simplest case with only two video qualities to choose from, namely a high quality (HQ) and a low quality (LQ). Keeping the quality decision as simple as possible (only based of FoV) indeed maximizes the interpretability of the impact of (i) motion and (ii) editing choices, on streaming performance. Our editorial strategies are however meant to be incorporated into full-fledged streaming systems, with refined quality selection algorithms and FoV prediction.

The streaming process is handled inside an infinite loop: at each iteration the player chooses a tile and a segment for that tile to be downloaded. The infinite loop is broken when the last segment of the last tile has been retrieved (information about video length and number of segments is available in the MPD file).

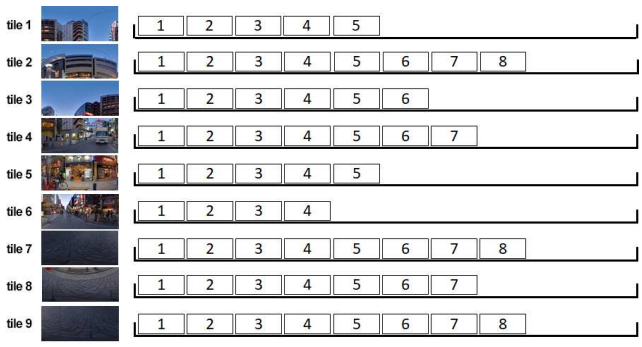

Figure 8: Each tile is assigned a playback buffer for the segments to be played out.

As depicted in Fig. 8, each tile has its own buffer. At each loop iteration the next buffer to have a segment downloaded is the one the most behind in terms of video seconds buffered (tile 6 in Fig. 8). Which quality to download for this segment is then decided on whether the buffer's tile is currently in the FoV (HQ selected if it is, LQ otherwise). Fig. 9 depicts the case where the FoV overlaps the four top-right tiles.

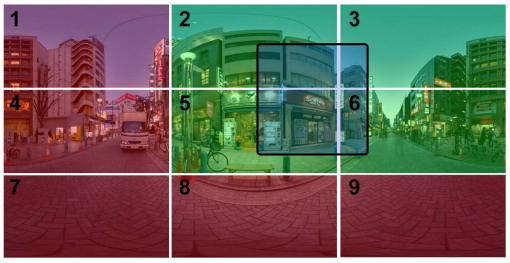

Figure 9: Quality selection: the blue square represents the user's FoV. Green and red tiles are requested in HQ and LQ, respectively.

4.2.3 Replacements: ensuring streaming responsiveness to user's motion. When all the tiles' buffers have reached the maximum buffer size (defined in Sec. 4.1), the buffering process is stopped and the player tries to improve the system responsiveness by improving the segments' quality of the tiles currently in the FoV.

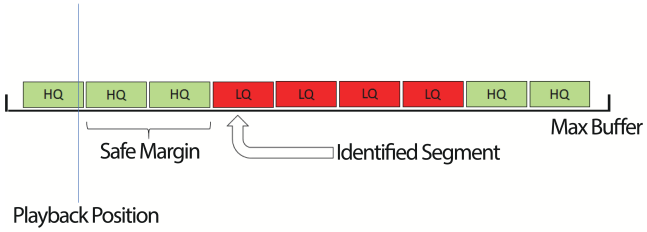

Figure 10: Replacement: identification of the possible segment to be replaced.

For each such tile, the first LQ segment is identified after a safety margin set to 2 seconds, as depicted in Fig. 10. It is indeed not worth trying to replace a segment that will be played very soon because there might not be enough time to re-download it. Let us mention the segment duration we consider is 1s. The LQ segment is not discarded until the HQ download in complete. The segment replacement is aborted if (i) the playback reaches the segment for which the replacement has been fired, in which case the LQ segment is shown to prevent a stall, or (ii) the downloading process takes too much time and any buffer has depleted down to the minimum buffer size: the buffering process is resumed. Owing to an available functionality of ExoPlayer, not exploited by default, the replacement strategy has been implemented by discarding and replacing all the chunks stored behind the first LQ segment after the safety margin. Although possibly increasing the number of replacements compared with a single-segment replacement approach, this high-level approach has allowed to easily develop a replacement strategy without modifying the low-level functions of the player.

Also, by setting the max buffer size equal to the min buffer size, we ensure that the maximum number of segments replaced are those downloadable within a segment duration (1s).

4.2.4 Logging process. In order to collect objective metrics from UX, a number of logging threads run together with the graphic-related threads during playback. They collect: consumed bandwidth (downloaded bytes), successfully replaced segments, freezing events, quality displayed for each tile's segment, tiles in the FoV, user's head position. The latter is obtained with the Samsung Gear VR framework's built-in functions. About 60 samples per second are collected for the angle triplet (pitch, roll, and yaw), later used to get the mean velocity of head motion. Logging threads have been carefully designed in order not to impact graphics performance of the application.

4.2.5 Benefiting from dynamic editing. In the buffering phase, for each tile, when the segment to be played when the snap-change occurs is scheduled for download, the quality selection is not made based on the current FoV, but instead based on the exact knowledge of the future FoV position: the segments of the tiles listed in the *foVTile* tag of the XML file (defined for the description of the snap-changes in Sec. 3.4) are requested in HQ, the others in LQ. Until the snap-change occurs, the download of the subsequent segments is made in the same way.

Similarly, a margin is considered to prevent replacements too close to a snap-change: if the FoV does not correspond to that of the snap-change, then it is useless to download in HQ tiles which will not fall into the future FoV, if the FoV corresponds, then by construction the HQ segments of the right tiles have been already buffered. So replacements are prevented 6s before a snap-change. This parameter is important as it controls the trade-off between (i) bandwidth overhead (number of replacements made), equivalently bandwidth savings ahead of the snap, (ii) and adaptation to a new FoV ahead of the snap. Its detailed impact on this trade-off will be investigated in future studies.

5 USER EXPERIMENTS: DESIGN

The design of the user experiments (UX) has been made following [34, Chap. 20]. This section first states formally the hypotheses to be tested from the UX, then exposes the experimental plan and finally gives details about its execution. In our concern for reproducible research, the dataset obtained and the videos, like the testbed, are made available as described in App. A.

5.1 Hypotheses to prove or disprove

Our goal is to know whether editing impacts user's motion, bandwidth, and subjective assessment. We therefore adopt a progressive approach, defining basic then more refined hypothesis.

First we define the so-called "random" counterpart to each editing strategy. In Random Static (RS) editing, the alignment of RoI from one scene to the next is random. Comparing RS with Static (S) editing will tell whether the base idea of reducing abrupt head motion occurring at scene transitions indeed verifies. In Random Dynamic (RD) editing, the user is repositioned at predefined instants to face a random spot in the same scene. It is compared with semantic Dynamic (D) editing, where the snap-changes are made at the same time as in RD, but the user is repositioned in front of a RoI meaningful at that time of the video. Comparing RD with D will tell whether the user's motion is only affected by the repositionings, independently of the meaning of the content in their new FoV, or if the latter has an impact. The sheer action of repositionings might indeed decrease people's natural tendency to move, the semantics not being the main contributor in motion reduction. The first set of hypotheses we want to test is therefore:

- H1: S induces less motion than RS
- H2: D induces less motion than RD
- H3: S consumes less bandwidth than RS
- H4: D consumes less bandwidth than RD

As shown in Sec. 6, H1 to H4 get verified. The hypothesis tested are then:

- H5: D induces less motion than S
- H6: D consumes less bandwidth than S
- H7: D improves the perceived quality of experience compared with S

We run an additional set of UX to focus on the pairwise comparison between S and D, specifically for the subjective assessment.

In order to limit the impact of external factors on these preliminary comparisons of editing strategies, we run the experiments in only one bandwidth condition, which allows to stream the contents without stalls. The numerical details are provided in Sec. 5.3.

Working at the content-level requires to consider the semantics of the content which may heavily impact the user's behavior. To be representative of different contexts, we have considered a commercial-related content, which is a luxury hotel virtual visit referred to as "Hotel", and a story-telling content, which is a thriller series taken from [32] and referred to as "Invisible". From episodes 3 and 6, we create a content with a variety of scene transitions, which lasts 5min 20 s, and whose story is intelligible. Details are given in Sec. 5.3.

5.2 Experimental Plan

Following the methodology and terminology of UX design [34], we define a treatment undergone by a user as one content watched with a given editing. To test the above hypotheses, we consider paired tests to investigate the difference in metrics obtained by a given user on a given content edited in two different ways. Indeed, as a high variability in the motion is expected between the users [40], having different users going through different editing versions of the same content might result in differences due to the users and not to the editing, if the user population is restricted. Paired tests allow to get freed from the inter-user motion variability. We set the number of samples per test to 4. It is known from statistics on small sample sizes that the drawback of having a low number of samples per test is that only big effects can be detected with sufficient power. So we consider that, for the editing to be worth investigating as a lever for streaming optimization, its effect must be big enough for being observed with few samples. This also allows to have a reasonable number of users to enroll in our preliminary experiment. We opt for a full factorial design. For 2 contents and 3 comparisons, to have 4 samples per test, 24 users would be required. We reduce this number in practice by suggesting each

user to take another test if they are willing too (7 in 17 users have accepted).

We thereby come up with the experimental plan detailed in Table 1, drawn in the case each user undergoes 2 tests. The design pays particular attention to randomization and blocking. Randomization is used such that: (i) the number of people starting with Content 1 is the same as the number of people starting with Content 2 and (ii) the order in which the editings are shown to the users is randomized and balanced to reduce the effects due to the order of visualization. Blocking ensures spreading of homogeneous groups of users over the different treatments. A group is homogeneous if the group's users are likely to respond similarly to the treatments. In our case, there are 4 blocks of 6 units each, and we tried to group people together as follows: below, between or above 25-40 year-old, past-VR experience, wearing glasses (not contact lenses) and gender. Blocking was made from the pre-questionnaire. The subjective assessments are performed with the single-stimulus method.

Paired editings		RS - S	D - RD	S - D
Hotel	Order 1	$U_{8,2}\ U_{12,1}$	$U_{3,1}\ U_{5,2}$	$U_{9,2}\ U_{10,1}$
	Order 2	$U_{2,2}\ U_{4,1}$	$U_{1,2}\ U_{6,1}$	$U_{7,1}\ U_{11,2}$
Invisible	Order 1	$U_{5,1}\ U_{6,2}$	$U_{10,2}\ U_{11,1}$	$U_{8,1}\ U_{12,2}$
	Order 2	$U_{1,1}\ U_{3,2}$	$U_{7,2}\ U_{9,1}$	$U_{2,1}\ U_{4,2}$

Table 1: Experimental plan. $U_{i,j}$ is sample from user i going through her/his j-th viewing session. Order refers to the order in which the editings are viewed: 1 or 2 for the order mentioned at the top of the array or the reverse, respectively.

5.3 Execution of user experiments

High-level parameters. The characteristics of the created videos are for Hotel (resp. Invisible): duration: 2min 10s (5min 20s); number of scenes (environments): 9 (15); number of built-in camera repositionings within the same environment: 0 (20); number of snap-changes (i.e., FoV repositionings) added: 9 (17).

Low-level parameters. The values of the client application parameters detailed in Sec. 4 have been set to: tiling scheme: 3×3; maximum buffer size: 10 s; minimum buffer size: 10 s; minimum buffer size for playback: 3s; minimum buffer size for playback after a stall: 3s; safe margin from playback position to trigger replacements: 2s; safe margin from snap-change to prevent replacements: 6s; number of tiles: 9; segment duration: 1s; encoding rates LQ (HQ) aggregated over all tiles, Hotel: 4.2Mbps (15.4Mbps), Invisible: 2.2Mbps (10Mbps); virtual network bandwidth (delay): 100Mbps VM, WiFi access 30Mbps (10ms).

Environment. As aforementioned, two sets of UX have been run, in August and October 2017. The first set is described in Table 1 and has involved 17 different people (graduate students, researchers and administrative staff), 7 of whom have accepted to run two tests. The second set has involved 21 persons (not overlapping with the first set). The second set has been devised to correct the ambiguities in the subjective assessment: for the task score, users are asked whether they have seen specific elements. After the second editing viewed, they could not remember whether they had seen it in the first or second version/viewing. To correct that, we split the content in two equal parts to ask questions about elements not appearing in both halves, so that the users in the second set of UX evaluated the editing on the same type of content but not the same piece of video.

In a nutshell, one user experiment is made of: a pre-questionnaire for blocking, introductory videos to correctly adjust headset and get familiar with the VR environment, the actual viewing session and finally the post-questionnaire with the subjective assessments is filled. After each viewing, the logs are archived. All experiments have been made standing not to constrain the head motion (the users could grab the back of a chair for balance if needed).

6 USER EXPERIMENTS: ANALYSIS

We first confirm that editing impacts head motion, specifically that H1, H2 and H5 are verified. We then confirm that proper editing induces less bandwidth consumption, confirming H3, H4 and H6. A refined analysis shows the share between savings in wasted and displayed bandwidth. We show that the relation between replacements (hence bandwidth) and head motion is close to linear with static editings (RS and S), but interestingly sub-linear with dynamic editings (RD and D), while maintaining a high quality in the FoV. Finally we analyze in detail subjective assessment comparing static and dynamic. H7 is verified for the task metric and sickness. More investigation is required for other metrics.

6.1 Head motion

Let us first analyze whether H1, H2 and H5 are verified, i.e., whether the editing impacts the head motion. Table 2 stores the t-values and p-values of the paired t-tests (difference made in the order of the naming). The p-value of each row represents the risk of accepting the hypothesis that the second mentioned editing in the second column decreases the head motion for the content, compared with the first mentioned editing. The corresponding samples are represented as boxplots in the top-left graphic in Fig. 11. Head motion and (mean) head velocity are used interchangeably.

With a significance level of 7%, we verify hypotheses H1 and H2 for comparison between random and non-random editing, both static and dynamic (that are RS-S and RD-D) for both contents. This the reason why we stated the hypothesis in Sec. 3.3Âăturns out reasonable: the lower head motion incurred with Static (that is with RoIs aligned between two successive scenes) than with Random Static (not aligned). As the difference between S and RS is only the matching of RoIs at scene change, we deduce the quantity of motion is lower at scene change, i.e., is lowered by the matching of RoIs. From Fig. 11, we can see the head motion reduction lies between 10% and 30% for the different pairwise comparisons. As aforementioned, each user compares a pair of editing, so that the difference in head motion is not impacted by the inter-person motion variability. Therefore we cannot directly compare the head motion statistics with the two extreme choices RS and D from the samples. Yet, combining the improvements from RS to S (d_{RS-S}) then S to D (d_{S-D}) leads to estimate the improvement from RS to D to about 34% ($1-(1-d_{RS-S})(1-d_{S-D})$). A careful editing, both

inter-scene (static) and intra-scene (dynamic), is therefore crucial to lower the mean head motion. More particularly, we observe the editing seems to impact more the user motion in the case of the story than the hotel visit.

Analyzing then the non-trivial editings (H5), i.e., static (S) versus dynamic (D), we observe the p-values are higher, around 0.1 for both contents. As these are the two most interesting editings, the second round of user experiments we have carried out has focused on this comparison. Owing to the factorial design, the first round of UX features 4 samples per comparison only, while the second round features 11 (Hotel) or 10 (Invisible). The p-values in Table 3 confirm that, for the story content, dynamic editing saves head motion compared with static editing, with a significance level of 6%. While the p-value of the second round of UX for Hotel tends to reject H5, it is worth noting that this comparison has been made between different pieces of the content (for the sake of the subjective evaluation, as described in Sec. 5.3). The videos have been cut in half, resulting in a quite short experience for Hotel (about 1min, contrary to Invisible, 2min 40s), hence mitigating the rejection. Boxplots as those in Fig. 11, not shown here for the sake of conciseness, confirm the median levels of head motion reduction, around 20%, as well as the lesser spreading for the story content.

Content	Difference of editing	t-value	p-value
Hotel	RS-S	2.17	0.059
	RD-D	6.99	0.003
	S-D	1.73	0.091
Invisible	RS-S	2.07	0.065
	RD-D	5.03	0.008
	S-D	1.57	0.107

Table 2: Head Motion, first set of UX: paired t-tests.

Content	Difference of editing	t-value	p-value
Hotel	S-D	0.19	0.427
Invisible	S-D	1.75	0.057

Table 3: Head Motion, second set of UX: paired t-tests

6.2 Bandwidth

Let us now analyze the impact of editing on bandwidth consumption.

The p-values shown in Table 4 for the difference in total consumed bandwidth between the different pairs of editing, confirm H3 and H4 with a significance level of 5%. The second set of UX allows to confirm the hypothesis for S-D too (H6), as shown in Table 5. The bottom-left graph of Fig. 11 shows decreases from worst (RS) to best (D) editing of 25% and 13% for Hotel and Invisible, respectively. Let us now analyze the reasons for such decreases, both qualitatively and quantitatively. First, we notice that the level of reduction in total bandwidth does not follow the level of reduction in head motion. For example, the range of decrease in total bandwidth from S to D for the Hotel content (middle-left graph) is higher than that of head motion (top-left graph).

Fig. 13 shows that the consumed bandwidth is linearly dependent on the number of replacements. However, Fig. 12 shows that the number of replacements varies differently in the head velocity, depending on whether it is static (RS or S) or dynamic (RD or D) editing. While the growth is close to linear for static, we observe it is sub-linear for dynamic. This is due to the consideration of the snap-changes in the streaming process, as detailed in Sec. 4.2.5: owing to the exact knowledge of the FoV upon the snap-changes, replacements are not allowed less than 6 seconds before the snap-change. Hotel (resp. Invisible) lasts 120s (resp. 320s) and features 9 (resp. 17) snap-changes, so the replacements cannot occur $6 \times 9/120 = 45\%$ of the time (resp. 32%).

We hereby expose that, in the dynamic editing strategy, the number of replacements, hence the consumed bandwidth, does not decrease only by the reduction in head velocity (as is the case for static), but also benefits from the exact knowledge of the future repositioning.

As a side note regarding Fig. 12, let us mention that, contrary to Invisible, Hotel results make appear only a slight increase in replacements with head velocity with dynamic. This is due to the ratio between the head velocity and the fraction of time with allowed replacements. Investigating the actual relation between replacements and head motion will be made easier with a motion generator, to get freed from the human head velocity limitations.

More in detail, the total consumed bandwidth is the sum of two components: the bandwidth spent in downloading tiles finally rendered, named the *displayed bandwidth*, and tiles not rendered because replaced, named the *wasted bandwidth*. The middle-bottom graph in Fig. 11 shows that the savings in wasted bandwidth are 26% (resp. 19%) for Hotel (resp. Invisible). As expected, Fig. 13 shows that the wasted bandwidth varies linearly with the number of replacements, so does the displayed bandwidth. This is the total weight of the tiles rendered on the sphere (not only in the FoV). Indeed, a replacement means that the tiles whose segments have been initially downloaded in HQ are not those in the current FoV. At a given time, the number of tiles rendered in HQ is therefore a linear function of the number of replacements, so is the displayed bandwidth.

Importantly, the top-right graph of Fig. 11 shows that the bandwidth savings are not made at the cost of a lower quality in the FoV. The relative quality, i.e., the fraction of tiles in HQ in the FoV, is indeed maintained throughout the different editings. This simple metric is representative of V-PSNR [29] because the compared content have the same encoding. Freezes do not occur because the streaming strategy is conservative by construction, and the set bandwidth condition allows streaming without stalls. Further discussion on stalls is provided in Sec. 7.

6.3 Subjective metrics of QoE

We present the subjective assessment of the editings only for static (S) and dynamic (D), discarding their random versions. The second round of UX has included 21 persons, 11 assessing the difference between S and D for Hotel, 10 for Invisible.

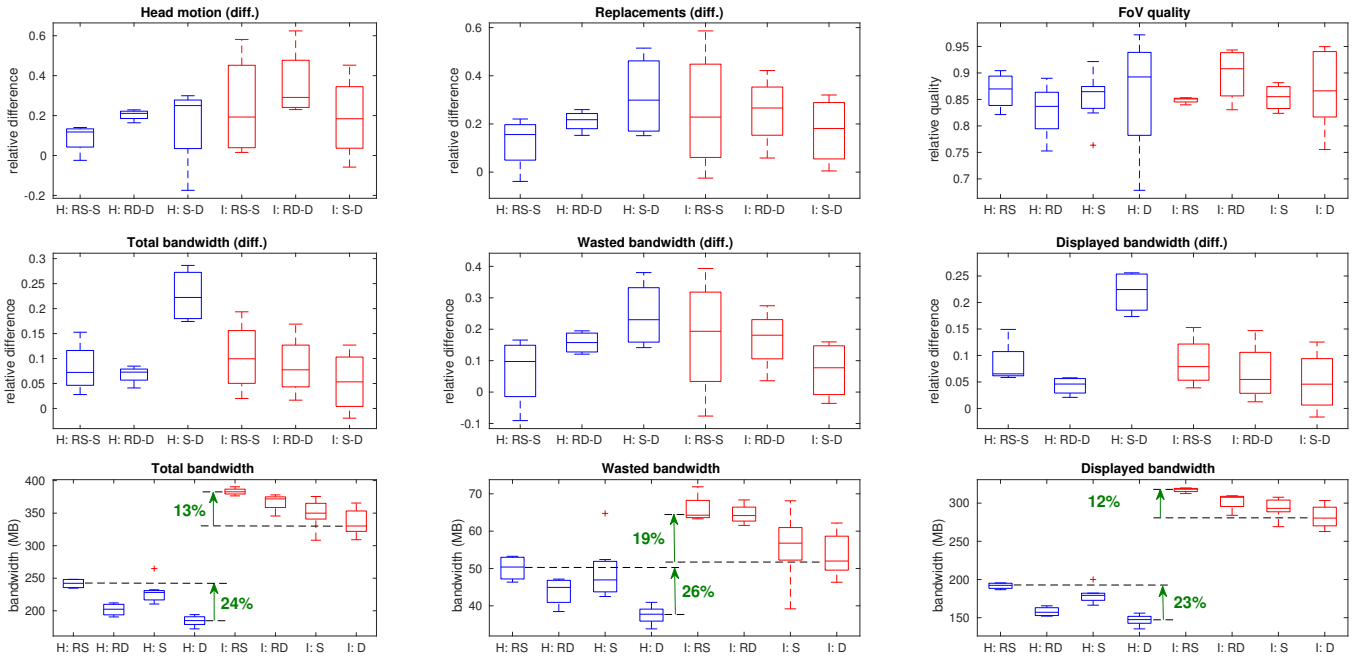

Figure 11: Boxplots of objective metrics, first set of UX. Comparison between all four editing strategies: Random Static (RS), Static (S), Random Dynamic (RD), Dynamic (D). Both contents are H (Hotel) in blue and I (Invisible) in red.

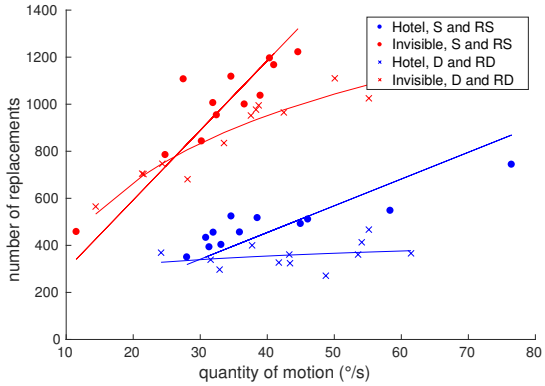

Figure 12: Number of replacements vs. head motion, first set of UX. Solid curves are linear and logarithmic regressions.

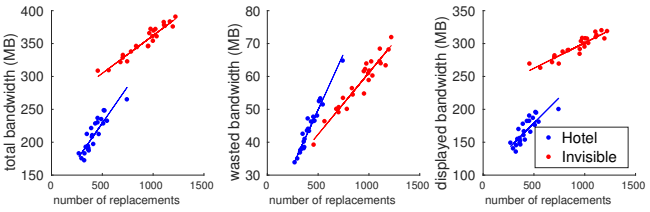

Figure 13: Total, wasted and displayed bandwidth vs. number of replacements, first set of UX. Solid curves are linear regressions.

Fig. 14 shows the subjective metrics collected at the end of each experiments. Each row corresponds to a set of questions on performance, satisfaction and comfort, respectively. All metrics are collected as Mean Opinion Scores (MOS) into $\{1, \ldots, 5\}$. The performance score is the sum of correct responses to each task-related question. A question is whether the user has seen a specific object or event in the video. In the Hotel video, these are assets such as jacuzzi or bar, and in the story content Invisible, these are key events not to miss to follow the story. The top-left graph shows that positioning the snap changes on the chosen events (as chosen by the creator), allows to reach a maximum score for both contents, higher than that obtained by static editing (i.e., taking care of the scene transitions but without snap changes).

The middle-left graph shows that the overall rating of the visual quality is the same for both editings and contents, which correlates with the similar relative quality in the FoV shown in Fig. 11. Users have also been asked whether they sometimes felt a repositioning (score 5 assigned to a 'no'), and if so, to rate (between 1 and 5) whether this has bothered them. The center graph shows that the snap changes have been sometimes detected, but not felt much disturbing, in particular even less for the story than for the hotel visit, leading to think that the repositionings truly help the user to follow an active story in VR.

One drawback we expected from the snap-changes was that the change of plan would make the users feel as watching TV, i.e., would have the action unfolding in front of them without being able to participate by moving, hence feeling less immersed. However, the middle-right graph shows that the users rated equivalently their feeling of immersion in the content, for both contents. Expectedly, the feeling of immersion is higher within a captivating

Metric	Content	Editing diff.	t-value	p-value
Replacements	Hotel	RS-S	2.14	0.061
		RD-D	9.08	0.001
		S-D	2.48	0.045
	Invisible	RS-S	2.00	0.07
		RD-D	3.5	0.02
		S-D	2.23	0.056
Total bandwidth	Hotel	RS-S	3.01	0.029
		RD-D	6.44	0.004
		S-D	6.11	0.004
	Invisible	RS-S	2.84	0.033
		RD-D	2.65	0.039
		S-D	1.67	0.097
Wasted bandwidth	Hotel	RS-S	1.23	0.154
		RD-D	6.58	0.004
		S-D	3.22	0.024
	Invisible	RS-S	1.83	0.082
		RD-D	3.44	0.021
		S-D	1.48	0.117
Displayed bandwidth	Hotel	RS-S	3.78	0.016
		RD-D	4.79	0.009
		S-D	8.35	0.002
	Invisible	RS-S	3.6	0.018
		RD-D	2.3	0.053
		S-D	1.67	0.097

Table 4: Replacements, bandwidth components, first set of UX: paired t-tests.

Metric	Content	Editing diff.	t-value	p-value
Replacements	Hotel	S-D	3.14	0.005
	Invisible	S-D	3.12	0.006
Total bandwidth	Hotel	S-D	6.63	3.00e-05
	Invisible	S-D	6.14	9.00e-05
Wasted bandwidth	Hotel	S-D	1.27	0.117
	Invisible	S-D	1.7	0.062
Displayed bandwidth	Hotel	S-D	8.01	6.00e-06
	Invisible	S-D	3.97	0.002

Table 5: Replacements, bandwidth components, second set of UX: paired t-tests.

story than within the visit of a hotel. Future user experiments will however feature a double-stimuli approach to better assess the editing impact on immersion. Also, we emphasize that we do not expect the editing strategies we introduce to be neutral to the user's feeling of immersion. Our rational is to introduce new degrees of freedom that allow to reach new operational points in the streaming process, trading, e.g., immersion for streaming performance when the network requires.

The bottom-left graph shows that the discomfort (users were asked for dizziness and sickness) is high for the story in static editing. Looking in detail at the reasons the users gave for feeling uncomfortable, it appeared the most common cause are the action scenes, in particular one in which there is a camera motion (from the shooting, independent of the editing), and one with a fight in an interior environment, close to and around the camera. The snap changes have been placed so as to lower the need to move in these action scene, which proved effective. The bottom-right graph depicts how much the users felt limited in their ability to explore the scenes as they wanted. Remarkably, despite the arbitrary repositionings, they did not feel more limited with dynamic than with static editing, except slightly for the Hotel content. However, the dominating reason for such feeling was the inability to move the camera in the space, e.g., to change rooms, which is again not related to editing. Hence, further experiments would be needed to determine whether dynamic editing adds action entertaining the user, feeling less bored in the closed headset environment. Interestingly, the users felt like they had to move their head too much to follow the story content with dynamic editing. This seems in contradiction with the previous values of discomfort (if we associate head motion with a more likely sickness). Detailed feedback from the user are required to sort out the exact reasons for both feelings, in particular the exact times of the video when it happened. This will allow to correlate the feelings with the introduced repositionings, or with other numerous possible factors in the video (scene motion, camera motion, wide angular range of action, etc.). Future user experiments will therefore be designed so as to enable temporally detailed collection of the user's feelings during the experiment to extract these needed time correlation. Also, the experiments were all made standing. The impact of dynamic editing may be more important in viewing environment where the user is sitting and may better appreciate the help to follow the action without needing to wring their neck.

The results on these last two metrics, feelings of limitation and excessive need to move, therefore show that more refined user assessments are needed to explain and hence control the impact of the editing on the user's experience in VR.

7 DISCUSSION

For our proof-of-concept, we have made 2 examples of editing tested on 2 examples of content, under a bandwidth condition allowing no freezes with a conservative buffering strategy. The results allow to envision that for example, in limited bandwidth conditions, the streaming decisions would control the frequency of snap-changes to improve quality in the FoV without needing replacements. The snap-changes would be picked as needed from an XML file prefilled by the creator at the time of content creation, or automatically generated by leveraging available computer vision tools, such as saliency maps.

Although we do not claim these examples to be fully representative of the variety of contents impacting the type of user behaviors, we believe our results call for a thorough investigation of user's attention driving tools from other communities. These tools would

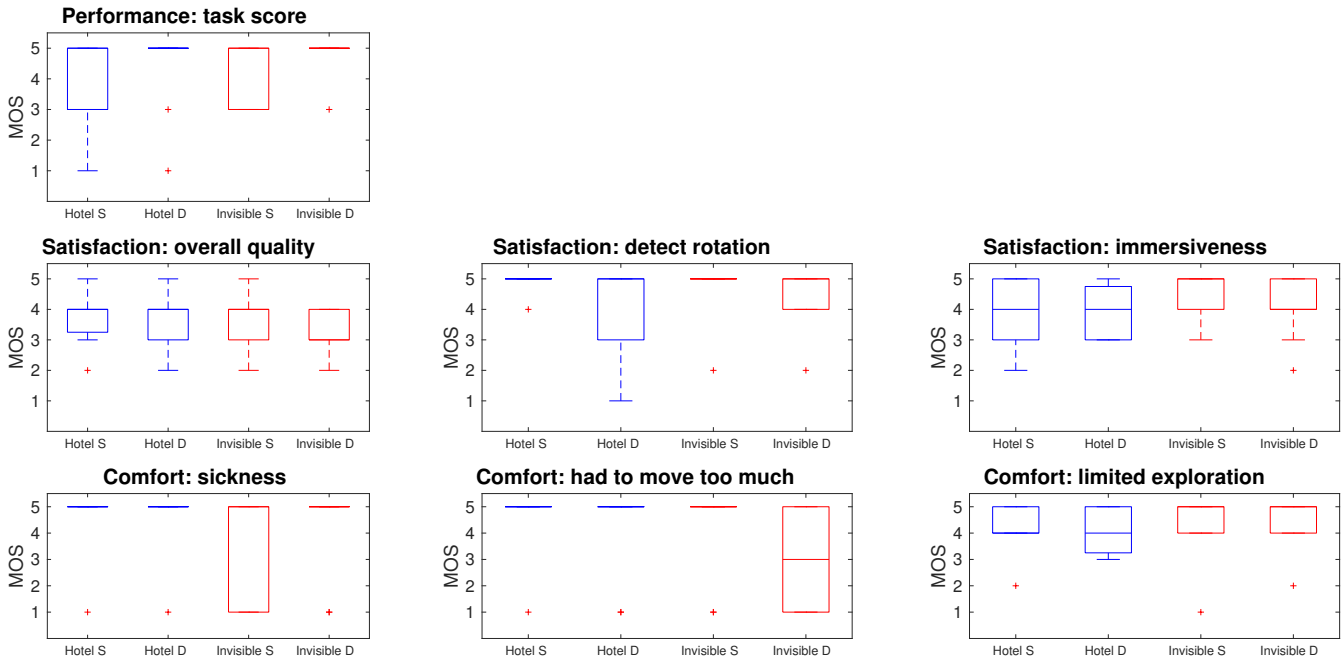

Figure 14: Subjective metrics, second set of UX. Comparison between both non-random editing strategies: Static (S) and Dynamic (D). Both contents are Hotel in blue and Invisible in red.

be designed to be wielded by the streaming module so as to improve the streaming experience in even lower bandwidth conditions.

8 CONCLUSION

This article has presented a strongly innovative approach to improve the streaming of 360° videos: designing high-level content manipulations (enlarged editing) to limit (with static editing) and even control (with dynamic editing) the user's motion in order to consume less bandwidth while maintaining the user's experience. We have designed editing from film-making techniques and VR HCI, which can be leveraged to improve streaming. From the most recent literature, we have built an MPEG DASH-SRD player for Android and the Samsung Gear VR, featuring FoV-based quality decision and a replacement strategy to allow the tiles' buffers to build up while keeping their state up-to-date with the current FoV as much as bandwidth allows. The dynamic editing we designed has been integrated within the player, and the streaming module has been extended to benefit from the editing. Two sets of user experiments on 17 and 21 users enabled to show that editing indeed impacts head velocity (reduction of up to 30%), consumed bandwidth (reduction of up to 25%) and subjective assessment (improvement of detection of desired elements, reduction in sickness in action scenes).

The results of our inter-disciplinary approach showed that user's attention driving tools from other communities can be designed in order to improve streaming. We believe this opens up the path to a whole new field of possibilities in defining degrees of freedom to be wielded in optimization of VR streaming.

ACKNOWLEDGMENTS

This work has been supported by the French government, through the UCAJEDI Investments in the Future project managed by the National Research Agency (ANR) with reference number ANR-15-IDEX-01.

REFERENCES

[1] 360fly. 2016. *360fly 4K User Guide.*
[2] P. Rondao Alface, M. Aerts, D. Tytgat, S. Lievens, C. Stevens, N. Verzijp, and J.-F. Macq. 2017. 16K Cinematic VR Streaming. In *ACM Multimedia Conf.*
[3] Android application. 2017. Within: Storytelling for Virtual Reality. https://with.in/. (2017).
[4] B. Begole. 2016. Why The Internet Pipes Will Burst When Virtual Reality Takes Off. (Feb. 2016). Forbes.
[5] D. Bordwell and K. Thompson. 2006. *Film Art: An Introduction.* McGraw-Hill.
[6] J. Brillhart. 2016. VR and cinema. https://medium.com/@brillhart. (May. 2016). Google I/O conf.
[7] C. Brown, G. Bhutra, M. Suhail, Q. Xu, and E. D. Ragan. 2017. Coordinating attention and cooperation in multi-user virtual reality narratives. In *IEEE VR.*
[8] Cisco. 2017. Cisco Visual Networking Index: Forecast and Methodology, 2016-2021. (June 2017). White paper.
[9] X. Corbillon, A. Devlic, G. Simon, and J. Chakareski. 2017. Optimal Set of 360-Degree Videos for Viewport-Adaptive Streaming. In *ACM Multimedia Conf.*
[10] X. Corbillon, G. Simon, A. Devlic, and J. Chakareski. 2017. Viewport-adaptive navigable 360-degree video delivery. In *IEEE ICC.*
[11] International Data Corporation. 2018. Demand for Augmented Reality/Virtual Reality Headsets Expected to Rebound in 2018. (Mar. 2018). Industry report.
[12] A. Coutrot and N. Guyader. 2017. Learning a time-dependent master saliency map from eye-tracking data in videos. *CoRR* abs/1702.00714 (2017). arXiv:1702.00714
[13] J.-E. Cutting, K.-L. Brunick, J.-E. DeLong, C. Iricinschi, and A. Candan. 2011. Quicker, faster, darker: Changes in Hollywood film over 75 years. *i-Perception* 2,

6 (2011), 569–576.
[14] S. Dambra, G. Samela, L. Sassatelli, R. Pighetti, R. Aparicio-Pardo, and A.-M. Pinna-Déry. 2018. TOUCAN-VR. *Software* (DOI: 10.5281/zenodo.1204442 2018). https://github.com/UCA4SVR/TOUCAN-VR
[15] S. Dambra, G. Samela, L. Sassatelli, R. Pighetti, R. Aparicio-Pardo, and A.-M. Pinna-Déry. 2018. TOUCAN-VR-data. *Software* (DOI: 10.5281/zenodo.1204392 2018). https://github.com/UCA4SVR/TOUCAN_VR_data
[16] S. Dambra, G. Samela, L. Sassatelli, R. Pighetti, R. Aparicio-Pardo, and A.-M. Pinna-Déry. 2018. TOUCAN-VR-parametrizer. *Software* (2018). https://github.com/UCA4SVR/TOUCAN_VR_parametrizer
[17] S. Dambra, G. Samela, L. Sassatelli, R. Pighetti, R. Aparicio-Pardo, and A.-M. Pinna-Déry. 2018. TOUCAN-VR-preprocessing. *Software* (2018). https://github.com/UCA4SVR/TOUCAN-preprocessing
[18] V. A. de Jesus Oliveira, L. Brayda, L. Nedel, and A. Maciel. 2017. Designing a Vibrotactile Head-Mounted Display for Spatial Awareness in 3D Spaces. *IEEE Trans. on Visualization and Computer Graphics* 23, 4 (Apr. 2017), 1409–1417.
[19] T. El-Ganainy and M. Hefeeda. 2016. Streaming Virtual Reality Content. *CoRR* abs/1612.08350 (2016). arXiv:1612.08350
[20] Facebook. 2017. Enhancing high-resolution 360 streaming with view prediction. (Apr. 2017). Facebook Developers Conference.
[21] Facebook. 2017. The Evolution of Dynamic Streaming. (Apr. 2017). Facebook Developers Conference.
[22] Facebook. 2017. VR201: Lessons from the Frontlines. (Apr. 2017). Facebook Developers Conference.
[23] C.-H. Fan, J. Lee, W.-C. Lo, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu. 2017. Fixation Prediction for 360 Video Streaming in Head-Mounted Virtual Reality. In *ACM NOSSDAV*.
[24] FFmpeg. 2017. FFmpeg. https://www.ffmpeg.org/. (2017).
[25] V. R. Gaddam, M. Riegler, R. Eg, C. Griwodz, and P. Halvorsen. 2016. Tiling in Interactive Panoramic Video: Approaches and Evaluation. *IEEE Trans. on Multimedia* 18, 9 (Sep. 2016), 1819–1831.
[26] Google. 2017. Exoplayer. http://google.github.io/ExoPlayer/. (2017).
[27] GPAC. 2017. MP4Box. https://gpac.wp.imt.fr/mp4box/. (2017).
[28] GPAC. 2017. Osmo4. https://gpac.wp.imt.fr/player/. (2017).
[29] M. Graf, C. Timmerer, and C. Mueller. 2017. Towards Bandwidth Efficient Adaptive Streaming of Omnidirectional Video over HTTP: Design, Implementation, and Evaluation. In *ACM MMSyS*.
[30] Q. Huynh-Thu, M. Barkowsky, and P. Le Callet. 2011. The Importance of Visual Attention in Improving the 3D-TV Viewing Experience: Overview and New Perspectives. *IEEE Trans. on Broadcasting* 57, 2 (Jun. 2011), 421–431.
[31] Hollywood lexicon. 2000. 30 Degree Rule. http://www.hollywoodlexicon.com/thirtydegree.html. (2000).
[32] D. Liman. 2016. Invisible. http://www.imdb.com/title/tt6178894/. (2016). VR series, Samsung.
[33] O. Niamut, E. Thomas, L. D'Acunto, C. Concolato, F. Denoual, and S.-Y. Lim. 2016. MPEG DASH SRD: Spatial Relationship Description. In *ACM MMSyS*.
[34] G.W. Oehlert. 2010. *A First Course in Design and Analysis of Experiments*. Macmillian.
[35] J. Orlosky, K. Kiyokawa, and H. Takemura. 2017. Virtual and Augmented Reality on the 5G Highway. *J. of Information Processing* 25 (2017), 133–141.
[36] S. Petrangeli, V. Swaminathan, M. Hosseini, and F. De Turck. 2017. An HTTP/2-Based Adaptive Streaming Framework for 360 Virtual Reality Videos. In *ACM Multimedia Conf.*
[37] S. Petrangeli, V. Swaminathan, M. Hosseini, and F. De Turck. 2017. Improving Virtual Reality Streaming Using HTTP/2. In *ACM MMSyS*.
[38] J.-T. Petty. 2016. Gone. http://www.imdb.com/title/tt4876244/. (2016). VR series, Samsung.
[39] J. De Praeter, P. Duchi, G. Van Wallendael, J.-F. Macq, and P. Lambert. 2016. Efficient Encoding of Interactive Personalized Views Extracted from Immersive Video Content. In *ACM Int. Workshop on Multimedia Alternate Realities*.
[40] M. Reilhac. 2016. Presence Design and Spatial Writing in Virtual Reality. (Jun. 2016). Lecture at FilmCamp Norway.
[41] S. Saarinen, V. Makela, P. Kallioniemi, J. Hakulinen, and M. Turune. 2017. Guidelines for Designing Interactive Omnidirectional Video Applications. In *IFIP INTERACT*.
[42] Samsung. 2017. Samsung Gear VR Framework. (2017).
[43] Preston Sturges. 1991. *Preston Sturges by Preston Sturges: His Life in His Words*. Touchstone.
[44] A. TaghaviNasrabadi, A. Mahzari, J. D. Beshay, and R. Prakash. 2017. Adaptive 360-degree video streaming using layered video coding. In *IEEE VR*.
[45] C. Milk (Within). 2016. The birth of virtual reality as an art form. (Jun. 2016). TED talk.
[46] M. Xiao, C. Zhou, Y. Liu, and S. Chen. 2017. OpTile: Toward Optimal Tiling in 360-degree Video Streaming. In *ACM Multimedia Conf.*
[47] C. Zhou, Z. Li, and Y. Liu. 2017. A Measurement Study of Oculus 360 Degree Video Streaming. In *ACM NOSSDAV*.

A ARTIFACTS OF THE ARTICLE

A.1 List of the artifacts

Our artifacts are organized into 4 Github repositories.

The 2 main repositories are:

- TOUCAN-VR [14] containing the main Android app, which is the streaming client.
- TOUCAN-VR-data [15] storing the data for and from the experiments, and the Matlab code to generate the figures.

The README of TOUCAN-VR references 2 other repositories:

- TOUCAN-VR-parametrizer [16], to parameterize TOUCAN-VR. It is an Android app allowing to choose the network and logging parameters, the content to be streamed and to launch the TOUCAN-VR app.
- TOUCAN-preprocessing [17], to prepare the content to be stored at a regular HTTP server. This java script executes a conversion from a regular 360° video (not yet SRD-described) into a DASH-SRD one.

A.2 Guidelines for testing

Our artifacts can be re-used in two ways:

- The results presented in the article can be reproduced from the data obtained in our user experiments. To do so, download the content of [15] and launch the Matlab script as detailed in the README file. The raw data made of the log files of the experiments are in the `data_from_exp` sub-folder. The content with our editing are described in the `data_for_exp` sub-folder.
- If one wants to use our apps, the Android install process is described in the README of [14] and [16].