Hypervideo, Augmented Reality on Interactive TV

Toni Bibiloni, Miquel Mascaró, Pere Palmer, Antoni Oliver
Universitat de les Illes Balears – Departamento de Matemáticas e Informática
Laboratorio de Tecnologías de la Información Multimedia, LTIM
Palma de Mallorca, Illes Balears, España
{toni.bibiloni, mascport, pere.palmer, antoni.oliver}@uib.es

ABSTRACT

In this paper, an Augmented Reality system for the Interactive and Connected TV is presented through the implementation of a Hypervido platform. This platform consists of two modules that enable editors and viewers to enjoy an AR experience on current generation Interactive TVs.

Two modules are introduced: the first provides the producers tools to manage the audiovisual content and points of interest, while the other is used by the viewers, in order to play the audiovisual production and obtain additional information about the points of interest that appear on the video.

This work presents an innovative way to mix these three technological concepts: interactive video, augmented reality and connected TV. The paper concludes with some improvement possibilities and extensions.

Categories and Subject Descriptors

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.

General Terms

Design, Experimentation.

Keywords

Augmented Reality; HbbTV; Android TV; Smart TV; Hypervideo

1. INTRODUCTION

A Hypervideo, or "interactive video" [1], is defined as an audiovisual content stream that is offered to the user with a non-linear navigation. The viewer is able to interact with the content through hyperlinks, which are complemented with other mechanisms, such as searching, additional information, sequence or content skipping, etc. all focused to improve the access to the information and with the goal to bring the viewer from a passive to an active state [2].

When mixing the hypervideo concept with real images, we are approaching to augmented reality (AR) applications. AR is the term used to define a direct or indirect vision of the real world, whose elements are combined with virtual elements to create a mixed reality.

In this paper, the development of an interactive video platform, the Hypervideo Platform, is presented. It is capable to deliver an AR experience to the viewer, through current generation Interactive TV solutions, such as HbbTV [3], Android TV [4] or Samsung Smart TV [5].

In the following chapters, the Hypervideo structure is presented, and the process needed to create and view a Hypervideo is shown, as well as the modules developed to enable that are also described in detail. Finally, this paper ends proposing some future work ideas, and the conclusions extracted from doing this work are presented.

2. TECHNOLOGICAL SITUATION

One of the first implementations performed of a hypervideo was in [6], where the links between the scenes of the video let the spectator choose the scene change. In [7], [8], a change in the link behavior is proposed: these can be used to obtain additional information about the content being played at that moment, taking a step towards AR.

In [9] the design and evaluation of "Hvet", a hypervideo environment for teaching veterinary surgery, are presented, which conclusions show the potential of these kind of environments in education systems.

The LinkedTV project [10] suggests a hypervideo platform based on industry web and broadcast specifications (HTML, HbbTV), assimilating additional information automatically accessing the data available in the WWW through LinkedData.

The current trend is the interactive media, as interactive image services show, such as Thinglink [11], or interactive video, such as Wirewax [12], although only a few technological platforms that aimed for interactive audiovisual content creation could be detected.

This situation and the lack of hypervideo productions with educative, promotional or informative purposes motivated the creation of this project: specify and develop a platform destined to interactive audiovisual content production, distribution and visualization on current generation TV technologies.

3. THE HYPERVIDEO SOLUTION

In this project, a hypervideo solution based on video streaming has been chosen, making use of a triple reality:

- An audiovisual track.
- A collection of points of interest.
- The markers that represent these points of interest over the audiovisual track.

3.1 The audiovisual track

First, the real world vision is represented in an indirect way through the visualization via streaming of audiovisual content through the Internet. This video track drives the user through the points of interest.

The audiovisual content can be made on purpose as well as an existing product can be used, taking into account that it has to represent adequately the points of interest.

3.2 The collection of points of interest

Next, the additional information shown in this hypervideo matches the data collected from the points of interest (PoI) represented in the video track:

- Textual information: name and description.
- Typological information: category.
- Visual information: pictures.
- Complementary information: web page, GPS location.

All this data should be gathered before starting the hypervideo creation, in order to assure a better organization.

3.3 The markers

Finally, the hyperlinks of our hypervideo approach are also called *markers*. They are always associated with a point of interest and they accomplish a double function:

- As markers or *hot-spots*, they indicate the point on the screen where a point of interest appears.
- As hyperlinks, they enable the user to browse to the additional information related to a certain point of interest.

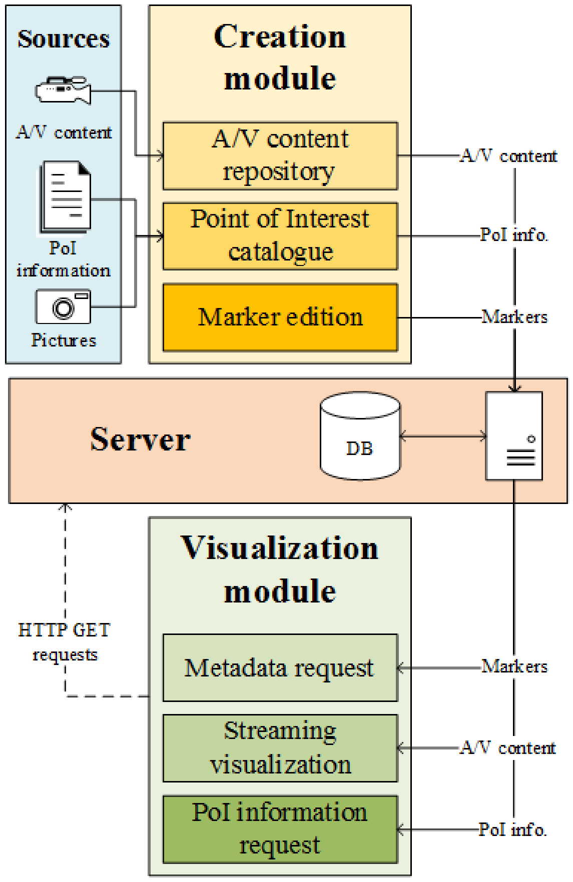


Figure 1. Hypervideo platform general architecture diagram.

4. HYPERVIDEO PLATFORM

In this section, the hypervideo content creation, publishing and visualization platform architecture is presented. As shown in Figure 1, the proposed architecture is composed of two modules that interact with a server in the middle, which stores and serves the needed data to create and play hypervideo productions.

These two modules are described below, and will be further explained in the following sections. In this section, the data model used by the platform is presented, as well as the communications of these modules with the hypervideo server.

4.1 Creation module

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

Once these steps have been completed, the spatiotemporal metadata needed to link the PoIs identified in the media with their markers position is generated.

4.2 Visualization module

The visualization module is represented by a hypervideo player application, which is able to playback the video track via streaming, represent the markers over it and show the additional information of the chosen points of interest.

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

4.3 Data model

The information used by the platform –audiovisual content, points of interest and markers– is stored in a MySQL database, following the data model shown in Figure 2, detailed below:

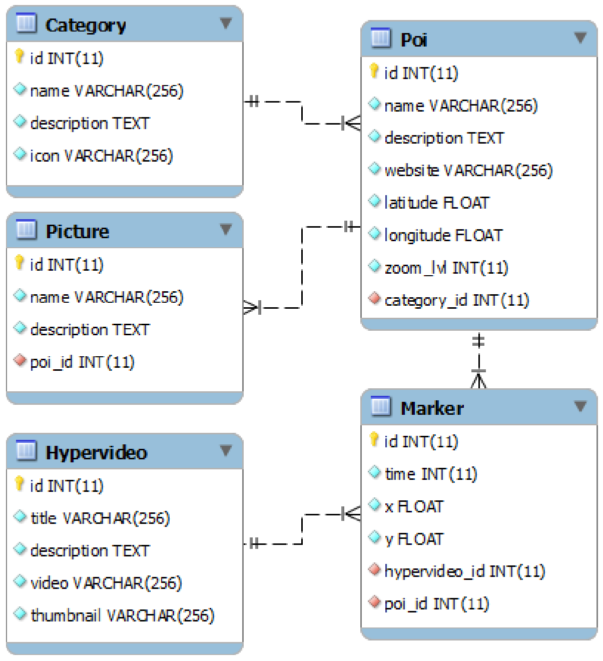


Figure 2. Hypervideo platform data model.

The *Hypervideo* class stores the basic information to identify and playback the audiovisual content. The *video* attribute contains the video track path to be streamed.

The *Poi* class represents, together with the next two classes, the additional information related with a point of interest. In this class, textual –*name* and *description*– and complementary –*website* and *latitude*, *longitude* and *zoom_lvl*– information is stored.

The *Category* class groups the points of interest according to a certain criteria, representing the typological information of a PoI. The *icon* attribute contains the image used to represent the markers whose related PoI belongs to a certain category.

The *Picture* class stores the visual information of a point of interest. Note that a PoI can have more than one picture.

Finally, the appearances of the points of interest on the hypervideo are expressed in the *Marker* class, pointing out the position (*x*,*y*) tracking through the lifetime of a marker of a PoI. Also note that the same point of interest can appear independently on multiple hypervideos.

4.4 Server and module communication

The hypervideo server acts as an intermediary between the creation and visualization modules with the database. The Apache server gets the HTTP requests of both modules: the first stores the data with HTTP POST requests, while the second gets that data with HTTP GET requests in JSON format.

While the HTTP POST requests from the creation module are pretty straightforward, the HTTP GET requests from the visualization module responses are more complex and are shown next.

In Figure 3, the hypervideo list request response body encoded in JSON format is shown. The *hypervideoList* key contains an array of hypervideos. Each of these have the properties defined in the data model –*id*, *title*, *description*, *thumbnail* and *video*–.

```
{
  "hypervideoList": [
    {
      "id": 1,
      "title": "This is an example",
      "description": "A description",
      "thumbnail": "http://server/thmb.png",
      "video": "http://server/video.mp4"
    },
    {
      "id": 2,
      "title": "This is another example",
      "description": "Another description",
      "thumbnail": "http://server/thmb2.png",
      "video": "http://server/video2.mp4"
    }
  ]
}
```

Figure 3. Hypervideo list request response body encoded in JSON format.

The *metadata* is the information required to play a hypervideo. The *hypervideo* property defines an object, whose keys can be put in 4 groups, as shown in Figure 4:

- Audiovisual content location: *title* and *video* properties.
- Category list: the *categories* property contains an object, whose keys are the identifiers of the categories of the PoIs that appear in the production. For each of these categories, its *name* and *icon* are provided.
- Points of interest and their category: the *pois* key contains an object, whose keys are the identifiers of the PoIs that appear in the production. For each of these points of interest, the identifier of its category is provided.
- Marker tracking: the *markers* key contains an object, whose keys match the seconds where a marker has to be rendered on the screen. For each of these seconds, another object is defined, whose keys are the identifiers of the points of interest that marker represents. For each of these, an object is defined, containing the *x* and *y* properties that specify the point on the screen where the marker has to be rendered.

```
{
  "hypervideo": {
    "title": "This is an example",
    "video": "http://server/video.mp4",
    "categories": {
      "1": {
        "name": "First category",
        "icon": "http://server/cat1.png"
      },
      "2": {
        "name": "Second category",
        "icon": "http://server/cat2.png"
      }
    },
    "pois": {
      "4": 1,
      "5": 1,
      "6": 2,
      "7": 2
    },
    "markers": {
      "21": {
        "4": {
          "x": 29.27,
          "y": 18.12
        },
        "5": {
          "x": 62.55,
          "y": 18.12
        }
      },
      "22": {
        "4": {
          "x": 28.34,
          "y": 18.47
        },
        "5": {
          "x": 63.64,
          "y": 18.33
        },
        "6": {
          "x": 13.82,
          "y": 36.89
        }
      }
    }
  }
}
```

Figure 4. Hypervideo metadata request response body encoded in JSON format.

Finally, Figure 5 describes the PoI additional information request response body, encoded in JSON format. As the PoI typological information is already known, the following data is received as properties of the *poi* key:

- Textual information: *name* and *description* properties.
- Visual information: the *pictures* property defines an array of picture URLs.
- Complementary information: the *website* property contains the PoI webpage URL, which is also encoded in a QR code, using base64 in png format and stored in the *qr* property. The *location* key defines an object, with *latitude*, *longitude* and *zoom_lvl* properties, used to display a map.

```
{
  "poi": {
    "name": "This is the name",
    "description": "Why it is important",
    "pictures": [
      "http://server/pic1.jpg",
      "http://server/pic2.jpg",
      "http://server/pic3.jpg"
    ],
    "website": "http://poi.com",
    "qr": "data:image/png;base64,...",
    "location": {
      "latitude": 39.637,
      "longitude": 2.644,
      "zoom_level": 10
    }
  }
}
```

Figure 5. PoI additional information request response body encoded in JSON format.

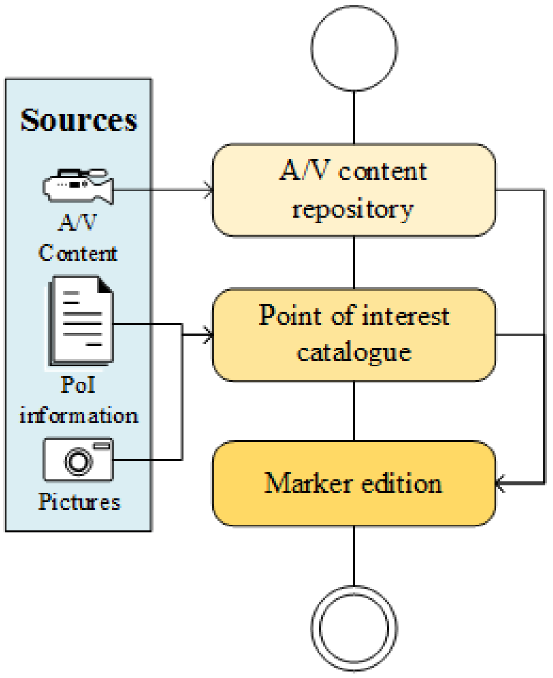


Figure 6. Creation Module detail.

5. CREATION MODULE

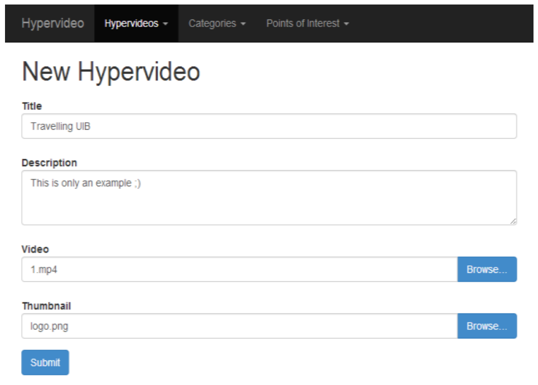
The goal of this module is enable the user to create hypervideos through three steps: manage the audiovisual content repository, create the point of interest catalogue and edit the placements of the markers. These steps are shown in Figure 6 and are defined below:

5.1 Audiovisual content repository

In order to guarantee compatibility with the specifications used in the visualization module –HbbTV 1.0 [13], Android [14] and Samsung Smart TV 2012 [15]–, multimedia content must meet the following specification:

- Video codec H264/AVC
- Audio codec HE-AAC
- Container MP4

This multimedia file is uploaded to the platform through the audiovisual repository management interface, as shown in Figure 7, enclosing descriptive information about the hypervideo.



New Hypervideo

Title: Travelling UIB

Description: This is only an example ;)

Video: 1.mp4

Thumbnail: logo.png

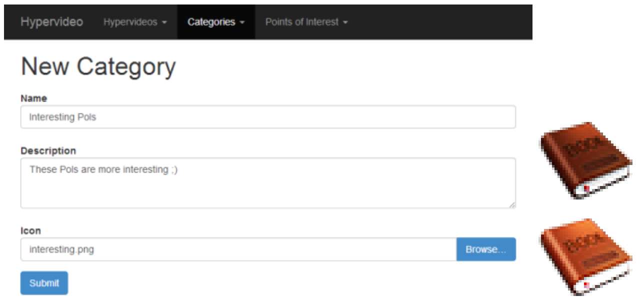
Figure 7. A/V repository management interface.

5.2 Point of interest catalogue

The PoI catalogue is based on categories. First, categories are created as needed through the category management interface, attaching its name and icon.

The category icon must be in PNG format with transparent background, and its size must be 36x72px. In the first half of the image the "inactive" category icon will be placed, while the "active" icon will be placed in the lower half.

In Figure 8 the category management interface is shown, as well as an example of category icon.



New Category

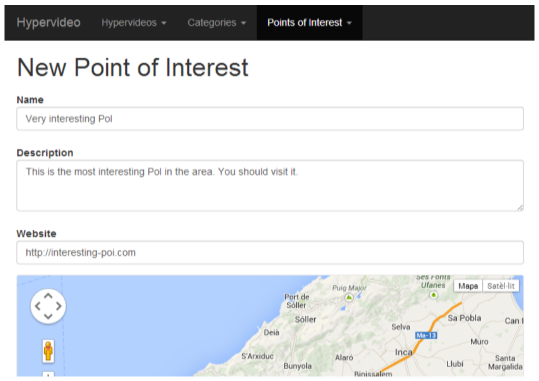
Name: Interesting PoIs

Description: These PoIs are more interesting ;)

Icon: interesting.png

Figure 8. Category management interface & icon example.

Once needed categories are created, the editor inserts the new points of interest in the platform. As shown in Figure 9, the form is filled with the PoI data: name and description (textual information), category (typological information), pictures (visual information), web page URL and location, selected with an interactive map (complementary information).



New Point of Interest

Name
Very interesting PoI

Description
This is the most interesting PoI in the area. You should visit it.

Website
http://interesting-poi.com

Figure 9. Point of Interest management interface

5.3 Marker edition

The markers purpose is to position the points of interest over the video images. In order to ease the marker edition, a tool inside the creation module is developed and shown in Figure 10.

This tools lets the producer browse the media content, select the temporal intervals where a point of interest appears, and specify its position, pointing out as many key positions as needed.

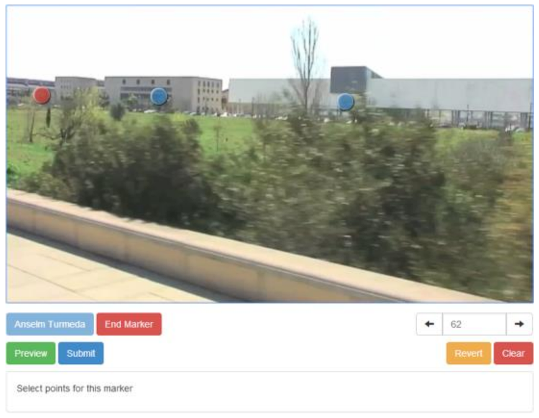


Figure 10. Hypervideo marker edition interface

Once the positions are defined, intermediate positions are generated through lineal interpolation, at a position per second frequency. If it is needed, the editor is able to correct these automatically generated positions.

The result of specifying the markers is stored in the database as the temporal tracking of the markers as a tuple, described in (1): in function of the hypervideo *h*, the second *t* and the PoI *p*, what position (*x*,*y*) relative to the video size a marker has.

$$f(h,t,p) = (x,y) \qquad (1)$$

6. VISUALIZATION MODULE

The visualization module is represented by the Hypervideo player, developed as a multiplatform application for the following interactive TV technologies: HbbTV 1.0, Android 4.0 and Samsung Smart TV 2012.

These technologies introduce a type of application known as "Web Application" [16] or "Javascript Application" [17] that eases multiplatform development. Having all the TV technologies using similar development model makes it suitable to code a single application logic, with an abstraction layer for every TV technology. The individual web technologies that this module makes use of are shown in Table 1.

Table 1. Web technologies used by Interactive TV

	Interactive TV technology		
	HbbTV 1.0	***Android WebView***	***Samsung Smart TV***
Markup	CE-HTML [18]	HTML5	
Style	CSS TV Profile 1.0 [19]	CSS3	
Interactivity	ECMAScript (Javascript)	Javascript	

6.1 Application states

The hypervideo player states sequence is shown in Figure 11 and the behavior of the application in each of these states is described next:

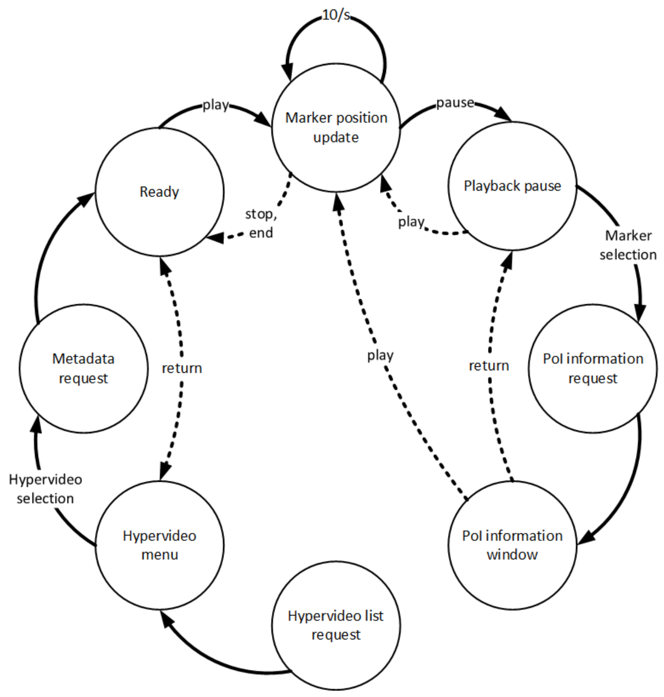


Figure 11. Sequence diagram of the module states. Dotted transitions denote backwards transitions.

In the first state, *hypervideo list request*, an HTTP GET request is done to the server via the XMLHttpRequest object [20]. When the application gets the response, the hypervideo menu is built and the application enters the next state.

In the second state, *hypervideo menu*, the user can select which hypervideo wants to play, heading into the *metadata request* state.

The application asks the server for the selected hypervideo metadata in the third state, named *metadata request*. This is done as before, through an HTTP GET request via the XMLHttpRequest object. When the reponse arrives to the application, the data structures needed to play the hypervideo are prepared and the application state changes to *ready*.

The fourth state, *ready*, shows the user that the hypervideo streaming can be played when it presses the *play* key, entering the next state.

In the fifth state, *marker position update*, the streaming playback is going on while the markers are shown over it. Their position is calculated synchronously via linear interpolation ten times per second, in order to smoothen their movement.

As shown in Figure 12, the current position is calculated from the position for the current second and the position for the following second.

```
t = current_second;
d = current_second % 1;
// for every PoI that is going to appear
p0 = position_in(t);
p1 = position_in(t + 1);
p = {'x': (p1.x - p0.x) * d + p0.x,
     'y': (p1.y - p0.y) * d + p0.y};
```

Figure 12. Marker position interpolation pseudocode

While in the fifth state, the user is able to filter the markers that appear by their category, through the category menu. In order to select a point of interest, following the link of a marker, the user first has to press the *pause* key, entering the next state.

In the sixth state, named *playback pause*, the user can select the marker(s) that appear on the hypervideo, if there is any marker. By pressing the *enter* key, the user follows the link denoted by the marker to the point of interest, entering the following state.

The application performs an HTTP GET request to the server in the seventh state, *PoI information request*, through the usual mechanism. When the application gets the response, the point of interest information window is built and the application enters the last state.

In the eighth and last state, *PoI information window*, the additional information of the selected point of interest is shown. The user is able to browse the textual information, as well as the visual information (pictures). The complementary information is shown as a QR-code (website) and a Google Static Map [21] (location).

6.2 User interface

Player user interface is described in the following figures, showing the different parts of the application, as well as the changes in the interface on state changes.

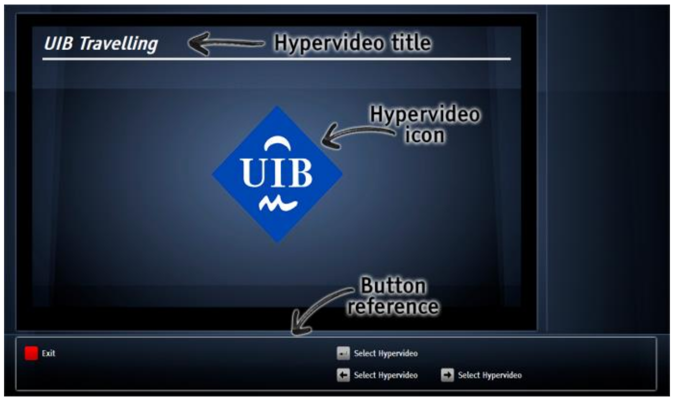


Figure 13. Hypervideo menu user interface

Figure 13 shows the user interface for the second state, *hypervideo menu*, in which the user is able to select the desired hypervideo using *left* and *right* keys.

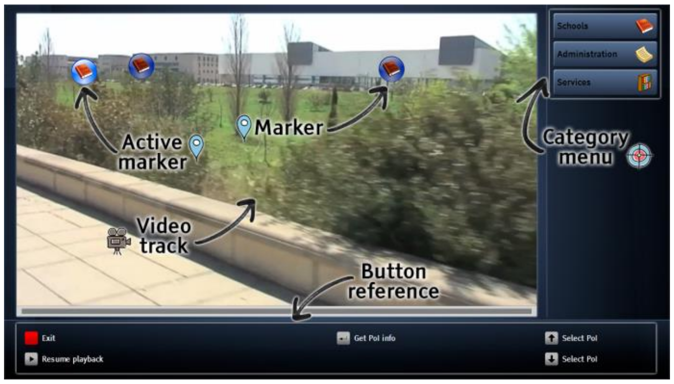
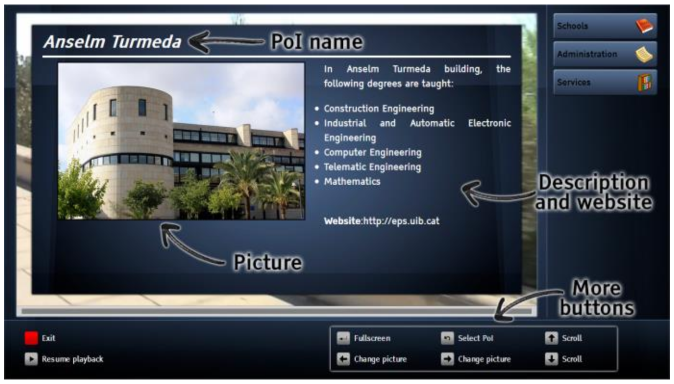


Figure 14. Point of interest selection user interface

The interface for states 4, 5 and 6 is shown in Figure 14. It has been divided in three areas: video and markers area, category menu and button reference. In the first area, the markers overlay the video streaming, representing the points of interest that appear in the video. If the user is selecting markers to access their additional information, these appear as "active" markers. On the right, the category menu lets the user filter the markers that appear by their category. Finally, at the bottom, the button reference informs the user of which buttons can it press and what is its function.


Anselm Turmeda

In Anselm Turmeda building, the following degrees are taught:

- Construction Engineering
- Industrial and Automatic Electronic Engineering
- Computer Engineering
- Telematic Engineering
- Mathematics

Website:http://eps.uib.cat

Figure 15. PoI information window user interface

When the user gets the additional information of a point of interest, entering the eighth state, that data is arranged in a window, showing the PoI textual information (name, description and website) and a picture, as shown in Figure 15.

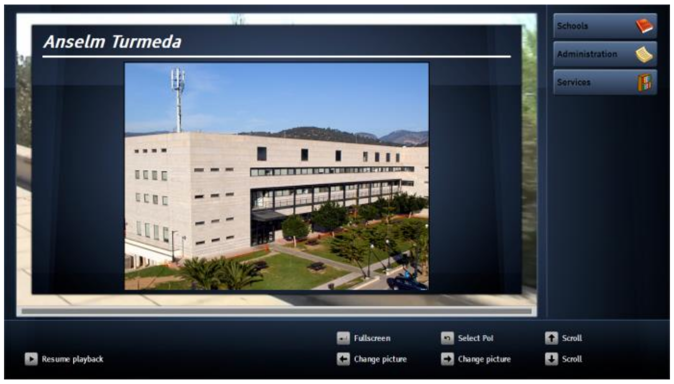


Figure 16. PoI picture full size user interface

The user can navigate through pictures with *left* and *right* buttons, as well as rendering them taking up the entire window by pressing the *enter* key, as described in Figure 16.

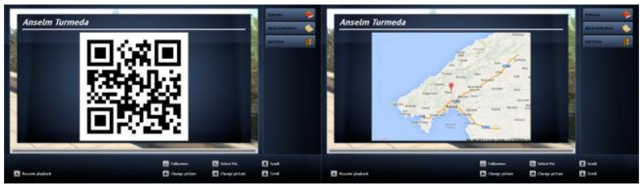


Figure 17. PoI complementary information user interface

Finally, as represented in Figure 17, the user can get the complementary information in a visual way, navigating across pictures: the point of interest location is shown in a Google Static Maps image and the website URL is encoded in a QR. Both of these images can be displayed taking up the entire window as well.

The user interaction with the interface is done mainly via the remote buttons: the D-pad (*up*, *right*, *down*, *left* and *enter*), the VCR buttons (*play*, *pause* and *stop*), the *return* button and the *red* button, used to close the application in any state.

In addition to these buttons, the application has been developed bearing in mind the gestures enabled in Samsung Smart TV, displaying a cursor where the user puts its hand, introducing hover, selection and drag in the interface.

6.3 Multiplatform development

As introduced before, the visualization module has been developed as a web application. In order to make it compatible for HbbTV, Android and Samsung Smart TV it was necessary to introduce an abstraction layer over the application itself for each TV platform.

In the end, each of these versions of the same application will be accessed in different ways: the HbbTV application is hosted in a web server and requested by HbbTV decoders; the Android version is requested from a WebView [22] in an Android application; while the Samsung application is packed in a zip file [23] and uploaded to the Smart Hub.

These abstraction layers are coded as Javascript constructors, named *HbbTV*, *AndroidTV* and *Samsung*, introducing virtual key definitions (such as *VK_LEFT*, etc), the following methods: *init*, *addKeyDownListener*, and *exit*, and the constructor *SystemPlayer*, which wraps the behavior of an mp4 streaming player by exporting the following methods: *play*, *pause*, *stop*, *seek*, *playState*, *currentState*, *duration*, *setOnPlayStateChange* and *setOnPlayPositionChanged*, and the *PLAY_STATES* object.

The HbbTV layer makes use of the elements *video/broadcast* and *application/oipfApplicationManager* present on the DOM, while the *SystemPlayer* introduces an *<object>* element with type *video/mp4* to stream the video.

The Android layer talks with the Java application that contains the *WebView* which is displaying the web app. The communication in Javascript to Java direction is made through the *window.AndroidInterface* object, which shows the method *destroyApplication*; the communication in Java to Javascript direction is obtained via the *window.android* object, whose *keydown* method is called by the Java application when a key is pressed. The *SystemPlayer* introduces an HTML5 *<video>* element to stream the video playback.

Finally, the Samsung layer makes use of the proprietary objects *Common.API.Widget* and *Common.API.TVKeyValue*, while the *SystemPlayer* follows the HTML5 way, just as the Android version.

All this layer choosing work is done by a tool that we built specially for this purpose: *TVmake*. It enables the developer to generate different versions of the same application, by including certain files in each version. It also is able to parse the files to be included with the PHP parser, so in the *<head>* of the *index.html* only the necessary Javascript files are included.

7. FUTURE WORK AND CONCLUSION

In this section, a series of future work proposals have been identified and are discussed below. Finally, the conclusion of this work ends the paper.

7.1 Future work

It has been noticed that during the PoI catalogue creation process, 3rd party information sources are checked in order to complete the PoI addition information. For this reason, and to ease the entire process, we are integrating the creation module with the point of interest database of Open Street Map, accessing its public API [24].

Another aspect that is going to be improved is the marker edition process. The manual model introduced in this paper can lead to mistakes and is a hard task in long videos. In order to automate this process, two techniques have been conceived, depending on the audiovisual content nature:

When working with a multimedia content recorded in the outdoors, with points of interest that can be identified and distinguished with their GPS position, such as buildings, the marker edition can be automated following a location-based process. The camera GPS position and orientation needs to be recorded during the content filming. This data is used together with the volume of the point of interest in GPS coordinates to calculate the viewing frustum (Figure 18), to know if a point of interest appears on the screen and which (*x*,*y*) position takes.

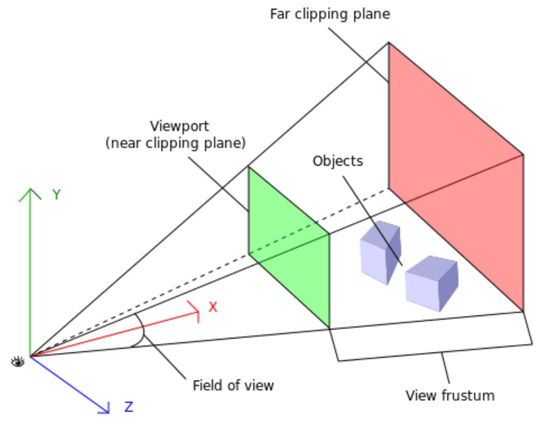


Figure 18. Viewing frustum to automate the marker positioning

However, when recording content in the interiors, with steady filming conditions, an alternate technique can be used: object contour detection and shape tracking. Examples of points of interest can be artwork in a museum. In order not to detect every object in the content, a picture of the desired PoIs needs to be supplied.

When applying any of these techniques, a reviewing process has to be introduced so as not to make the user experience worse.

In order to improve user interaction with the audiovisual creations, non-linear hypervideo navigation through their points of interest is presented:

- Intra-hypervideo navigation: a PoI links to a related PoI in the same content, enabling the user to explore other points of interest of the same topic.
- Inter-hypervideo navigation: a PoI links to its apparition in other hypervideo, enabling the user to look into that point of interest from another topic.

Currently, visualization module has been developed in three interactive TV technologies: HbbTV, Android TV and Samsung Smart TV. It is planned to make it compatible with more platforms and devices, such as Internet browsers and tablets, through its implementation in HTML5, iOS and other technologies.

One of the interactive TV technologies used, HbbTV, allows the development of live broadcast-based applications. It is wanted to deliver the audiovisual content through the broadcast channel and enabling marker selection and the PoI information window over it. It is being studied whether to pause the video track when selecting a PoI and then continue via streaming or not doing so, in conjunction with next proposal.

A common trend among interactive TV applications is the introduction of a second-screen application [25]. This application has many different uses, from replacing remote controller to social networking. The requirements for the hypervideo second-screen application are focused on a multi-user model:

- Obtain the additional information of a PoI: the PoI information window will be represented in the mobile application, so as not to disturb other users pausing the video and with additional information of a PoI that is not of their interest.
- Share the additional information of a PoI: the PoI information window goes back to the TV, so a user can share its experiences.
- Complementary information access: the user is able to access the additional information of the PoI directly from its device.
- Social networking: like and share hypervideo content, such as content snapshots or points of interest.

Finally, 360-degree video [26] support is being studied, largely improving user interactivity with the content.

7.2 Clonclusion

The results of the testing stage between audiovisual producers and university students has been very positive, emphasizing the added value of hypervideo productions destined to educative, promotional or informative purposes. Two key aspects were identified: 1) the importance of the audiovisual sources, revealing the need write down a filming script, listing the points of interest that are wanted to be displayed, and 2) it is essential keeping in mind the user perception in the visualization of a hypervideo: the time interval while the markers are alive has to be long enough to allow the users to see them and these cannot appear too piled up. These aspects ensure the system usability, easing the user to focus on the markers and tell them apart.

A test amongst users, University students, has been very successful, two factors being: 1) the increase in interest for making use of audiovisual content in an interactive way, and 2) the valuation of links between hypervideos, proving that the navigation through topics of interest via the points of interest is accepted and understood by the user.

8. ACKNOWLEDGMENT

This work was supported by project ConTVLab IPT-2012-0871-430000 of the Spanish Government.

9. REFERENCES

[1] N. Sawhney, D. Balcom, and I. Smith, "Hypercafe: Narrative and Aesthetic Properties of Hypervideo", *in Proceedings of the seventh ACM conference on Hypertext*, 1996. p. 1–10.

[2] G. Landow and P. Kahn, "Where's the Hypertext? The Dickens Web as a System-Independent Hypertext", *in Proceedings of the ACM conference on Hypertext*, 1992. p 149–160.

[3] *HbbTV* [website]. 2014. [Accessed: 1 August 2014]. Available on: http://hbbtv.org/.

[4] *Android TV* [website]. 2014. [Accessed: 1 August 2014]. Available on: http://www.android.com/tv/.

[5] *Samsung Smart TV* [website]. 2014. [Accessed: 1 August 2014]. Available on: http://www.samsung.com/smarttv.

[6] N. Sawhney, D. Balcom, and I. Smith, "Authoring and navigating video in space and time", *IEEE Multimedia*, no. 4, pp. 30–39, October-December 1997.

[7] J. Doherty et al., "Detail-on-demand hypervideo", *in Proceedings of the 11th ACM international conference on Multimedia*, 2003, pp. 600–601.

[8] F. Shipman, A. Girgensohn, and L. Wilcox, "Combining spatial and navigational structure in the hyper-hitchcock hypervideo editor", *in Proceedings of the fourteenth ACM conference on Hypertext and hypermedia*, 2003, pp. 124–125.

[9] C. Tiellet et al., "Design and evaluation of a hypervideo environment to support veterinary surgery learning", *in Proceedings of the 21st ACM conference on Hypertext and hypermedia*, 2010, pp. 213–222.

[10] *Proyecto LinkedTV* [website]. 2014. [Accessed: 1 August 2014]. Available on: http://linkedtv.eu.

[11] *Thinglink* [website]. 2014. [Accessed: 1 August 2014]. Available on: https://thinglink.com/

[12] *Wirewax* [website]. 2014. [Accessed: 1 August 2014]. Available on: https://wirewax.com/

[13] ETSI, "HbbTV specification Version 1.0", 2010. [Accessed: 1 August 2014]. Available on: http://www.hbbtv.org/pages/about_hbbtv/specification.php.

[14] Google, "Supported Media Formats | Android Developers", 2014. [Accessed: 1 August 2014]. Available on: http://developer.android.com/guide/appendix/media-formats.html.

[15] Samsung, "Player Specification", 2014. [Accessed: 1 August 2014]. Available on: http://www.samsungdforum.com/Guide/rel00010/index.html.

[16] Google, "Web Apps | Android Developers", 2014. [Accessed: 1 August 2014]. Available: developer.android.com/guide/webapps.

[17] Samsung, "Coding Your JavaScript Application", 2014. [Accessed: 1 August 2014]. Available on: http://www.samsungdforum.com/Guide/art00011/index.html

[18] W. Dees, P. Shrubsole, "Web4CE: accessing web-based applications on consumer devices", *in proceedings of the 16th ACM international conference on World Wide Web*, 2007, pp. 1303–1304.

[19] W3C Consortium, “CSS TV profile 1.0”, *W3C Candidate Recommendation*, 2003. [Accessed: 1 August 2014]. Available on: http://www.w3.org/TR/css-tv.

[20] WHATWG, “XMLHttpRequest”, *Living Standard*, 2014. [Accessed: 1 August 2014]. Available on: http://xhr.spec.whatwg.org/.

[21] Google, “Google Static Maps API V2”, 2014. [Accessed: 1 August 2014]. Available on: https://developers.google.com/maps/documentation/staticmaps/.

[22] Google, “Building Web Apps in WebView | Android Developers”, 2014. [Accessed: 1 August 2014]. Available on: http://developer.android.com/guide/webapps/webview.html.

[23] Samsung, “Testing Your Application on a TV for 2014”, 2014. [Accessed: 1 August 2014]. Available on: http://www.samsungdforum.com/Guide/art00121/index.html.

[24] OpenStreetMap Foundation, “OpenStreetMap API v0.6”, 2014. [Accessed: 1 August 2014]. Available on: http://wiki.openstreetmap.org/wiki/API.

[25] C. Courtois, and E. D’heer, “Second screen applications and tablet users: constellation, awareness, experience, and interest.”, *in Proceedings of the 10th ACM European conference on Interactive tv and video*, 2012, pp. 153–156

[26] Kolor, “360º video solotuions”, 2014 [Accessed: 1 August 2014]. Available on: http://www.kolor.com/video.