Automatic View Tracking in 360° Multimedia Using xAPI

Antoni Oliver(✉), Javier del Molino, and Antoni Bibiloni

Laboratorio de Tecnologías de la Información Multimedia (LTIM),
Departamento de Matemáticas e Informática,
Universitat de les Illes Balears, Palma, Spain
{antoni.oliver,toni.bibiloni}@uib.es,
j.delmolino1@estudiant.uib.es

Abstract. 360° video can consume up to six times the bandwidth of a regular video by delivering the entire frames instead of just the current viewport, introducing an additional difficulty to the delivery of this kind of multimedia. Many authors address this challenge by narrowing the delivered viewport using knowledge of where the user is likely to look at. To address this, we propose an automatic view tracking system based in xAPI to collect the data required to create the knowledge required to decide the viewport that is going to be delivered. We present a use case of an interactive 360° video documentary around the migratory crisis in Greece. In this case, omnidirectional content is recorded using a 6-camera array, rendered in an equirectangular projection and played later by an HTML5 web application. Interactive hotspots are placed on specific coordinates of the space-time of the production, introducing a connection of the viewer with the story by playing additional multimedia content. The current view and other usage data are recorded and permit us to obtain metrics on the user behavior, like most watched areas, with the goal to obtain that required knowledge.

Keywords: Hypervideo · 360° video · Interactive documentary · Video tiling
View-adaptive streaming · xAPI

1 Introduction

360° videos have greatly grown in popularity recently thanks to the introduction of this kind of multimedia by major video social networks Youtube and Facebook. To consume this kind of videos on a web browser, several types of interfaces can be used to select the desired viewport. Popular choices include a drag interface (using either a pointing device or touches on a screen) or a sensor-enabled device, like a smartphone or a head-mounted display (HMD).

All these interfaces work by selecting a portion of the 360° viewport to be displayed at the users' choice. To be able to display that portion in a reasonable quality, the original video needs to be encoded in a very high resolution. This results in a very large bandwidth usage, of which only a rough 8% is displayed as the selected viewport.

M. J. Abásolo et al. (Eds.): jAUTI 2017, CCIS 813, pp. 117–131, 2018.
https://doi.org/10.1007/978-3-319-90170-1_9

To address this misusage, a number of approaches are suggested in the literature to reduce the delivered amount of information, most of them revolved around trying to predict where the user is going to view at and assign less (if any) bitrate to the area outside of that predicted viewport.

In this paper we introduce a method to automatically record the position at which the user is looking at when watching a 360º video. We demonstrate the feasibility of this approach through a use case of an interactive 360º video documentary that, following the structure of a hypervideo, depicts the situation of the refugees that flee from war and arrive in Greece since 2015.

This section includes a revision of the current state of the art in interactive multimedia, VR videos and how the literature addresses their bandwidth usage. Section 2 reviews previous work on which this method is based, including the tools needed to create and play the 360º documentary and automatically record the viewport watched by the users. Section 3 explains how to collect this data using xAPI, represent it using heatmaps and how we propose to analyze that data to be used to reduce the bandwidth used by this kind of videos. Finally, the conclusion can be found in Sect. 4, where we also talk about the future work.

1.1 State of the Art

A hypervideo is a navigable stream of video that offers the viewer the possibility of choosing a path in the narrative combining several multimedia tracks through spatial and temporal links present in the media [1]. We extend this concept to support 360º media, including links around the sphere of the 360º panorama. Interactive documentaries have been analyzed in [2], encouraging users to explore and navigate the production, rather than simply viewing it.

Bleumers et al. introduce the users' expectations of omnidirectional video in [3]. 360º multimedia belongs to the field of Virtual Reality (VR) [4, 5]. This field has traditionally made use of several gadgets and devices to create an immersive experience, like head-mounted displays (HMD) or the CAVE [6]. Despite this, there is an increasing trend to display 360º video in an interactive scene inside web browsers in desktop and mobile devices, given the recent adoption of this novel medium by popular online video platforms, like YouTube[1] or Facebook[2], that have begun offering immersive 360º video upload and visualization services. This is thanks to the ability to display a 3D scene to project the equirectangular video texture inside the browser itself since the introduction of WebGL[3], in 2011. The current state of 360º video playback is shared between HMD like the Oculus Rift[4] and web-based navigation in video services like the ones mentioned before, featuring a desktop drag-and-drop interface and a sensor-enabled view for smartphones. In the literature,

[1] https://youtube-creators.googleblog.com/2015/03/a-new-way-to-see-and-share-your-world.html.

[2] https://facebook360.fb.com.

[3] https://www.khronos.org/webgl/.

[4] https://www.oculus.com/.

omnidirectional lectures were given using HMD [7], a drag interface for 360º hypervideo is presented in [8], and more recently Augmented Reality has been introduced in smartphones [9].

Regarding the representation of the users' actions inside a 360º video, commercial solutions like Wistia[5] or LiveHEAT[6] use heatmaps to show the area in the 3D space that is watched by the users. Similar techniques are used in the literature [10, 11].

Finally, to reduce the bandwidth consumed when streaming VR videos, current research strives in identifying the viewport that is going to be displayed so that bandwidth can be saved, or additional bit rate can be assigned to the areas that are actually displayed, resulting in a form of viewport-adaptive streaming. It is unthinkable not using some sort of adaptive streaming at this point; most of the initiatives in the literature are based on MPEG-DASH [12]. [13, 14] performed an experiment with a HMD in which they recorded the users' motion. Using neural networks, they demonstrate that motion prediction is feasible within 0.1 and 0.5 s, achieving a reduction of bandwidth of more than 40% and a failure ratio of under 0.1% with a prediction window of 0.2 s. Other authors make use of what is called *frame tiling*, consisting in dividing the omnidirectional frames in a matrix of tiles, which are requested by the client specifically: [15] propose a tiled streaming scheme over cellular networks, achieving theoretical bandwidth savings of up to 80%; [16] present a system based on HMDs that leverages MPEG-DASH SRD [17] to describe the spatial relationship of the tiles in the 360º space; [18] study how the choice of the projection mapping (equirectangular, cubemap, pyramid, dodecahedron), the length of the segments and the number of representations impact the viewport quality, concluding that the cubemap projection, segments of 2 s and between 5 and 7 representations achieves the best quality with a given budget bit rate. Facebook has been researching this topic since 2016: they first proposed [19] the use of a pyramidal projection, that is dependent on the view, to achieve reductions of up to 80% at the expense of creating 30 different representations of the same media. These are pregenerated and stored. A year later, they optimized [20] dynamic streaming decreasing bit rates by up to 92% and decreasing interruptions by more than 80% by introducing offset cubemaps. After that, they seek [21] to further enhance the experience by using a predicted position instead of the current one, using a prediction system based on a heatmap of the most popular zones of every frame. Since this heatmap alone does not suffice to provide an accurate prediction, they create a gravitational predictor from that, creating valleys in a sphere that correspond to most viewed areas. The position of the user is affected by the geography using a physics simulator, resulting in local influence for point of interest (PoI) and the ability to escape of that influence with strong enough kinetic momentum.

[5] https://wistia.com/.

[6] http://www.finwe.mobi/main/liveheat/.

2 The Interactive 360° Documentary

Over a million people have arrived in Greece since 2015. Refugees flee from war horrors and dreadful armed conflicts. Babies, children and elderly people are especially vulnerable. Migrants arrive and will be still arriving while conflicts last.[7]

In collaboration with PROEM-AID[8], a group of emergency professionals who voluntarily help in Greece, the team recorded 360° footage in the refugee camp and, together with additional, regular video content, prepared a use case for a novel format of interactive documentaries, letting the user to explore a 360° video and obtain additional information in the form of audiovisual content at their request.

We propose the following format for describing an interactive experience with a 360° video: a main video track plays as a panorama, allowing the user to pan the scene and look at any direction. This track serves as a common thread for additional media that is linked to specific points in the space-time of the main track. In these moments, a marker is displayed in the scene, linking to a follow-up video of what is seen in the main track.

This format is based on [22], allowing us to automatically record the user behavior (video controls such as play, pause; 360° controls and also interaction controls, in the form of the selection of a point of interest), especially keeping track of the users' view position at any given moment.

In this section we review how this production was created, by recording the raw images in situ and editing them, how these videos are turned into an interactive experience and how the final product is displayed in the viewers' web browsers, by following Fig. 1. We invite the readers to visit the web application at http://ltim.uib.es/proemaid.

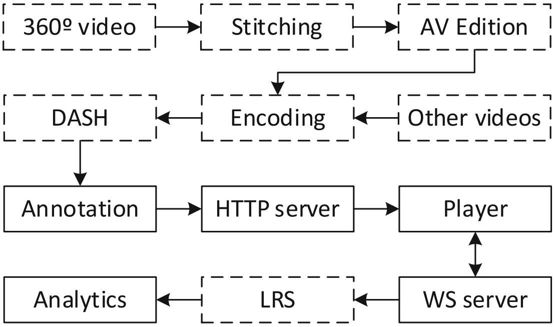

Fig. 1. General overview.

[7] Adapted from http://www.proemaid.org/en/lesbos/.
[8] http://www.proemaid.org/en/.

2.1 Recording and Editing

Two kinds of footage were recorded in the refugee camps in Lesvos and Piraeus, Greece: the 360º media and several linear clips. Four 360º scenes of approximately 90 s each were captured using a 6-camera array, placed on a stick. The media was stitched using Kolor's Autopano Video Pro[9], obtaining an equirectangular clip.

Since both the equirectangular and the linear clips were recorded using high bit rates, not easily supported by most network conditions, they were encoded following the MPEG DASH standard [12] into multiple bit rates. This resulted in several multimedia files and the MPD file for each multimedia file, including the 360º one.

2.2 Introducing the Interactivity

The process of creating the interactive production from the main 360º video track and multiple linear video files is supported by a web application that generates the metadata needed to display the markers.

This annotation tool provides an interactive interface to edit the list of positions that this marker takes in the 360º video. The user is able to navigate the media to select a desired position in space and time to place one of these key positions. This application is designed and built with React[10], and makes use of a scene component to preview and place the key positions of the markers that is also included in the player.

The representation of the positions of a given point of interest is a sequence of points (at least two) in the space-time of the 360º video track, where t is the time in seconds; δ is the latitude in degrees from the equator of the scene; and λ is the longitude in degrees from the meridian zero of the equirectangular frame (see (1) and Fig. 2).

$$\left(t_0, \delta_0, \lambda_0\right), \ldots, \left(t_N, \delta_N, \lambda_N\right) \tag{1}$$

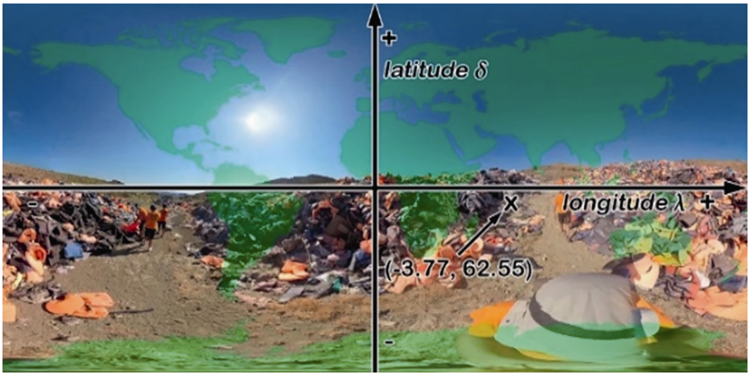

Fig. 2. Notation of latitude and longitude in the equirectangular frame

[9] http://www.kolor.com/autopano-video/.

[10] https://facebook.github.io/react/.

The additional metadata for each point of interest includes a descriptive name, an IRI (Internationalized Resource Identifier) to identify that point, the IRI of the video that should play when selected and an optional image to customize the marker that appears in the scene. This information is stored in a JSON file and retrieved by the player.

2.3 Playing the Production in a Web Browser

Once the interactive 360º video has been defined, it can be played in a WebGL enabled browser. The player is also a web application built with React, and it recycles the same scene component included in the editor to display the 360º video. This time, the extra functionalities of this scene are enabled: the video controls, the marker help area and all events are actually listened (see Fig. 3).

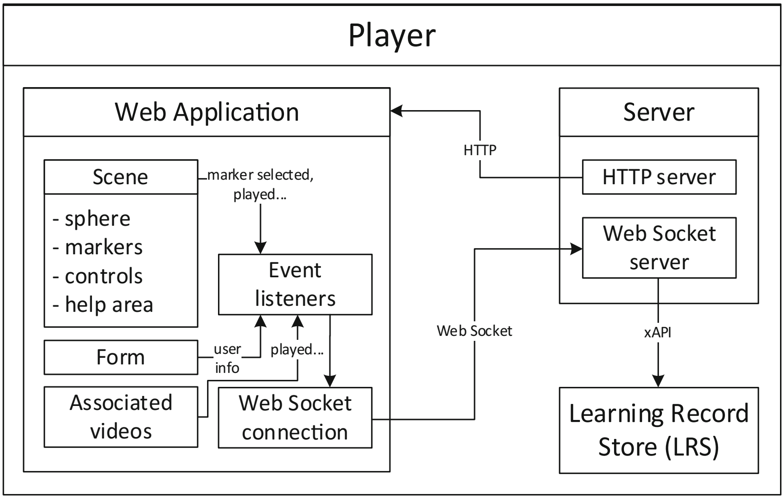

Fig. 3. Architecture of the interactive 360º video player environment.

Before playing the production, a form prompts the user to share some personal information (gender, age, profession, country) with us, so we will be able to obtain some statistics by this information.

By listening these events, the player is notified when a marker is selected, so its associated video will play in a pop-up view, interrupting the main track playback. When this video ends or the user closes the pop-up view, the main playback continues.

A Web Socket connection is established with the server keeping track of these events, which are stored in an LRS thanks to xAPI.

Every video file in this production has been encoded at multiple bit rates. In the player side, the MPD manifests are loaded by the DASH.js[11] JavaScript player and, when the media starts playing, it automatically adapts its bit rate.

[11] https://github.com/Dash-Industry-Forum/dash.js/.

The 3D scene. This component, shared by the editor and the player, houses a WebGL scene managed by Three.js. The core of this scene is the sphere used to display the frames of the 360° texture provided by the main equirectangular video track. This component is provided the list of points of interest, from which circles are created in their respective positions, firing an event when selected. That position is calculated via linear interpolation in spherical coordinates using Ed William's aviation formula[12].

$$f = \frac{t - t_0}{t_1 - t_0}$$

$$d = \cos^{-1}\left[\cos\delta_0 \cos\delta_1 \cos\left(\lambda_0 - \lambda_1\right) + \sin\delta_0 \sin\delta_1\right]$$

$$A = \frac{\sin(1 - f)d}{\sin d}$$

$$B = \frac{\sin f \cdot d}{\sin d}$$

$$p_t = R\begin{bmatrix} A\,\cos\delta_0 \cos\lambda_0 + B\,\cos\delta_1 \cos\lambda_1 \\ A\,\sin\delta_0 + B\,\sin\delta_1 \\ A\,\cos\delta_0 \sin\lambda_0 + B\,\cos\delta_1 \sin\lambda_1 \end{bmatrix} \tag{2}$$

For a particular time t, $t_0 \leq t \leq t_1$, and knowing two key positions at t_0 and t_1, $\left(t_0, \delta_0, \lambda_0\right)$ and $\left(t_1, \delta_1, \lambda_1\right)$, the position p_t in Cartesian coordinates for that marker in that time at a distance of R is obtained via (2).

To be able to detect when a marker is selected, intersections are looked for with a ray casted from the camera position to the selected position. Other events are fired when play conditions change, that will be used by the component that includes the scene, either the editor or the player.

Additional parameters can be passed to the scene to enable two player-specific features: the controls to adjust playback of the 360° video, featuring a progress bar that hints the times when points of interest appear and a help area that displays again the markers that are shown somewhere in the scene, so the user can still see them if they are outside the camera's field of view (see Fig. 4).

Usage tracking. The events fired by the scene are listened by the player and then forwarded via a Web Socket connection to the server. Instead of collecting all the data and submitting it in the end, this approach was chosen to permit us to visualize in real time the users' actions. Once this information is received by the server, it is stored into the LRS via xAPI.

xAPI[13] is a simple but powerful mechanism to store and retrieve logs and share them with any imaginable system. It is a Representational State Transfer (RESTful) API in

[12] http://edwilliams.org/avform.htm#Intermediate.
[13] https://experienceapi.com/.

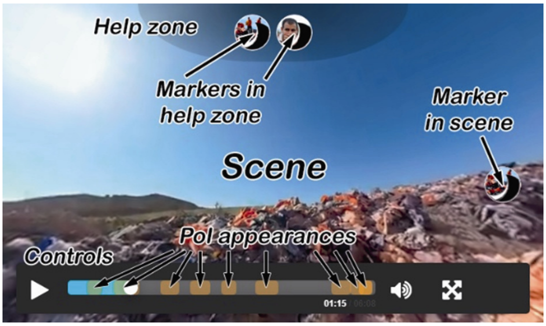

Fig. 4. User interface of the interactive player, featuring the markers in the scene, the help zone with its markers and the controls with its PoI appearances.

which any statement of experience (it can happen anywhere and on any platform) can be tracked as a record. Any kind of experience can be tracked, although xAPI was developed with learning experiences in mind.

A statement is a simple construct, written in JavaScript Object Notation (JSON) format, to store an aspect of an experience consisting of, at a minimum, three parts: an actor, a verb and an object. A set of several statements, each representing an event in time, can be used to track complete details about an experience: its quantity may depend on the need for more or less detail in later reporting.

A Learning Record Store (LRS) is the heart of xAPI, because it receives, stores and returns data about the defined experiences.

The following text extends Table 1, that describes the relation between events and statement verbs. The vocabulary used in this application is based on ADL's video vocabulary [23] and on the general xAPI vocabulary registry [24]. Additional vocabulary has been created and submitted to the general registry.

Table 1. Relation between events and xAPI statement verb IRIs

Event	Verb
The form is submitted	https://w3id.org/xapi/adl/verbs/logged-in
A video track is ready	http://adlnet.gov/expapi/verbs/initialized
A video track is played	https://w3id.org/xapi/video/verbs/played
A video track is paused	https://w3id.org/xapi/video/verbs/paused
A video track is sought	https://w3id.org/xapi/video/verbs/seeked
A video track ends	https://w3id.org/xapi/video/verbs/completed
The main video track is navigated	https://ltim.uib.es/xapi/verbs/lookedat
A marker is selected	http://id.tincanapi.com/verb/selected
The associated video is closed	https://w3id.org/xapi/video/verbs/terminated
The user closes the player	http://id.tincanapi.com/verb/abandoned

The actor for these statements is specified either with using their e-mail address, if provided, or a version 4 UUID as the name of a local account, to deal with anonymous users. The *logged-in* statement is sent to store the users' personal information. Just after that, an *initialized* statement is submitted specifying the main video's IRI, creating a video session from its ID, according to the video vocabulary.

The *played*, *paused*, *seeked* and *completed* statements are used according to the video vocabulary, attaching the following video extensions: *session-id*, *quality*, *screen-size*, *user-agent*, *volume*, *video-playback-size* and *time* (or *time-from* and *time-to*, in the case of a *seeked* statement, refer [23]). An additional extension is defined: https://ltim.uib.es/xapi/extensions/orientation, containing an object with the current view's latitude and longitude.

Following the same design, the *looked at* statement is submitted when the user changes the current view. Successive submissions are throttled to 250 ms so as not to flood the LRS. In this case, *orientation-from* and *orientation-to* are used instead of the *orientation* extension, following the design of the *seeked* statement with *time*.

The *selected* statement is sent to the LRS whenever a point of interest is selected through its marker. This statement includes the additional data present in the *played* statement as context extensions and the IRI of the point of interest as a result extension. From this moment, the associated video window opens, and it begins to load. When it does, an *initialized* statement is generated following the previous behavior, but specifying an object definition type of https://w3id.org/xapi/video/activity-type/video and still using the previous *initialized* statement ID as its *session-id*.

As it is natural, the user is free to interact with the associated video, and *played*, *paused*, *seeked* and *completed* statements will be generated much like previously specified, but with the following differences: the object definition type changes, as in this second *initialized* statement; this second *initialized* statement's ID is used as the *session-id* and the *orientation* extension is unused.

Whenever the user closes the associated video's window, a *terminated* statement is stored in the LRS, including the same additional data these previous statements would. This marks the end of the session of an associated video.

Finally, when the user closes the player, an abandoned statement is submitted, marking the end of the session.

3 Results

After sufficient users have played the 360º video, the LRS contains lots of valuable data that needs to be analyzed; at the time of this writing, over 300 sessions are stored. This section describes the process to retrieve a preliminary set of data from the LRS, the analysis performed on that data and our proposal of actuation from the knowledge gained from this experience. Bokeh[14] version 0.12.13 was used to generate the interactive graphs shown in the following figures.

[14] https://bokeh.pydata.org/en/latest/.

3.1 Data Retrieval

Statements may be fetched from the LRS using standard methods prescribed by xAPI. A big advantage is that we do not need to know how the multimedia player works internally, so the information gets ready in the LRS to be digested at any time by any external system. For complex reporting, all relevant statements can be moved into a data warehouse to be processed later. Although, for simple reporting, metrics can be directly obtained from these statements (this is the approach we have chosen).

It is possible to analyze this information to discover trends, obtain measurements, perform evaluations, comparisons or tracking, or create validation reports on user experiences. A preliminary set of results have been obtained so far from the data collected in the LRS using xAPI, detailed next.

For example, in Fig. 5, two complementary graphs are drawn with data coming from three different 360º video sessions (each session has its own color). They are complementary because the same information is represented in both graphs (with different coordinates). The graph on the left represents playback time versus longitude. Dashed gray lines are superposed to help us know where the markers are displayed in the scene. The graph on the right represents playback time versus relative timestamp: relative to the initialized statement, to be able to coherently compare different sessions.

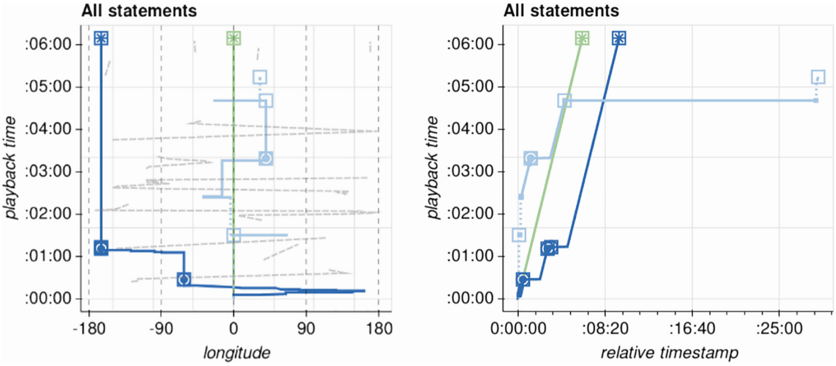

Fig. 5. Statements generated during three sessions. (Color figure online)

Solid lines indicate that the video track was played, was navigated or was paused (not filled squares indicate the moment when the video was paused). Dotted lines indicate that it was sought. Circles indicate the moment when a marker was selected (and an associated video was played, partially at least): filled circles are used the first time the marker was selected, and not filled circles are used the following times. Asterisks indicate the moment when the video was completed. Any session starts at longitude 0º, as the left graph represents.

These graphs allow us to observe, very quickly, how a user has interacted with the video: if they have made left and right movements (the graph on the left tells us exactly where), if they have paused the video (the graph on the right tells us exactly how long), if they have selected markers (and which ones), if they have sought to a new point (the graph on the right tells us if a student has watched all the video content or just made jumps over it until they have reached the end), etc.

For example, in the particular case of Fig. 5:

- The green session corresponds to a user whose attention has not been grabbed by the interactivity offered by the activity: they have only played the 360º video from the beginning, nothing else.
- The dark blue session corresponds to a user that has moved left and right during the first 20 s, until they have played an associated video. Then they have continued watching the 360º video, without interacting with it, during about 40 s, followed by new movements before another associated video has been played twice. No more interactions come later and the 360º video has been watched until the end.
- The pale blue session corresponds to a user not interested in the beginning of the activity: they have directly sought to second 90 (:01:30), where they moved for a while, with the 360º video still paused, followed by a new jump to second 140 (: 02:20), where new movements have been made. Then they have watched the 360º video, without interacting with it, during nearly 60 s, followed by some movements until they have watched an associated video. Later, they have watched the 360º video, without interacting with it, during about 80 s, at which moment they have paused the activity for around 25 min, with some movements in the meantime. Finally, they have sought to second 310 (:05:10) and have skipped the rest of the activity.

3.2 360º Navigation Analysis

We have just shown one of the many possible results that can be obtained from user interactions stored in the LRS, which starts to unveil the potential of this platform.

Heatmaps become another possibility, very useful to show the spatial distribution of statements generated by a subset of users. For this example, in which we are interested in the most frequently selected viewports, we will draw the location distribution of continuous paths: only *played* and *looked at* statements.

This kind of histogram is built from its equivalent playback time versus longitude graph (like the one on the left in Fig. 5), as follows:

1. The plane is divided into a grid of bins (each bin has the same width and height as the rest of the bins).
2. We assign the value 0 to each bin.
3. Solid segments (corresponding to *played* and *looked at* statements in its equivalent playback time versus longitude graph Fig. 5) are sampled following the created grid.
4. We add 1 to the value corresponding to any bin containing a segment.

The graphs in Fig. 6 represent two heatmaps of continuous paths (playback time versus longitude) for 10 independent sessions: the darker the color, the higher the quantity of statements present in the area delimited by each bin. Both graphs represent the same 10 sessions, but the left graph has 9 × 9 bins (each bin has width 40º and height 41 s) and the right graph has 25 × 25 bins (each bin has width 14.4º and height 14.8 s). The number of bins is customizable, of course.

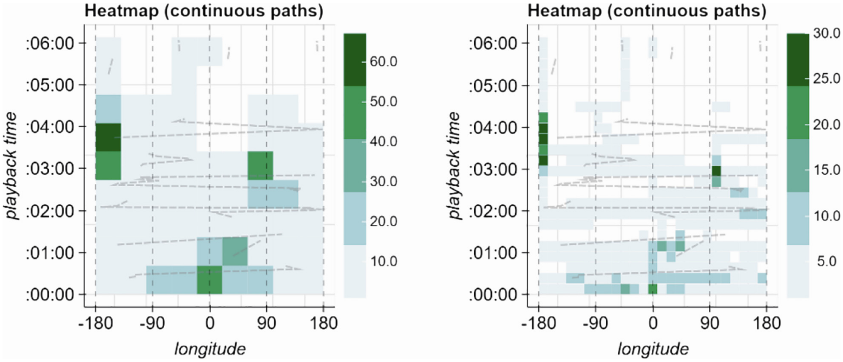

Fig. 6. Heatmaps of continuous paths corresponding to ten sessions.

Dashed gray lines are superposed (in the same way as in the left graph of Fig. 5) denoting the position of markers: we can thus compare how the statements are distributed around selectable objects. The graph on the left can give us a general insight of which zones have been more or less visited by users, whereas the graph on the right is able to refine them more. Depending on the actual need, one or the other could be used (or both of them). As expected, it is clear that around coordinates (0, 0), where the activity always begins, a lot of continuous paths exist.

3.3 Proposal of Actuation

The results of playing the 360° video contribute to create a richer set of data that produces a more accurate heatmap, representing the areas of the scene that are more likely to appear in the viewport when playing the video again.

This information can be exploited, as seen in the literature, to condition the quality at which the video file is encoded. Following the state of the art, the VR video will be encoded in different quality settings for a given budget bit rate. Currently, the video is encoded in 4 different bit rates and the source to be played is chosen following the MPEG DASH specification. We propose to still have a number of bit rate configurations available, but a number of different representations of each of these will be created. These representations will be encoded assigning more bit rate to a particular area of the 360° scene, using one of the approaches suggested in the literature: equirectangular tiling, pyramidal mapping, offset cube maps, etc. To be able to play this complex video structure, modifications will be applied to the MPEG-DASH player that not only will choose a version according to the available bit rate, but also a representation according to the predicted position the user is going to center their viewport during the next period, so that predicted position is adjusted again, and a new representation is fetched.

4 Conclusion

The work presented in this document is an extension of the state of the art in interactive documentaries, introducing user-navigable 360º video and interaction in the form of additional videos that give further details about a specific topic. A use case was conducted with 360º and linear video footage in the refugee camps in Lesvos and Piraeus, which resulted in an interactive web application available to the public. Precise user interactions, including the tracking of the viewport in the 360º space, are logged in real time and analyzed to comprehend their decisions and navigation choices in this novel format. The web application is accessible at http://ltim.uib.es/proemaid.

Knowing (almost) everything the user does with the application enables us to perform very accurate monitoring of their behavior. Simply by observing the proposed graphs we can detected, in a moment, in what proportion users watch the 360º video, both in time and in position: if a video is linearly played or if users like to move inside the 360º environment (and to what extent: if they navigate around markers or if they navigate randomly).

Being able to obtain the information related to the selection of viewport performed by a preliminary set of data permitted us to create visualizations of that choice in the space-time of the 360º production, advancing us to our goal of reducing the bandwidth wasted by this kind of multimedia applications. The method based on xAPI we explained in this paper and previously introduced in [22] resulted adequate for the task and we expect the information it will generate will guide us into our objective. xAPI not only completely fulfills our logging needs, with the help of the LRS, but it also allows that different systems (not even developed by us) are able to talk to each other, thanks to its open interface.

Further enhancements are planned for the system, designing a mobile specific interface, using the device sensors and enabling the use of VR technologies like Oculus Rift and even Google Cardboard. It will be interesting to assess whether different interfaces need similar solutions to address their bandwidth usage.

We expect that we can achieve a significant bandwidth reduction by implementing a view-adaptive system based on the state of the art and refining its parameters for different user interfaces with the help of the automatic viewport tracking system we demonstrated in this paper.

Acknowledgements. The authors want to thank the people that appear in the documentary for sharing their stories with us. We also want to thank PROEM-AID for their work where they are much needed, and especially Manuel Elviro Vidal for recording the scenes shown in the documentary *in situ*. A. Oliver was supported by a predoctoral contract with the Universitat de les Illes Balears (UIB).

References

1. Sawhney, N., Balcom, D., Smith, I.: HyperCafe. In: Proceedings of the Seventh ACM Conference on Hypertext – HYPERTEXT 1996, pp. 1–10. ACM Press, New York (1996)

2. Gaudenzi, S.: The living documentary: from representing reality to co-creating reality in digital interactive documentary (2013)
3. Bleumers, L., Van den Broeck, W., Lievens, B., Pierson, J.: Seeing the bigger picture. In: Proceedings of the 10th European Conference on Interactive TV and Video – EuroiTV 2012, p. 115 (2012)
4. Sherman, W.R., Craig, A.B.: Understanding Virtual Reality: Interface, Application, and Design. Morgan Kaufmann, Los Altos (2003)
5. Neumann, U., Pintaric, T., Rizzo, A.: Immersive panoramic video. In: Proceedings of the Eighth ACM International Conference on Multimedia - MULTIMEDIA 2000, pp. 493–494. ACM Press, New York (2000)
6. Lantz, E.: The future of virtual reality. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques – SIGGRAPH 1996, pp. 485–486. ACM Press, New York (1996)
7. Kavanagh, S., Luxton-Reilly, A., Wüensche, B., Plimmer, B.: Creating 360° educational video. In: Proceedings of the 28th Australian Conference on Computer-Human Interaction – OzCHI 2016, pp. 34–39. ACM Press, New York (2016)
8. Neng, L.A.R., Chambel, T.: Get around 360° hypervideo. In: Proceedings of the 14th International Academic MindTrek Conference on Envisioning Future Media Environments – MindTrek 2010, p. 119. ACM Press, New York (2010)
9. Berning, M., Yonezawa, T., Riedel, T., Nakazawa, J., Beigl, M., Tokuda, H.: pARnorama. In: Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication – UbiComp 2013 Adjunct, pp. 1471–1474. ACM Press, New York (2013)
10. Kasahara, S., Nagai, S., Rekimoto, J.: JackIn Head: immersive visual telepresence system with omnidirectional wearable camera. IEEE Trans. Vis. Comput. Graph. **23**, 1222–1234 (2017)
11. Williamson, J.R., Sundén, D., Bradley, J.: GlobalFestival: evaluating real world interaction on a spherical display. In: Proceedings of the Joint International Conference on Pervasive and Ubiquitous Computing, International Symposium on Wearable Computers, pp. 1251–1261 (2015)
12. Sodagar, I.: The MPEG-DASH standard for multimedia streaming over the internet. IEEE Multimed. **18**, 62–67 (2011)
13. Bao, Y., Wu, H., Zhang, T., Ramli, A.A., Liu, X.: Shooting a moving target: motion-prediction-based transmission for 360-degree videos. In: Joshi, J., Karypis, G., Liu, L., Hu, X., Ak, R., Xia, Y., Xu, W., Sato, A.H., Rachuri, S., Ungar, L., Yu, P.S., Govindaraju, R., Suzumura, T. (eds.) 2016 IEEE International Conference on Big Data (Big Data), pp. 1161–1170. IEEE, New York (2016)
14. Bao, Y., Wu, H., Ramli, A.A., Wang, B., Liu, X.: Viewing 360 degree videos: motion prediction and bandwidth optimization. In: 2016 IEEE 24th International Conference on Network Protocols (ICNP) (2016)
15. Qian, F., Ji, L., Han, B., Gopalakrishnan, V.: Optimizing 360 video delivery over cellular networks. In: Proceedings of the 5th Workshop on All Things Cellular Operations, Applications and Challenges – ATC 2016, pp. 1–6. ACM Press, New York (2016)
16. Hosseini, M., Swaminathan, V.: Adaptive 360 VR video streaming: divide and conquer (2016)
17. Niamut, O.A., Thomas, E., D'Acunto, L., Concolato, C., Denoual, F., Lim, S.Y.: MPEG DASH SRD. In: Proceedings of the 7th International Conference on Multimedia Systems – MMSys 2016, pp. 1–8. ACM Press, New York (2016)

18. Corbillon, X., Simon, G., Devlic, A., Chakareski, J.: Viewport-adaptive navigable 360-degree video delivery. In: 2017 IEEE International Conference on Communications (ICC), pp. 1–7. IEEE (2017)
19. Kuzyakov, E., Pio, D.: Next-generation video encoding techniques for 360 video and VR. https://code.facebook.com/posts/1126354007399553/next-generation-video-encoding-techniques-for-360-video-and-vr/
20. Kuzyakov, E.: End-to-end optimizations for dynamic streaming. https://code.facebook.com/posts/637561796428084/end-to-end-optimizations-for-dynamic-streaming
21. Kuzyakov, E., Chen, S., Peng, R.: Enhancing high-resolution 360 streaming with view prediction. https://code.facebook.com/posts/118926451990297/enhancing-high-resolution-360-streaming-with-view-prediction/
22. Bibiloni, T., Oliver, A., del Molino, J.: Automatic collection of user behavior in 360° multimedia. Multimed. Tools Appl. 1–18 (2017). https://doi.org/10.1007/s11042-017-5510-3
23. Experience API (xAPI) - Video Vocabulary. http://xapi.vocab.pub/datasets/video/
24. The Registry. https://registry.tincanapi.com/