A Structured Document Model for Authoring Video-based Hypermedia

Tina T. Zhou
School of Information Technologies
The University of Sydney
NSW2006, AUSTRALIA
tzhou@it.usyd.edu.au

Jesse S. Jin
School of Design, Communication and I.T.
University of Newcastle
NSW 2308, AUSTRALIA
jesse.jin@newcastle.edu.au

Abstract

This paper describes a new method for authoring video-based hypermedia. It defines an XML-based (Extensible Markup Language) data modeling language called HyperVideo Authoring Language (HyVAL) for constructing structured documents in which the composition of internal video objects (segments, scenes, shots, frames and visual objects in frames) and external media objects (text, audio, images, html pages, etc.) is specified. The model allows an author to interactively specify various internal objects and relationship among internal and external objects. Through the language, a flexible presentation of video-based hypermedia is achieved and comprehensive viewer interactions with video objects are carried out. The experiment of structuring video-based hypermedia using HyVAL in our authoring and presentation environment is also described in this paper.

1. Introduction

Traditional video organizes the video clips along with an unstructured linear timeline. The interaction between viewer and video relies on the backward/forward function of video players. DVD authoring tools developed a navigation menu to provide more friendly interactive access to the video content. However, this interactive access ability has a constraint that the viewer has to always go back to the navigation menu before jumping to a video clip from the current clip. The video-based hypermedia intends to provide viewers with a comprehensive interaction capability. It allows viewers who require more information about a given section of video presentation to be able to jump to this information as easily as they navigate in Web pages through the hyperlinks. The video-based hypermedia is also searchable based on keywords or text from narration so that the viewer can quickly access a given section within the video [1].

Many multimedia authoring tools such as HypeCafe [2], HyperSoap [3], Hyper-Film [4], Macromedia Director [5], and Flash [6], have exploited the concept of video-based hypermedia. These authoring tools use scripts of different programming languages to implement the navigational interaction. There are some problems with these approaches. The development cost using such scripts is high. Writing scripts needs special training. The structure of the final products is also complicated and hard to update [7]. Most important of all, they do not support database operations or search engine functions.

The structured document approaches for creating multimedia presentation separates the specification of the form from the media content, permitting to vary the presentation at run-time [8]. It simplifies the authoring process and makes various media elements reusable. The data structure of structured document normally supports the search engines. Thus, we apply the structured document approach to author the video-based hypermedia. We propose a data modeling language called HyperVideo Authoring Language (HyVAL).

Compared with existing multimedia document models such as MHEG [9], HyTime [10], SMIL [11], and Z_YX [12], HyVAL focuses on the video-based hypermedia authoring rather than the production of multimedia presentation. HyVAL allows to access video objects, therefore, provides more adequate support for semantic modeling of video, interaction with video objects, and flexible integration of external media objects into the video. Like SMIL and Z_YX, HyVAL is based on XML (Extensible Markup Language) by which the searching can be done easily and accurately.

The work presented here is performed in the context of the development of a HyVAL document authoring and presentation system. The prototype of this system allows composition of structured

Proceedings of the 11th International Multimedia Modelling Conference (MMM'05)
1550-5502/05 $20.00 © 2005 **IEEE**

documents to form the video-based hypermedia by using HyVAL. Based on this experience, the discussion of this paper is devoted to the use of structured document to edit complex video-based hypermedia.

The rest of this paper is organized as follows. Section 2 gives an overview of HyVAL video authoring chain. Section 3 presents our model of video content description and the design of HyVAL. Section 4 describes the main features and the architecture of our HyVAL document authoring and presentation system prototype. Section 5 concludes this paper with a discussion on future research directions.

2. HyVAL video authoring chain

We have defined HyVAL document as a structured document written in HyVAL. The result of the execution of HyVAL document is a video-based hypermedia consisting of the presentation of a traditional video along with the author-specified visual effects, interaction opportunities and video-related media information. We name the video-based hypermedia as "HyVAL video" and the original traditional video as "HyVAL primary video". The simplest HyVAL document could form a HyVAL video as a traditional video without any special effects, interaction opportunities, or external media information, since the basic HyVAL document must have a description of traditional video structure elements and the relationships among these structure elements. The video structure model we used will be explained in Section 3.1.

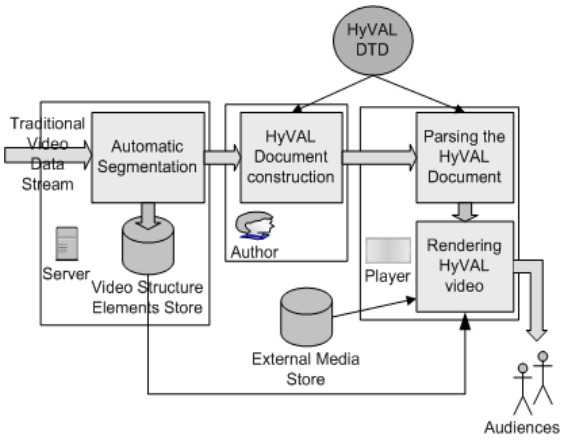

Figure 1. The process of authoring HyVAL video

The simplest HyVAL document is generated either through manual editing operations or through an integrated automatic segmentation tool by which the detection of segments, scenes, shots, frames of the HyVAL primary video is performed. In order to offer author a flexibility, the integrated automatic segmentation tool would be based on any shot/scene detect algorithm that has been currently well developed. Each frame is stored in the server's storage space (e.g., database, etc.) with an identical ID. To add additional information or interaction opportunities to the HyVAL document, the author need to modify the simplest HyVAL document with a consistency to HyVAL DTD (Document Type Definition). Once the author finished the modification, HyVAL document is delivered to a HyVAL player. The player parses the HyVAL document according to HyVAL DTD, and retrieves required video and Medias from server or other external media storage spaces and renders them on its display device. Figure 1 describes the steps involving in the process of HyVAL document creation and delivery.

3. HyVAL design

As a language supporting the video-based hypermedia authoring, HyVAL is designed to meet two important requirements of video-based hypermedia: 1) supplying data to search engines for a quick access to video content; 2) integrating a set of independent multimedia objects into a sequence of video with a navigational capability. The first requirement is met by the various information (including video format information, video production information, and author specified Meta data set) contained in the *head* element of HyVAL. The semantic information (e.g., information contained in "alt" or "description" attribute) provided in the declaration of various HyVAL objects also contributes to meet the first requirement. The model we used to define this information is based on the models that are currently well developed, such as the creation and production description scheme defined in MPEG-7 [13]. We will focus on how HyVAL meets the second requirement by addressing four important aspects of HyVAL: the organization of HyVAL objects, the layout of HyVAL video presentation, the temporal behavior of HyVAL video, and the viewer interactions.

3.1. The object organization

We have defined three types of objects in HyVAL video: 1) video structure objects; 2) internal video objects; and 3) external media objects.

The video structure objects refer to the elements derived from classic dramatic structure model of traditional video that can be found in a number of existing approaches [14]: the video is divided into a number of successive segments, each segment is

composed of successive scenes, each scene contains successive shots, and each shot is composed by frames. We define the video structure objects as essential elements of HyVAL video. The arrangement of *frame*, *shot*, *scene*, or *segment* elements based on timeline composes the traditional linear video. The hierarchical structure of this type of objects is shown in Figure 2.

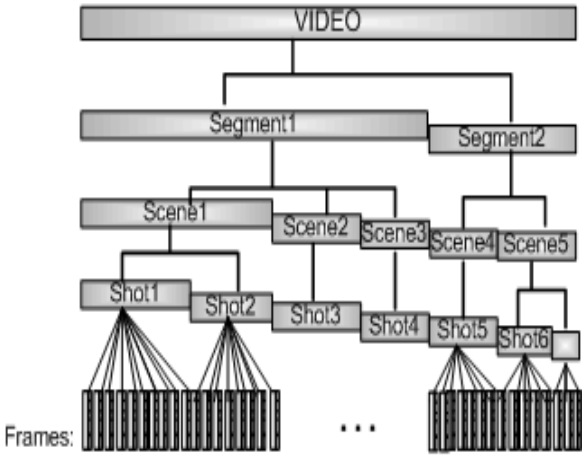

Figure 2. The structure of video structure objects

HyVAL defines the *internalVideoObject* element and the *externalMediaObject* element to describe the internal video objects and the external media objects, respectively. The internal video object refers to the natural 2-D visual object appeared in HyVAL primary video, such as people, dog, ball, etc. The external media object refers to the media to which the HyVAL primary video links. It could be a traditional video, a HyVAL video, an image, a text, a text-stream, an animation, or an audio. Both internal video objects and external media objects can be further broken up into spatial/temporal subparts by the support of *fragment* element. An example of using *fragment* element can be found in Figure 3 where the "Dog" internal video object has been separated spatially into two parts, "head" and "body".

Internal video objects and external media objects are declared within the video structure objects since they are both related to the certain part of HyVAL primary video. According to the inheritance relationship between the video structure objects, the children video structure object nodes of a video structure object inherit and only inherit the internal video objects (or external media objects) that its parent node has, so that the children video structure object nodes are able to access the internal video objects (or external media objects) that its ancestors have. Thus, the duplicate declaration of the same object is avoided. Figure 3 shows an example in which a "dog" object declared within the "scene1" *scene* element is shared by all shot elements of "scene1".

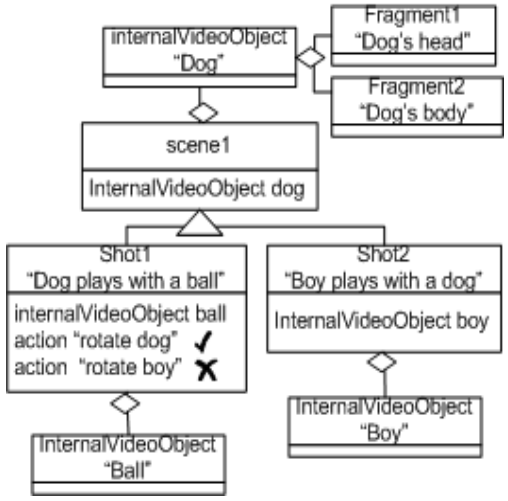

Figure 3. Example of objects relationship graph

3.2. Layout

HyVAL offers a layout model comparable with the one used in SMIL[11]. HyVAL adopts the *root-layout* and *region* element of SMIL. However, since the design of HyVAL is mainly targeting on the authoring of hypermedia video, its *root-layout* element and *region* element has a different meaning from SMIL. The *root-layout* element in HyVAL determines the size of the view screen – the window where the HyVAL primary video is rendered. The *region* element refers to the small block inside the video rendering widow defined in *root-layout* element. HyVAL brings an additional *window* element to define more display spaces besides the HyVAL primary video rendering window.

Root-layout, region and *window,* control the position, size and scaling of the visual objects involved in HyVAL document. The HyVAL video structure objects are rendered in the window defined in *root-layout* element by default. All other visual objects have to be assigned to either a region or window for presentation purpose when they are called.

3.3 Temporal behavior

HyVAL sets up its video sequential manner based on the structure defined in Section 3.1. There is no element defined in HyVAL to specify the sequential relationships between segment, scene, shot, and frames. A HyVAL video player renders HyVAL video structure objects in a sequential manner by default.

The synchronization is specified by the synchronized element *par*. *Par* binds the video structure objects with *objectLocator* elements and *actionEventLocator* elements to indicate the duration that an external media object, an internal video object, or an action event (detailed in Section 3.4) is available.

HyVAL offers the *objectLocator* element to allocate the presentation of an internal video object or an external media object. The *objectLocator* element has two important attributes: "begin" and "end". The value of "begin" and "end" determines when an object presentation is activated and disabled respectively. The value of both two attributes could be a video *frame* ID or an *actionEvent* ID (detailed in Section 3.4). With a *frame* ID, the presentation of an object starts on the point that the primary video with the specified *frame ID* is displaying. With an *actionEvent* ID, the presentation of an object starts when an event with the specified *actionEvent ID* occurs. Figure 4 shows an example of using a begin value within a *par* element.

```
<scene id="scene1" begin="1" end="600">
   <externalMediaObject id="videoObject1"
           src="d:/lecture1.mpg"
           type="video" alt="lecture1">
      <fragment id="intro" begin="00:00:00"
                end="00:10:00">
      <fragment id="topic1" begin="00:10:00"
                end="00:20:00">
   </externalMediaObject>
   <externalMediaObject id="imageObject1"
           src="d:/diagram1.jpg"
           type="image" alt="diagram1">
   </externalMediaObject>
   <par>
     <par>
        <shot id="shot1" begin="1" end="300" />
        <objectLocator
             object="videoObject1#intro"
             begin="6" window="vidoeWindow1" />
     </par>
     <shot id="shot2" begin="301" end="600" />
     <objectLocator object="imageObject1"
          begin="id(videoObject1#intro)(4m)"
          end="600" region="imageRegion1" />
   </par>
</scene>
```

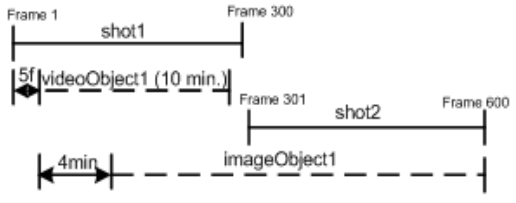

Figure 4. Example of using begin value in par element

3.4. Viewer interaction

HyVAL supports more sophisticated viewer interactions comparing to the interaction capabilities of the existing multimedia presentation document data models. The interaction model of HyVAL is built by using the *actionEvent* element and *actionEventLocator* element. The *actionEventLocator* element determines the author defined "actionEvent" according to the viewer action (e.g., mouse over, mouse right click, etc.) on specified object. The *actionEvent* element determines which object acts and how it acts when the viewer has an action on the same or different object. It can contain many types of object action elements including *link*, *font*, *size*, *color*, *zoom*, *fade*, *rotate*, and *3D*.

The *link* element achieves the navigational interaction between viewer and HyVAL video objects. Figure 5 shows an example that a link starts up the video "vObject1" in the "videoWindow1" window and the current HyVAL video continues playing in its own window when the viewer mouse-left-clicks on the screen while the "shot1" is playing.

```
<scene id="scene1" begin="1" end="600">
   <externalMediaObject id="vObject1"
             src="d:/lecture1.vaml"
             type="video" alt="lecture2">
   </externalMediaObject>
   <actionEvent id="link1" object="vObject1">
      <link>new</link>
   </actionEvent>
   <par>
      <shot id="shot1" begin="1" end="300" />
      <actionEventLocator actionEvent="link1"
           object="shot1" action="left-click"
           window="videoWindow1" />
   </par>
   <shot id="shot2" begin="301" end="600" />
</scene>
```

Figure 5. Example of a navigation interaction

The *font*, *size*, and *color* elements are applied to objects that have a text or text-stream type. The *zoom* element contains the percentage value. The percentage values less than 100% create the zone-out effect while the values over 100% create the zone-in effect. The *fade* element allows the author to specify a transparency percentage for visual objects to appear as in the initial frame and incrementally change the opacity setting over the specified number of frames to the transparency percentage designated in the element content. The *rotate* element allows the rotation of a visual object to specified degree. The *3D* element creates a three dimensional effect of a 2-D visual object.

4. HyVAL authoring tool

According to the authoring chain described in Section 2 and the design of HyVAL described in Section 3, we have developed a HyVAL authoring and presentation system. In our system, the HyVAL Editor

(figure 6) allows to semi-automatically specify the video structure and the relationship between video and its linked media objects using the HyVAL description format discussed in Section 3. The HyVAL Editor provides a simple way for visualization, navigation and modification of HyVAL video content.

Figure 6. The view of HyVAL Editor : 1. the HyVAL document structure section; 2. the attribute section; 3. the HyVAL video presentation section; 4. the video control functions; 5. the video, scene and shot information section(including the video information, the scene information and the shot information); 6. the visualization of video structure section (including the scenes in video, the shots in scene and the frames in shot).

If the author wants to create a new HyVAL video application, he/she simply selects the primary video and the HyVAL Editor automatically creates the basic structure of the primary video based on the result of various shot detection algorithm according to user's preference, thus, the HyVAL Editor gives its users a flexibility to integrate different shot detection tool. The author could also choose to manually create this basic video structure according to the frames displayed in the video structure section of the HyVAL Editor. After the basic video structure has been identified, the simplest HyVAL document will be shown in the HyVAL document structure section (Part 1 of Figure 6), and the corresponding attributes' value will be shown in the attribute section (Part 2 of Figure 6) if the author clicks on any node of tree structure of HyVAL document in HyVAL document structure section. Then, the author could adjust video structure and add relationships between the video objects and external media objects through the editing function of HyVAL document structure section and attribute section. The visualization of final video structure will be showed in the visualization of video structure section of HyVAL Editor (Part 6 of Figure 6). The HyVAL video presentation section of HyVAL Editor (Part 3 of Figure 6) gives author an idea about how the editing HyVAL video looks like.

5. Conclusion

This paper introduces HyVAL, an XML-based hypervideo authoring language for constructing the structured documents to author video-based hypermedia. HyVAL meets the requirements of video-based hypermedia by supplying structured data set for search engines and providing a comprehensive support for video presentation along with various viewer interactions. In addition, our experimental work, HyVAL Editor, has provided a way to implement such video-based hypermedia applications. Future developments will include the modeling of indications of interaction opportunities and the generation of personalized video-based hypermedia content.

6. References

[1] W. Ma, Y. Lee, D. H. C. Du, and M. P. McCahill, "Video-Based Hypermedia for Education-on-Demand", Multimedia, IEEE, Vol.5, Issue1, 1998, pp.72-83.

[2] N. Sawhney, D. Balcom, and I. Smith, "Hypercafe: Narrative and Aesthetic Properties of Hypervideo", Proc, Hypertext 96, ACM, 1996, pp.1-10.

[3] J. Dakss, S. Agamanolis, E. Chalom, and V. M. Bove, "Hyperlinked Video", SPIE Multimedia Systems and Applications, Vol. 3528, 1998.

[4] R. Tua, "From Hyper-Film to Hyper-Web: The Challenging Continuation of A European Project", EVA 2002 Proceeding, Pitagora Ed., Bologna, 2002.

[5] URL: http://www.macromedia.com/software/director.

[6] URL: http://www.macromedia.com/software/flash.

[7] N. Sawhney, D. Balcom, and I. Smith, "Authoring and navigation Video in Space and Time", Multimedia, IEEE, vol. 4, Issue 4, 1997, pp.30-39.

[8] A. Csinger, K. S. Booth, S. Gribble, D. Poole and S. E. Rathie, "Dynamic Hypervideo: Knowledge-based Annotation and Presentation of Video Documents", Department of Computer Science, University of British Columbia, URL: http://www.cs.ubc.ca/spider/csinger/abstracts/HT93/handout.html, Canada, 1994.

[9]ISO/IEC JTC1/SC29, “Information Technology-coding of Multimedia and Hypermedia Information – Part1: MHEG Object Representation”, ISO/IEC 13522-1, ISO/IEC IS, 1997.

[10] ISO/IEC 10744:1997, “Information technology – Hypermedia/Time-based Structuring Language (HyTime)”, URL:http://www.y12.doe.gov/sgml/wg8/docs/n1920/html/n1920.html, 1997.

[11] P. Hoschka, S. Bugaj, D. Bulterman et al. “Synchronized Multimedia Integration Language – W3C, Working Draft 2-February-98”, W3C, URL: http://www.w3.org/TR/1998/WD-smil-0202, Feb. 1998.

[12] S. Boll and W. Klas, “Z_YX – A Semantic Model For Multimedia Documents and Presentations”, Proceedings of the 8th IFIP Conference on Data Semantics (DS-8): Semantic Issues in Multimedia Systems, New Zealand, Jan. 1999.

[13] ISO IEC. 15938-5 FDIS “Information Technology – Multimedia Content Description Interface – Part 5: Multimedia Description Schemes”, March 2002.

[14] D. Gatica-Perez and M. Sun, “Linking objects in videos by importance sampling”, Multimedia and Expo, 2002, ICME’02, Proceedings of 2002 IEEE International Conference, Vol. 2, August 2002.