360-Degree Video Head Movement Dataset

Xavier Corbillon
IMT Atlantique, France
xavier.corbillon@imt-atlantique.fr

Francesca De Simone
LTS4, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
francesca.desimone@epfl.ch

Gwendal Simon
IMT Atlantique, France
gwendal.simon@imt-atlantique.fr

ABSTRACT

While Virtual Reality applications are increasingly attracting the attention of developers and business analysts, the behaviour of users watching 360-degree (i.e. omnidirectional) videos has not been thoroughly studied yet. This paper introduces a dataset of head movements of users watching 360-degree videos on a Head-Mounted Display (HMD). The dataset includes data collected from 59 users watching five 70 s-long 360-degree videos on the Razer OSVR HDK2 HMD. The selected videos span a wide range of 360-degree content for which different viewer's involvement, thus navigation patterns, could be expected. We describe the open-source software developed to produce the dataset and present the test material and viewing conditions considered during the data acquisition. Finally, we show some examples of statistics that can be extracted from the collected data, for a content-dependent analysis of users' navigation patterns. The source code of the software used to collect the data has been made publicly available, together with the entire dataset, to enable the community to extend the dataset.

CCS CONCEPTS

• **Human-centered computing → Displays and imagers**; • **Computing methodologies → Virtual reality**;

KEYWORDS

360-degree Video, Omnidirectional Video, Head Mounted Display, Virtual Reality, Focus of Attention

ACM Reference format:
Xavier Corbillon, Francesca De Simone, and Gwendal Simon. 2017. 360-Degree Video Head Movement Dataset. In *Proceedings of MMSys'17, Taipei, Taiwan, June 20-23, 2017,* 6 pages.
https://doi.org/http://dx.doi.org/10.1145/3083187.3083215

1 INTRODUCTION

Business and technology specialists predict that immersive applications will become mainstream by 2020 and that 360-degree videos will be one of the main applications for Head-Mounted Display (HMD) [10].

MMSys'17, June 20-23, 2017, Taipei, Taiwan

ACM ISBN 978-1-4503-5002-0/17/06…$15.00
https://doi.org/http://dx.doi.org/10.1145/3083187.3083215

360-degree videos, also called omnidirectional videos, are spherical signals: a user who watches a 360-degree video on a HMD can choose which portion of spherical content to display by moving the head to a specific direction. The portion of spherical surface attended by the user is projected to a segment of plane, called *viewport*.

Video service providers, such as YouTube and Facebook, have recently started to deliver 360-degree videos from their streaming platforms, adapting the visual signal in order to use their existing streaming technologies. Particularly, spherical videos are mapped onto planar videos using sphere-to-plane projections [12] that make 360-degree content compatible with existing file formats, encoders, streaming architectures, and content delivery networks (CDNs).

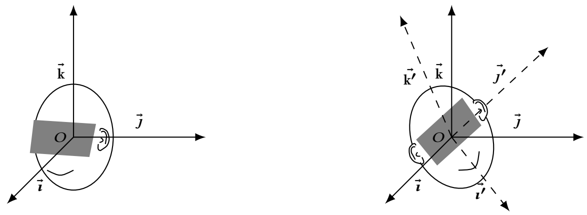
The new generation of delivery systems for 360-degree videos aims at improving the efficiency of the delivery by maximizing the video quality in the viewport and minimizing the waste of bandwidth due to the transmission of parts of the sphere that are never attended by the user, thus displayed. Instead of streaming the entire spherical content at each instant in time, *viewport adaptive streaming* solutions [6, 8, 13, 20] have been proposed, where the streamed content depends not only on the available bandwidth between the client and the server but also on the user's instantaneous viewing direction.

The adoption of viewport-adaptive streaming raises many open questions regarding the navigation patterns of users. For instance: how fast do people move their heads when wearing an HMD? do different people focus on the same parts of the sphere? does the user's behaviour depend on the type of video? and does this behaviour depend on the user's characteristics? In order to answer these questions, the availability of data collected while users are watching 360-degree videos via HMDs is critical.

For this reason, in this paper, we present a 360-degree video head movement dataset gathered by recording the navigation patterns of 59 users watching five 70 s long 360-degree videos. The dataset is available on our website.[1] In order to help the community to reproduce our results and to upgrade this dataset, we release the open source software developed for our data collection campaign and allow new contributors to share their datasets on our website.[1]

The rest of the paper is organized as follow. Section 2 reviews the existing works that have reported an analysis of the navigation patterns of viewers consuming 360-degree content via HMDs. Section 3 describes the choices made to setup our data collection, the software and the experimental conditions. Section 4 details the structure of our dataset so that it can be reused by the community. Section 5 portrays a first analysis of the collected data to illustrate the kind of information that can be extracted from the dataset. Finally, Section 6 concludes this paper.

[1] http://dash.ipv6.enstb.fr/headMovements/



Reference position Current user position

Figure 1: Choice for the stationary reference frame (O, $\vec{\imath}$, $\vec{\jmath}$, $\vec{\mathbf{k}}$) and for the rotating reference frame (O, $\vec{\imath}'$, $\vec{\jmath}'$, $\vec{\mathbf{k}}'$) linked to the user head.

2 RELATED WORK

Yu et al. [19] use the average viewport-based head motion trajectory of ten users viewing ten 360-degree videos to compute a weighted version of the Peak Signal to Noise Ratio. The authors observed that the users tend to look around the equator more than the poles. The average viewing probability map is reported in the paper, but the navigation patterns per user and content are not publicly available. Also, the viewing conditions and test material used to collect the data are not disclosed.

Upenik et al. [17] describe a testbed to perform controlled quality assessment experiments on 360-degree images via HMDs. The software allows to capture, among other data, the viewing direction of the user at a chosen sampling rate. Examples of average viewing probability maps obtained during a quality assessment subjective test are reported for three test images. The method used to process the information on user's head movements and derive saliency maps is described by Upenik and Ebrahimi [16].

De Abreu et al. [7] describe a study of navigation patterns collected during 360-degree image viewing on a HMD. The testbed and collected dataset are publicly available.

To the best of our knowledge, the research community misses a dataset of head movements of users consuming 360-degree videos via HMDs, which is the contribution described in this paper.

3 EXPERIMENTAL SETTINGS

In this Section we describe the notation chosen to describe user's head movements, the software implemented to capture these movements during 360-degree video consumption on a HMD, as well as the test material and conditions considered during the viewing sessions.

3.1 Head Positions

Since current delivery platforms and HMD technologies are restricted to 3-Degrees of Freedom (DOF), we captured rotational head movements and ignored the translational movements of users. To measure the head position we chose the following conventions, illustrated in Figure 1:

- We consider the $\mathbb{R}^3$ Euclidean space with the direct orthonormal basis (O, $\vec{\imath}$, $\vec{\jmath}$, $\vec{\mathbf{k}}$).
- We denote by (O, $\vec{\imath}'$, $\vec{\jmath}'$, $\vec{\mathbf{k}}'$) the direct orthonormal basis linked to the user's head position. The $\vec{\imath}'$ axis goes through the center of the HMD, the $\vec{\jmath}'$ axis goes through the viewer's left ear and the $\vec{\mathbf{k}}'$ axis goes through the top of the head.
- The reference head position (i.e. the position without any rotation) is the position where (O, $\vec{\imath}$, $\vec{\jmath}$, $\vec{\mathbf{k}}$) and (O, $\vec{\imath}'$, $\vec{\jmath}'$, $\vec{\mathbf{k}}'$) coincide. The reference position (i.e. the (O, $\vec{\imath}$, $\vec{\jmath}$, $\vec{\mathbf{k}}$) basis) is set at boot time by the HMD. $\vec{\mathbf{k}}$ is always vertical but $\vec{\imath}$ and $\vec{\jmath}$ can change each time the HMD restarts. Between two reboots the reference position never changes.

Using the software described at Section 3.2, we captured any variation of the head position during a viewing session, with respect to the reference position. This is described by the rotation $\mathscr{R}$ that transforms (O, $\vec{\imath}$, $\vec{\jmath}$, $\vec{\mathbf{k}}$) into (O, $\vec{\imath}'$, $\vec{\jmath}'$, $\vec{\mathbf{k}}'$). There are many ways to characterize a rotation in $\mathbb{R}^3$ [5]: we use the unit Hamiltons quaternions representation. According to Euler's rotation theorem, any rotation or sequence of rotations of a three-dimensional coordinate system with fixed origin is equivalent to a single rotation around an axis, represented by a unit vector $\vec{\mathbf{v}} = (x, y, z) = x\vec{\imath} + y\vec{\jmath} + z\vec{\mathbf{k}}$ in $\mathbb{R}^3$, and by a given angle θ, using the right hand rule. This axis-angle representation of $\mathscr{R}$ can be expressed by four scalars defining the unit quaternion [5]: $q = (q_0, q_1, q_2, q_3) = (q_0, q_1\vec{\imath} + q_2\vec{\jmath} + q_3\vec{\mathbf{k}}) = (cos(\theta/2), sin(\theta/2)\vec{\mathbf{v}})$.

We chose the quaternion representation because (i) it has the advantage of being a compact representation (four scalars instead of the nine required by the 3x3 matrix representation), (ii) quaternion are not subject to the *gimbal lock* [18], which is a well-known issue of the Euler angles representation, and (iii) quaternion representation of rotations is less sensitive than matrix representation to rounding errors occurring when scalars are represented at floating point precision.

3.2 Software

We developed a 360-degree video player to capture and save to a log file the user's head position at each frame or whenever a head movement (*i.e.* a motion event) occurs during the visualization of a video on an HMD. The main purpose of the software is to accurately associate each motion event to a timestamp corresponding to a video timestamp at frame level, thus, to the id of the video frame displayed by the HMD when the motion event occurred.

Please note that we share the software in a public repository[1] with the MIT open source license [1]. Therefore, it can now be used by the scientific community, potentially to extend the dataset with more viewers, more videos, and different viewing conditions.

The software has been implemented in C++, based on the Open-Source Virtual Reality (OSVR) Application Programming Interface (API) [4], the Open Graphics Library (OpenGL) [3], and the *ffmpeg* library [2]. We performed all tests and the data collection campaign on Linux OS with the Razer OSVR Hacker Development Kit 2 (HDK2) HMD [14] but the software is expected to be compatible with any HMD and Windows OS.[2] Figure 2 illustrates a high-level diagram of the input and output interfaces of our software.

[2]The OSVR API is available on Linux and Windows OS and is agnostic to the HMD used.

YouTube Id	Name	Content Description & Expected Focus of Attention	Spatial Resolution	Frame Rate	Bit Rate	Start Offset
2bpICIClAIg	Elephants	Elephants along a river side. Fixed camera, main content along the equator line. One main azimuthal focus expected along the equator line.	3840 × 2048 pixels	30 fps	16522 kbps	15 s
7IWp875pCxQ	Rhinos	Rhinos in the nature. Fixed camera, main content along the equator line. Focus expected along the equator line.	3840 × 2048 pixels	30 fps	13462 kbps	15 s
2OzlksZBTiA	Diving	Diving scene. Slowly moving camera, no clear horizon. No main focus expected within the sphere.	3840 × 2048 pixels	29.97 fps	19604 kbps	40 s
8lsB-P8nGSM	Rollercoaster	Rollercoaster. Fast moving camera fixed in front of a moving roller-coaster. Strong main focus following the rollercoaster trail.	3840 × 2048 pixels	30 fps	16075 kbps	65 s
CIw8R8thnm8	Timelapse	Timelapse of city streets. Fixed camera, clear horizon with a lot of fast moving people/cars, many scene cuts. Focus expected along the equator line.	3840 × 2048 pixels	30 fps	15581 kbps	0 s
s-AJRFQuAtE	Venice	Virtual aerial reconstruction of Venice. Slowly moving camera. No main focus expected within the sphere.	3840 × 2048 pixels	25 fps	16101 kbps	0 s
sJxiPiAaB4k	Paris	Guided tour of Paris. Static camera with some smooth scene cuts. Focus expected along the equator line.	3840 × 2048 pixels	60 fps	14268 kbps	0 s

Table 1: Description of the YouTube 360-degree videos used. The entries with grey background in the table identify the videos used for training.

Number of users	Minimum age	Average age	Maximum age	Ratio of women	Ratio of users using a HMD for the first time
59	6	34.15	62	20%	61%

Table 2: Statistics on the users who took part to the experiment

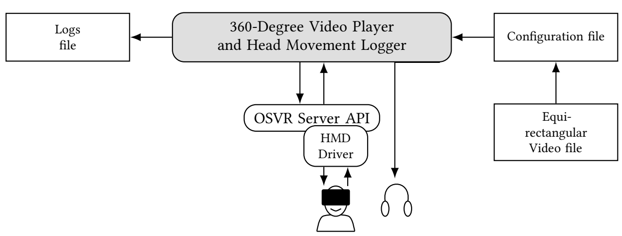


Figure 2: High level diagram of components and interfaces of our OSVR Video Player and head movement logger. The gray block is the components we developed.

The experimenter can set some input parameters by using a configuration file. This file specifies: (i) the sphere to plane projection used to produce the planar video file ; (ii) the time offset in second starting from which the video file is displayed; (iii) the number of video frames, i.e. the duration of the segment, to be displayed. The time offset and duration are specified because the input 360-degree video can be a video of several minutes of duration: the configuration file allows to specify what portion of the video to display to the user, so that the navigation patterns are measured over a fixed limited duration.

The player communicates with the OSVR Server API to get the last known position of the user's head. With this information, the player extracts from the input video the viewports to be displayed (one for each eye) and sends them to the HMD using the OSVR API. In parallel, the audio signal is sent to a headphone. Each time a new head position is measured, the quaternion that represents the rotation of the head with respect to the reference head position is stored in a log file alongside the current timestamp and the picture id of the displayed video frame.

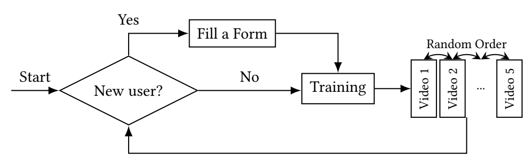


Figure 3: Flow for a new user evaluation

3.3 Test Material

The navigation patterns in the dataset have been collected on five 360-degree YouTube videos, described in Table 1. We limited the test material to few videos to be able to perform a viewing session of reasonable duration and collect data from many users. Therefore, the test videos have been chosen to span a wide range of 360-degree content, including tourism, thrill, and discovery, for which different viewer's involvement, thus navigation patterns, could be expected. Each video file has been downloaded in equirectangular format, at the maximum resolution and bit-rate available on YouTube (reported in Table 1), The videos do not have the same frame rate but this does not impact the data collection since, as detailed in Section 3.2, the head position is captured for each motion event, rather than at a fixed rate. A 70 s-long portion of each video (starting at an offset indicated in Table 1) has been selected and used in the viewing session. Two additional videos (with grey background in Table 1) have been used as training material to familiarise the users with the viewing set-up.

3.4 Viewing Session

All participants to our study performed one viewing session of a total duration of seven minutes. Before the beginning of the session, oral instructions were provided to describe the main steps of the viewing session and explain that the user's navigation patterns were going to be recorded. Each user was informed about the presence of a training session, to familiarize with the 360-degree viewing experience and adjust the HMD calibration, if needed, followed by

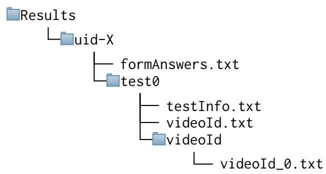


Figure 4: Folder structure

a viewing session of seven minutes, consisting of the sequential display of videos, separated by a grey screen, displayed for 5 s to 10 s between two consecutive videos.

The exact flow (Figure 3), followed by every user, is described hereafter:

1. Before the user puts the HMD on, he/she is asked to fill in a questionnaire, displayed on a PC via a Graphical User Interface (GUI), concerning the user's gender, age, vision impairments if any, and the level of familiarity with HMDs.
2. The training session takes place, during which a 70 s training video, randomly selected for each user by the GUI, is displayed. Viewers were orally instructed to adjust the HMD vision correction settings, if needed, by using the wrench adjuster under the HMD and familiarise with the 360-degree viewing experience by moving their heads.
3. All five test videos are consecutively displayed in a random order.
4. At the end of the viewing session, the user is asked to remove the HMD.

We invited people to stand during the entire viewing session but some asked to sit for some videos (often for the *Rollercoaster*). When seated, people sat on a rolling chair and were still able to turn easily in any direction. An operator always stood next to the user to hold cables out of user's range and guarantee free movements.

3.5 User Sample

At the time this paper was written, 59 users took part to our data collection. Most people in the sample group are students or staffs from the IMT Atlantique school in France. Some are children from staff members and some are employees from IMT Atlantique's startups. Table 2 shows some statistics about this sample group. Users are aged from 6 to 62 with an average age equal to 34 years. 80% of the sample is composed by men and 61% of the sample was using a HMD for the first time. Half of the users who had already used a HMD, did it for less than 12 minutes.

4 DATASET STRUCTURE

The dataset was created and structured to allow other research teams to use it and add new traces, while protecting the privacy of the users. The folder structure, data format and meaning of the results collected by using the software described at Section 3 are detailed hereafter.

4.1 Result Folder Structure

Figure 4 pictures the structure of the result folder. When a new user participates to the data capture, he receives a unique identifier [9] that guarantees there will be no naming collision. The data collected

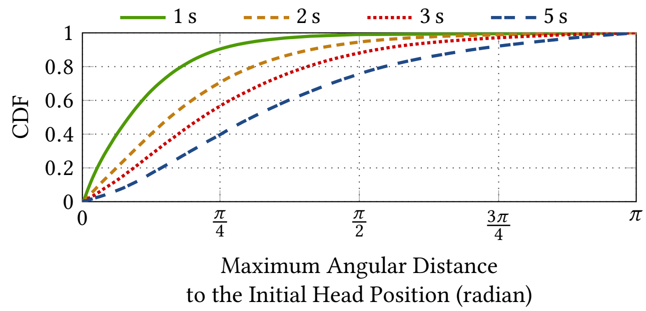


Figure 5: CDF of the maximum angular distance from the head position at the start of the segment for different segment lengths.

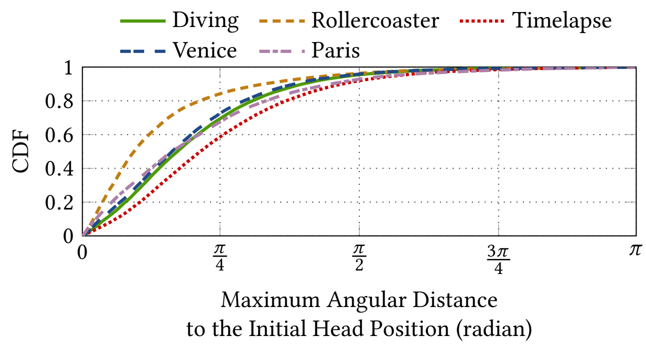
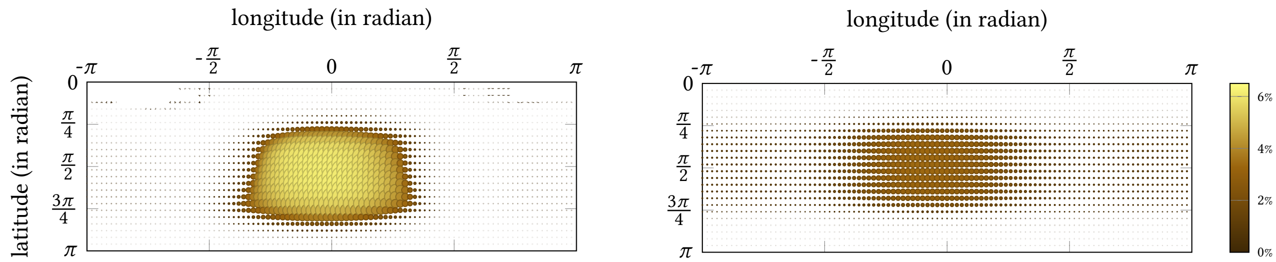


Figure 6: CDF of the maximum angular distance from the head position at the start of the segment, per video, for a 2 s long segment.

for each user is stored in a dedicated folder, named "uid-X", with X being the user's identifier. This folder contains:

- one file named "formAnswers.txt", containing the answers of the user to the questionnaire;
- one folder per viewing session , named "testN", with N being the viewing sessions id associated to the viewing session, generated by the GUI , useful to structure the results when the same user participates to multiple viewing sessions. This folder contains: (i) a file named "testInfo.txt", reporting on each line a video id followed by the MD5 sum [15] of the video file. These identify the videos displayed to the user during the corresponding viewing session test, according to the order of presentation, the first video being the training video; for each video id in the file "testInfo.txt", (ii) a corresponding file named "videoid.txt", reporting the configuration file of the C++ OSVR video player and head movement logger used to displayed this video to the user (*cf.* Section 3.2 for more details about the software), and (iii) a folder named "videoid" containing the user's head movements for this specific video, according to the format detailed in the next subsection.



(a) Roller-Coaster

(b) Timelapse

Figure 7: Probability for a pixel to be be attended by a user during a whole video

4.2 Head Position Log Structure

Each log file of the user's head position has the same structure. There is one sample per line. Values are separated by spaces, according to the following format :

```
timestamp frameId q0 q1 q2 q3
```

The first value, at floating-point precision, is the timestamp in second relative to timestamp 0: timestamp 0 is the time when the video player started to display the first frame and is the first timestamp in the file. The second value, an integer, is the Picture Order Count (POC) of the video frame displayed at time equal the timestamp. Here, POC zero corresponds to the first picture displayed after the player seeks to the start offset of the video. The next four values, at floating-point precision, are the q_0, q_1, q_2 and q_3 values of the unit quaternion q used to identify the head position of the user (*cf.* Section 3.1): $q = (q_0, q_1\vec{\imath} + q_2\vec{\jmath} + q_3\vec{\mathbf{k}})$.

In the log files, head position samples are recorded each time the video player renders a new picture for the HMD. This means the sampling rate is not constant and may vary within each test. Note that rendering refresh rate can be higher than the video frame rate.

5 A TYPICAL USAGE OF THE DATASET

In this Section we illustrate a possible use of the dataset by focusing on the viewport adaptive streaming scenario presented in the Section 1. This analysis does not aim to be exhaustive of all possible usages of the dataset.

5.1 Pre-Processing

The population of users from which we collected the dataset is close to the typical audience expected for viewport adaptive streaming, therefore there is no need to filter the users.

In Section 4.2, we mentioned that the head position sampling rate is not constant. This means that some sample are very close in time and some are farther away. In order to compute head movement statistics, we resampled the collected data, i.e. the quaternions, using a sampling frequency of 30 Hz. We chose 30 Hz because most of the videos used to generate the dataset have a frame rate of 30 fps. There are multiple ways to interpolate quaternions based on existing samples: we chose to use the spherical linear interpolation (SLERP) [11], using the two samples that are the closest in time to the timestamp we want to extract. The SLERP formula assumes that the user moves on the shortest great-circle arc between the two measured positions, with a constant velocity.

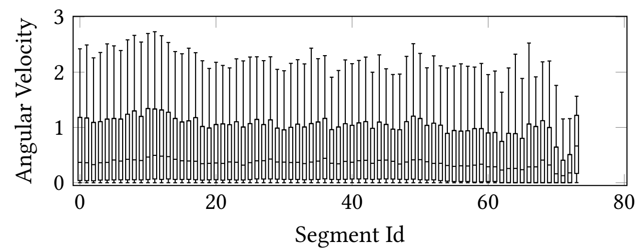


Figure 8: Norm of the angular velocity vector within segments of 1 s long. We expected to see this velocity decrease but that is not what we get.

5.2 Head movements

With adaptive streaming, it is often not possible to switch from video representations once a specific representation started to be displayed to the user, so it is important for the video segment size to be short enough to allow frequent representation switching. On the other hand, if the video segment is too short, video codecs are less efficient and the service provider needs to store and transmit more metadata to describe the segments.

To estimate the maximum duration a video segment should have, we compute "how static the user is". Particularly, for each user, we compute the angular distance between the center of the viewport at the beginning of a segment and the center of each viewport attended by the user during the duration of a segment. Figure 5 shows the cumulative density function (CDF) of the maximum angular distance traveled during a fixed duration inside each segment by each user and for each video. We used segments of length 1 s, 2 s, 3 s, and 5 s. It can be noticed that, for instance, within a segment duration of 2 s, 95% of the users move less that $\pi/2$ radians. This means that within a 2 s length segment, 95% of the user stay inside the hemisphere centered on the head position of the user at the beginning of the segment. Therefore, we conclude that 2 s is probably a good compromise for the duration of the segment.

Figure 6 shows the CDF of the maximum angular distance per video for a segment length of 2 s. The data shows that, in this dataset, there are three categories of videos: a video triggering with very few head movements (Roller-Coaster), a video triggering many head movements (Timelapse), and intermediary videos (Venice and

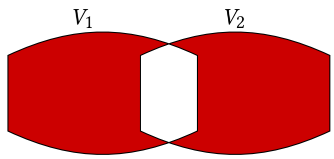


Figure 9: In red the area of the symmetric difference between viewport V_1 and viewport V_2 represented in the equirectangular domain.

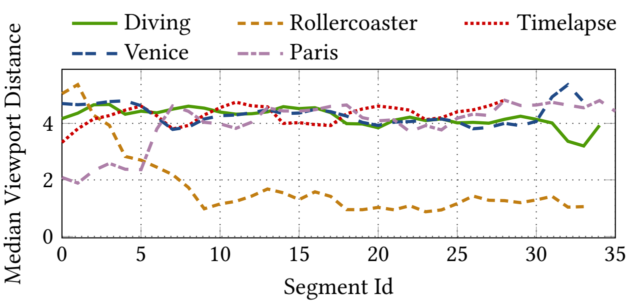


Figure 10: Vision distance for a 2 s long segment on a the Roller-coaster video

Diving). The video Paris is an hybrid, triggering very few head movements at the beginning and many more towards the end.

5.3 Viewing probability

Statistics on where users looked at in a specific 360-degree video can be used to (iv) re-encode representations with Quality Emphasized Regions (QERs), adapted to a majority of users, or (v) send information to the client to help its representation selection decision.

Figure 7 shows the probability for a pixel in the equirectangular domain to be inside the viewport of the user, aggregated for the whole video duration. Figure 7a shows the statistics for the Roller-Coaster video and Figure 7b for the Timelapse video. We observe that for the Roller-coaster video there is a very well defined Region of Interest (RoI) at the center of the equirectangular picture. This is in the direction of the rails. For the Timelapse video, there is no prominent viewing direction, but most viewports stay near the horizon.

To have a better understanding of the time variation of the concentration of the viewports in the videos and similarity across users navigation patterns, we compute for each video frame the area of the symmetric difference between all possible couple of viewports attended by each user during the navigation. Figure 9 depicts the symmetric difference of two viewports. The area of the symmetric difference is a pseudo-distance that is equal to zero when the two viewports are identical and is equal to two time the area of a viewport when the intersection of the two viewports is empty. Figure 10 represents the median distance between all couple of viewports attended by each user inside the same frame during video segments of 2 s. We observe that for the Roller-coaster video, the users focus their gaze in the same direction after 10 s (5 segments of 2 s). For the Timelapse, Venice, and Diving videos, user viewports are spread across multiple directions. Regarding the Paris video, at the beginning of the video, most people look in the same direction (at the tourist guide) and then most users look in different directions.

6 CONCLUSION

In this paper we presented a dataset including the head positions of 59 users recorded while they were watching five 70 s-long 360-degree videos using the Razer OSVR HDK2 HMD [14]. The dataset is available on our website[1] alongside the used videos and the open-source software that we developed to collect the dataset. We described our settings, the test material and how we performed the data collection. We also detialed the structure of the dataset. Finally we introduced examples of statistics that can be extracted from the dataset to provide an overview of the users' behaviour and the videos characteristics, focusing on the viewport adaptive streaming scenario.

We expect that this dataset will help researchers to study and understand 360-degree video consumption. The prediction of navigation patterns is a cornerstone of the new generation viewport-adaptive streaming systems for 360-degree content. The dataset will hopefully enable researchers to test new prediction algorithms.

REFERENCES

[1] MIT open source license. https://opensource.org/licenses/MIT, March 2017.
[2] Libav home page, March 2017. https://libav.org/.
[3] OpenGL home page, March 2017. https://www.opengl.org/.
[4] Y. S. Boger, R. A. Pavlik, and R. M. Taylor. OSVR: An open-source virtual reality platform for both industry and academia. In *Proc. of IEEE Virtual Reality (VR)*, 2015.
[5] S. B. Choe and J. J. Faraway. Modeling head and hand orientation during motion using quaternions. Technical report, SAE Technical Paper, 2004.
[6] X. Corbillon, A. Devlic, G. Simon, and J. Chakareski. Viewport-adaptive navigable 360-degree video delivery. In *IEEE ICC*, 2017.
[7] A. De Abreu, C. Ozcinar, and A. Smolic. Look around you: Saliency maps for omnidirectional images in vr applications. In *Proceedings of IEEE QoMEX*, 2017.
[8] M. Hosseini and V. Swaminathan. Adaptive 360 VR video streaming based on MPEG-DASH SRD. In *Proc. of IEEE Int. Symp. on Multimedia ISM*, 2016.
[9] P. J. Leach, R. Salz, and M. H. Mealling. A Universally Unique IDentifier (UUID) URN Namespace. RFC 4122, July 2005. URL https://rfc-editor.org/rfc/rfc4122.txt.
[10] MPEG Experts. Summary of survey on virtual reality (m16542). ISO/IEC JTC 1/SC 29//WG 11, Oct. 2016.
[11] R. Mukundan. Quaternions: From classical mechanics to computer graphics, and beyond. In *Proc. of the Asian Technology conference in Mathematics*, 2002.
[12] K. Ng, S. Chan, and H. Shum. Data Compression and Transmission Aspects of Panoramic Videos. *IEEE Transactions on Circuits and Systems for Video Technology CSVT*, 15(1):1–15, January 2005.
[13] F. Quan, B. Han, L. Ji, and V. Gopalakrishnan. Optimizing 360 video delivery over cellular networks. In *ACM SIGCOMM AllThingsCellular*, 2016.
[14] C. Quin. The teardown: Razer OSVR HDK2 virtual reality headset. *Engineering & Technology*, 11(10):80–81, 2016.
[15] R. Rivest. The MD5 message-digest algorithm. 1992.
[16] E. Upenik and T. Ebrahimi. A simple method to obtain visual attention data in head mounted virtual reality. In *IEEE Int. Conf. on Multimedia and Expo*, 2017.
[17] E. Upenik, M. Rerabek, and T. Ebrahimi. A Testbed for Subjective Evaluation of Omnidirectional Visual Content. In *32nd Picture Coding Symposium*, number EPFL-CONF-221560, 2016.
[18] J. Vince. *Rotation transforms for computer graphics.* Springer Science & Business Media, 2011.
[19] M. Yu, H. Lakshman, and B. Girod. A framework to evaluate omnidirectional video coding schemes. In *Proc. of IEEE Mixed and Augmented Reality (ISMAR)*, 2015.
[20] A. Zare, A. Aminlou, M. M. Hannuksela, and M. Gabbouj. HEVC-compliant tile-based streaming of panoramic video for virtual reality applications. In *Proc. of ACM Conf. on Multimedia MM*, 2016.