A Second-Screen Meets Hypervideo, Delivering Content Through HbbTV

Toni Bibiloni

Universitat de les Illes Balears

Departamento de Matemáticas e Informática

Laboratorio de Tecnologías de la Información Multimedia, LTIM

Palma, Illes Balears, Spain

toni.bibiloni@uib.es

Miquel Mascaró

Universitat de les Illes Balears

Departamento de Matemáticas e Informática

Laboratorio de Tecnologías de la Información Multimedia, LTIM

Palma, Illes Balears, Spain

mascport@uib.es

Pere Palmer

Universitat de les Illes Balears

Departamento de Matemáticas e Informática

Laboratorio de Tecnologías de la Información Multimedia, LTIM

Palma, Illes Balears, Spain

pere.palmer@uib.es

Antoni Oliver

Universitat de les Illes Balears

Departamento de Matemáticas e Informática

Laboratorio de Tecnologías de la Información Multimedia, LTIM

Palma, Illes Balears, Spain

antoni.oliver@uib.es

Abstract

In this paper two improvements for the Hypervideo platform, used to represent augmented reality on Interactive TVs thanks to the hypervideo concept, are presented: the introduction of a second-screen application to the platform, enabling the user to obtain the additional information on its handheld device and delivering the video track through the broadcast channel, thanks to the HbbTV capability.

Author Keywords

Hypervideo: Augmented Reality; HbbTV; Second-Screen Application.

ACM Classification Keywords

H.5.1 Information interfaces and presentation (e.g., HCI): Multimedia information systems—Artificial, augmented, and virtual realities.

H.5.4 Information interfaces and presentation (e.g., HCI): Hypertext/Hypermedia—Architectures, Navigation.

Introduction

A hypervideo, or "video with hyperlinks" [1] is an interactive video stream in which the user is able to interact with the content through hyperlinks, leading to non-linear navigation, searching, sequence skipping,
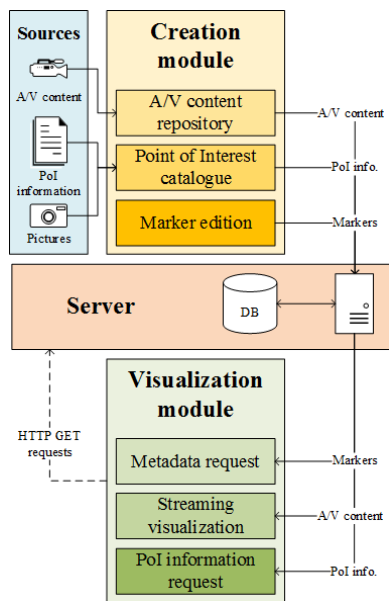
etc. with the purpose of improving the access to the information and bringing the viewer from a passive to an active state [2].

When the hypervideo concept is applied to real images recorded in a video product, augmented reality can be experienced, when this indirect view of the real world is combined with virtual elements, creating a mixed reality.

This paper follows the work from previous papers [3, 4] and demo [5], where a hypervideo platform capable of creating and delivering an AR experience to the viewer through current generation Interactive TV solutions, such as HbbTV, Android TV or Samsung Smart TV was presented.

Two of the improvements proposed in the testing of that prototype are the object of study in this paper: enabling a second-screen device to connect to the application in order to get the additional information in a handheld device and using the capability of HbbTV of representing the broadcast video.

In the first section, the previous work is reviewed to introduce the platform to the reader. Then the second-screen companion application is introduced in the hypervideo platform. Finally, we discuss how this improvement enables the platform to deliver the video through the broadcast channel.

Previous work
The hypervideo solution
The hypervideo format chosen in this project has three dimensions:

- An audiovisual track, which represents the PoIs and is the base for the whole product. In previous work it was delivered via streaming and now is intended to be played through the broadcast channel.
- The points of interest (PoIs), plus their additional information, which can be textual, visual and complementary.
- The markers that represent these PoIs on the video track that enable the user to identify them (*hot-spot* role) and access its additional information (hyperlink role).

This format was proven to be understood by audiovisual producers in previous work and was used to create functional demos.

The hypervideo platform
The hypervideos are created and viewed thanks to the hypervideo platform. The proposed architecture for the platform is shown in Figure 1, consisting in two modules which interact with a server in the middle.

The creation module comprises the tools needed to create a hypervideo, starting by managing the audiovisual repository and inserting new data in the PoI catalogue.

Once these steps have been completed, PoIs are linked to the media through the positioning of the markers that represent them with the aid of an interactive tool.

This module has been developed as a Javascript web application hosted in the server.

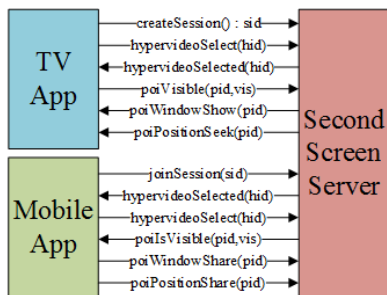The visualization module is a hypervideo player application, with the ability to playback the audiovisual

Figure 1. Diagram of the architecture of the hypervideo platform

track, represent the markers over it and show the additional information of the PoIs requested by the viewer.

A multiplatform development has been followed, being implemented in HbbTV, Android TV and Samsung Smart TV technologies, using HTML and Javascript.

Finally, the hypervideo server is the agent between both modules. It serves the creation module as a web application, and the creation module for the HbbTV and Android TV platforms, as they use a web-based approach.

The server also stores all the data related to the hypervideos and handles the requests from the modules.

Second-Screen companion application

One of the most commented future work proposals in the previous work was the inclusion of a second-screen companion application to enable the user to access the additional information in a separate, personal device.

As stated in [6], a common trend among Interactive TV applications is the introduction of a second-screen companion application. In [7] several insights from the perspective of both the user and the creators of this type of application.

The reasons to introduce a second-screen companion app in the hypervideo platform were:

- The need to pause the video playback to obtain the additional information breaks the intended storytelling and forces other viewers to pause their

experience, having or not interest in the PoI which additional information is being obtained.

- Displaying the additional information on the same screen as the audiovisual track, hiding it, can distract the viewer from the video track.

Having a second-screen companion application, many features related to obtaining additional information from the PoIs can now be carried within the personal device, such as:

- selecting the hypervideo to playback;

- selecting a PoI to obtain additional information from it, whether it is visible at the moment or not;

- navigating that additional information, including text and pictures;

- accessing the complementary information of the PoI, such as their website or location within a map;

- sharing the additional information of the PoI to social networks;

- marking a PoI (and its related information) as "read later", so the user can focus on the audiovisual content;

- sending the additional information to other second-screen devices or to the TV application, so as to share the experience to the other viewers;

- seeking the audiovisual content to the moment when a PoI appears.

To enable the communication between the visualization module and the new second-screen application, a few key changes were made to the first and a Second Screen Server was introduced to intermediate between the second-screen application and the visualization module.
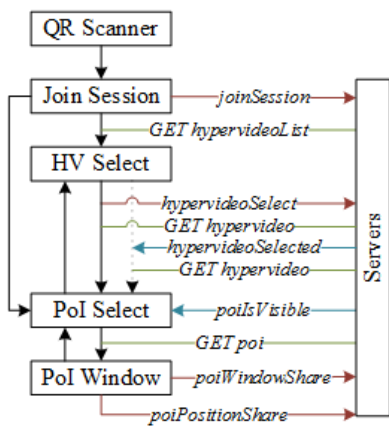

Figure 2. Diagram of the communication protocol between the visualization module, the second-screen app and the Second Screen Server

Figure 3. Detailed diagram of the relation between second-screen app states and communication protocol

Communication protocol

Many interactive TV platforms have begun to propose their own second-screen framework, such as Samsung's [8] or LG's [9]. Besides, HbbTV 2.0 will include support for second-screen apps [10]. To avoid this mix of different techniques and frameworks, a simple, HTML5-based communication protocol is proposed using the WebSocket protocol [11], already implemented in most of Interactive TVs.

The purpose of the Second Screen Server is to act as the target of the communication by both clients, TV application and mobile app. It runs a socket server implemented in Javascript, using the Node.js runtime [12] in a Linux environment, thanks to einaro's ws WebSocket server implementation for Node.js [13].

The purpose of this server is to coordinate events between the TV application and the mobile applications associated to a single TV application. Therefore, it waits for a TV application to start a session, creating a unique session ID (sid), used by the second-screen applications to join it.

Once the link between TV and mobile application is done, the Second Screen Server bounces the events shown in Figure 2 from one application to the other(s), taking into account that more than one mobile application can simultaneously connected to a single TV application.

Changes made to the visualization module

A new second-screen option is shown in the visualization module on the TV. Once the user selects it, the TV application creates a WebSocket connection and requests a session ID to the Second Screen Server

and displays it encoded in a QR-code, to be scanned by the second-screen device(s), joining the session started by the TV application.

The TV application can fire these events:

- *createSession*, explained above;
- *hypervideoSelect*, fired when a hypervideo is selected to be played in the Visualization Module;
- *poiVisible*, fired when a PoI becomes visible or it leaves the screen.

The TV application listens for these events:

- *hypervideoSelected*, making the Visualization Module change the hypervideo being played;
- *poiWindowShow*, making the application display the additional information of a PoI;
- *poiPositionSeek*, making the TV application seek the audiovisual content to the next occurrence of a given PoI.

The second-screen companion app

The companion app is being developed as an HTML5 application, available through URL or inside a native app for Android devices (smartphones and tablets).

Following the same idea as the previous application, the companion app syncs with the TV app thanks to the Second Screen Server, and gets the needed hypervideo data from the Hypervideo Server. In Figure 3, sync messages sent from the app are shown as a right arrow, sync messages sent to the app as a left arrow and data requests without any arrow.

The application has a QR-code scanner built in, that enables it to join a session with the TV (scanning the

code outside the app would trigger opening it), sending the *joinSession* command to the Second Screen Server.

After doing that, the companion app gets the hypervideo ID being currently played (if any). Depending on this, the hypervideo selection window (no hypervideo is being played) or the PoI selection window (a hypervideo is being played) are displayed in the companion app.

In the hypervideo selection window, the list of hypervideos is received from the Hypervideo Server and displayed, allowing the user to select a hypervideo. When doing so, the event *hypervideoSelect* is fired and the data related to that hypervideo is requested. If another user selects a hypervideo, the event *hypervideoSelected* is received and the hypervideo data is requested too.

In the PoI Select state, the hypervideo is being played in the TV application, and the event *poiIsVisible* is received when a PoI enters or leaves the screen, as explained in the previous section.

The user can choose any PoI in the list, visible at that time or not, and request their additional information to the Hypervideo Server.

Finally, in the PoI Window state, the user can browse the additional information of a given PoI, as shown in Figure 4, and access its web page and location through the native apps in their device. Additionally, the user can emit two messages to the Second Screen Server: *poiWindowShare*, to bring that additional information window to the TV screen, and *poiPositionShare*, to

make the visualization module seek for the next appearance of that PoI in the media played.

Finally, in the PoI Window state, the user can browse the additional information of a given PoI, and access its web page and location through the native apps in their device. Additionally, the user can emit two messages to the Second Screen Server: *poiWindowShare*, to bring that additional information window to the TV screen, and *poiPositionShare*, to make the visualization module seek for the next appearance of that PoI in the media played.

Content delivery through HbbTV

As introduced before, HbbTV is one of the present-day Interactive TV technologies to develop applications on. An interesting feature offered by this technology is the ability to interact with the broadcast emission.

This feature enables the Hypervideo platform to deliver the audiovisual track through the broadcast channel, instead of via streaming through the broadband channel.

The main obstacle of doing so was the need to pause the video in order to select a PoI and get their additional information, but with the introduction of the second-screen application, there is no need of doing so.

When playing the audiovisual track through the broadcast channel it is possible that the user starts the application in the middle of the program, so a synchronization mechanism is needed. This can be achieved thanks to the introduction of DSM-CC Stream Events to the HbbTV specification [14], section 8.2.1.

Figure 4. Prototype of the second-screen application, showing the additional information of a Point of Interest

This way, the hypervideo ID and current position in time can be transmitted to the visualization module.

Conclusion
Two improvements for the Hypervideo platform have been introduced, which are still under development.

The addition of a second-screen application to the Hypervideo platform introduces a whole set of new opportunities to the user, permitting multiple viewers to obtain different information simultaneously.

A "faire mode" is being studied, so as not to permit the viewers to interfere each other by changing the video, or seeking the video track: they would only be able to obtain additional information in their devices.

To obtain additional information when playing audiovisual content through the broadcast channel, it could not be paused, and another technique was needed. The second-screen application proves to be adequate for this purpose.

Acknowledgement

References
1. Shawhney, N., Balcom, D., Smith, I. Hypercafe: Narrative and Aesthetic Properties of Hypervideo. In *Proc.* Hypertext *1996*, ACM (1996), 1-10.

2. Landow, G., Kahn, P. Where's the Hypertext? The Dickens Web as a System-Independent Hypertext. In *Proc. Hypertext* 1992, ACM (1992), 149-160.

3. Bibiloni, T., Mascaró, M., Palmer, P., Oliver, A. Realidad Aumentada en HbbTV: Implementación de una plataforma Hypervideo para la Televisión Digital Conectada. In Proc. CISTI 2014, AISTI (2014), 743-748.

4. Bibiloni, T., Mascaró, M., Palmer, P., Oliver, A. Hypervideo: Augmented Reality on Interactive TV. In Proc. jAUTI 2014, ACM (2014).

5. Bibiloni, T., Oliver, A. Augmented Reality on HbbTV: An Hypervideo approach. In Demo Proc. TVX 2014, ACM (2014).

6. Courtois, C., D'heer, E. Second screen applications and tablet users: constellation, awareness, experience and interest. In Proc. EuroITV 2012, ACM (2012), 153-156.

7. Geerts, D., Leenheer, R., De Grooff, D., Negenman, J., Heijstraten, S. In Front of and Behind the Second Screen: Viewer and Producer Perspectives on a Companion App. In Proc. TVX 2014, ACM (2014), 95-102.

8. Samsung. Multi-Screen SDK. http://www.samsungdforum.com/Guide/?FolderName=d30&FileName=index.html

9. LG. Connect SDK. http://connectsdk.com

10. HbbTV Consortium. HbbTV 2.0 FAQ. https://www.hbbtv.org/pages/news_events/pdf/HbTV%202.0%20FAQ.pdf

11. Fette, I., Melnikov, A. The WebSocket Protocol. RFC 6455. IETF (2011).

12. Joyent, Inc. Node.js. http://nodejs.org

13. Stangvik, E. ws: a node.js websocket implementation. http://einaros.github.io/ws

14. ETSI. TS 102 796 v1.1.1. Hybrid Broadcast Broadband TV. http://www.etsi.org/deliver/etsi_ts/102700_102799/102796/01.01.01_60/ts_102796v010101p.pdf