Video adaptation using the Variation Factory

Mulugeta Libsie and Harald Kosch
Department of Information Technology
University Klagenfurt
Klagenfurt, Austria
{mulugeta, harald.kosch}@itec.uni-klu.ac.at

Abstract—Video adaptation is an active research area aiming at delivering heterogeneous content to yet heterogeneous devices under different network conditions. This paper presents an architecture for generating variations (different versions) from an MPEG-4 source video to be used during adaptation. Different products are defined and the Variation Factory is introduced. It generates different versions of the source and an MPEG-7 metadata document. The information contained in the metadata document helps the system to identify the most appropriate version that meets the required Quality of Service (QoS). In addition to the implementation of the commonly used reduction methods, two novel methods, viz. object-based and segment-based variations are introduced. Our proposals are implemented and experimentally validated.

Keywords—Variation; Adaptation; Metadata; MPEG-4; MPEG-7; UMA.

I. INTRODUCTION

A huge amount of multimedia content is transmitted on the Internet. Client devices vary in their display capabilities, processing power and memory size. Different multimedia content are also stored in different formats. Hence, there is a growing need for applications to bring such diverse multimedia information to yet diverse devices under different network conditions and user preferences. Stored content has to be converted between different bit rates and frame rates since content is usually available in a single modality, resolution and format. It must also account for different screen sizes, decoding complexity and resource constraints of client terminals. This scenario is often referred to as *Universal Multimedia Access* (UMA) [1]. UMA deals with the delivery of multimedia content under different network conditions, user and publisher preferences, and capabilities of terminal devices [2]. As a result, there is currently an increasingly growing effort and research to address the issue.

To enable ubiquitous access, content variations must be generated either prior to delivery or on-the-fly [3]. Different versions (variations) can be generated, stored, selected and delivered. The MPEG-7 standard has identified UMA as an important issue and included descriptors to ensure that UMA applications are supported [2], [4]. Alternatively, media objects can be manipulated on-the-fly during delivery by using methods such as video transcoding, media conversion and summarization.

This work is funded in part by FWF (Fonds zur Förderung der wissenschaftlichen Forschung - Austrian Science Fund) under the project number P14789.

The work in this paper concentrates on video data and aims to show video variation supported with metadata as an approach to adaptation to enable ubiquitous access. By video variation we mean to generate different versions (possibly with different modalities) from the source video with reduced data size and quality by applying reduction methods and are stored in a media-database. The aim is to adapt content to potential terminal and network capabilities. Variations are created prior to delivery easing the real-time requirements of on-the-fly adaptation. In addition to the implementation of the widely known reduction methods, two novel methods, viz., object-based and segment-based variations are introduced and realized. MPEG-7 metadata descriptors are used to describe both the source and the variations. In our implementation, metadata are extracted automatically and stored in a meta-database. They are used to select the most appropriate variation that meets the required QoS. Alternatively, they can be transmitted to an adaptation engine (which could be a proxy, a router, or a gateway) or to end-users so that they may carry out the adaptation more efficiently. We also introduce a unifying framework architecture, called the *Variation Factory*. It is implemented in a server component, called the *Variation Processing Unit*.

The rest of the paper is organized as follows. Section II briefly surveys some of the important works related to the work in this paper. Section III details the variation products. Section IV introduces a unifying framework of the variation methods called the *Variation Factory*. Metadata description is covered in Section V. Implementation details are given in Section VI. Finally, Section VII concludes the paper.

II. RELATED WORK

There are a lot of research efforts to address UMA. Techniques for transcoding content on the Internet for heterogeneous devices was described in [1]. The authors describe a solution with two key components: a data representation scheme that provides multi-modal and multi-resolution hierarchy for multimedia, known as the *InfoPyramid*, and a method to customize the content to meet client capabilities, while delivering the most value. A personalized multimedia content delivery system dealing with both user preferences and terminal/network capabilities was presented in [3] to provide UMA within the EC-funded R&D project PERSEO. A video transcoder bank to resolve congestion and/or bandwidth limitation for mobile communications was proposed in [5]. Other works include in the areas of temporal adaptation [7], spatial reduction [8], syntax conversion [9], and object-based adaptation [10]. However, none of them considers the metadata creation process for adaptation.

III. Variation Products

A source video can be subjected to different reduction methods (which we prefer to call them variation methods or products) in generating variation videos. The following is a brief summary of the variation methods most of which are included in the *Variation Factory* (see Section IV).

- *Temporal variation:* is a variation of a video in the time domain. It reduces the frame rate of the visual stream. It can be achieved by dropping frames. In a layered coding system such as MPEG-4, it can also be achieved by dropping enhancement layers that contribute to temporal resolution.
- *Spatial variation:* is a variation of a video in the spatial domain where the size of each frame is reduced. In a layered coding system, it can also be done by dropping enhancement layers that contribute to spatial resolution.
- *Color variation:* is a variation of a video in the color domain by reducing the color depth of each pixel.
- *Bit Rate Variation* or *Quality Scaling:* reduces the quality or details of a video by changing encoding parameters.
- *Syntax conversion:* refers to re-encoding a video using a different encoding technique so that devices that cannot handle (decode) the current coding method can receive it.
- *Extraction*: extracts key frames, audio or video (visual part) from the source.
- *Object-based variation:* In MPEG-4, a scene can be composed of many objects, each coded separately. Hence, different reduction methods can be applied on each object. For instance, each object can be coded using different bit rates thereby scaling down the overall bit rate. Entire objects can also be dropped. In this work, the foreground and background components were isolated and treated as separate objects so that object-dropping (object-based adaptation) can be made possible which is the case in videos that were coded without objects. Since the relative importance of the foreground and the background regions vary from video to video, instead of using the terms "foreground" and "background", we will without loss of generality use the terms *primary* and *secondary* as defined below:

  A *primary object* is a foreground or background region extracted as an object that is relatively *more* important to the viewer.

  A *secondary object* is a foreground or background region extracted as an object that is relatively *less* important to the viewer.
- *Segment-based variation*: A shortcoming of the widely known methods such as temporal, color, and spatial is that they are indiscriminately applied on the entire video without paying attention to the particular characteristics or end-user preferences of its constituent parts. For instance, some parts have fast moving regions while others have stationary ones; some parts are colorful while others have less colors. Instead of applying the same method across the entire video, it will be advantageous in terms of minimizing quality loss and/or maximizing the gain in data size reduction to apply different methods on different parts. For instance, applying temporal reduction on a fast moving segment will have a jerky effect thereby highly degrading its quality. Instead, spatial or color reductions are better for such segments while temporal reduction is better applied on stationary ones. Similarly applying color reduction on a segment with a higher color depth provides a significant reduction in data size.

  In view of the above scenarios, a novel method, called *segment-based variation*, is introduced that partitions a video into homogeneous segments based on the physical characteristics of motion, texture and color and applies different reduction methods on different segments. The segmentation process is different from that used in content based video retrieval (CBVR) applications which relies more on semantics rather than physical characteristics.

IV. The Variation Factory

To realize the variation products defined above under a unifying framework, a novel architecture called the *Variation Factory* is defined as shown in Fig. 1. It is responsible for generating variations by making use of those products and an MPEG-7 metadata document.

The process of variation creation is summarized as follows. The input is an MPEG-4 video and the outputs are: (1) one or more MPEG-4 variation videos, and (2) an MPEG-7 metadata document that describes the source and the variations. Descriptors are extracted automatically. The emphasis is on MPEG-4 videos to exploit the extensive adaptation options provided by the standard, although most of the variation products defined in this paper are applicable to other coding methods as well.

The chain of applying variation methods as shown in Fig. 1 is not necessarily sequential. It is possible for a video to pass through only one variation method or more than one as shown by the vertical bi-directional lines. For instance, it is possible to apply only spatial variation and bypass all the other variation methods. It is also possible for a video to pass through more than one variation method, say for example, spatial variation followed by temporal variation, etc. The order in which methods are applied is not also important. This is because of the *order invariance* property possessed by the variation products as defined below, after establishing the notion of what we call *variation equivalence*.

<u>Definition 1: **Variation Equivalence**</u>

Two videos, V_1 and V_2 are said to be *variation equivalent*, denoted by $V_1 \overset{VE}{\equiv} V_2$, if they are the results of a variation from the same source video, V_S, and the following conditions are met: they have the same number of frames,

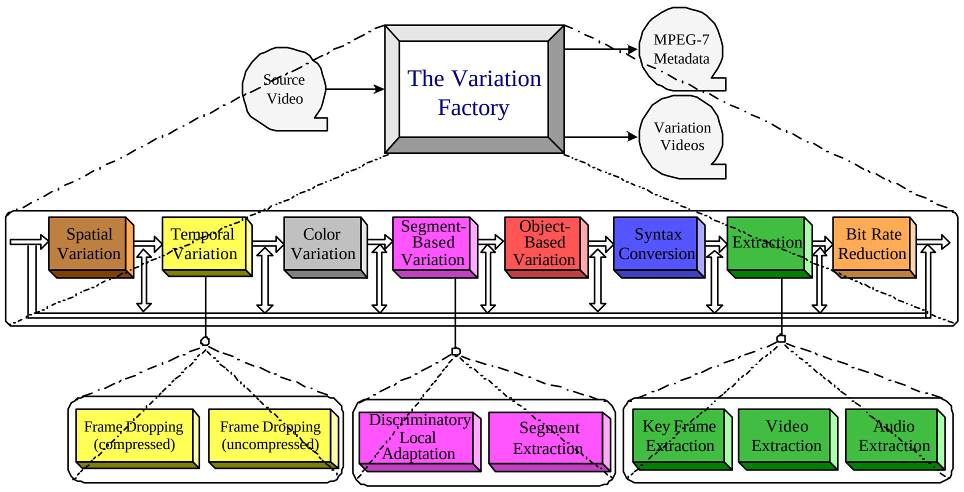

Figure 1. The Variation Factory.

frame rate, frame (picture) size, color depth, quality, and both have or have not audio.

The basic idea behind variation equivalence is that any one of the variation equivalent videos can be used to service a given request providing the same viewing experience to the user. It also helps to avoid the explosion of variations by eliminating equivalents.

Property 1: **Order Invariance**

Let M_1 and M_2 be variation methods from the set of variation methods defined earlier. Then, M_1 and M_2 are said to be *order invariant* iff

$$V_{M_1M_2} \overset{VE}{\equiv} V_{M_2M_1}$$

This property stipulates that the order in which the variation methods are applied on the source video is immaterial. In other words, it states that a video that is obtained by first applying the variation method M_1 followed by M_2 is variation equivalent to a video obtained by first applying M_2 followed by M_1.

The variation methods defined in this paper fulfill the property of order invariance which can be shown experimentally by examining the variation videos that result and checking if they adhere to Property 1. This property can be generalized to any number of variation methods. As a result of this property, when a video is piped through a series of variation methods, the results of the same set of variation methods but with different sequence or order produce *variation equi*valent videos.

Note, however, that the order of variation methods may be important for optimization purposes. For instance, if the three variation methods color, temporal and spatial are to be applied simultaneously, then it will be faster if they are applied in the order temporal, spatial and color. This is because a frame that will be in any case dropped can be dropped first (temporal variation) before any other variation method is applied on it, which otherwise would be a waste of time. This will be followed by spatial variation to get rid of those pixels that will not be part of the final variation video. Lastly, color variation can be applied on the remaining pixels. Our implementation takes this fact into account.

V. Metadata Description

MPEG-7 was selected for metadata description because it defines the *VariationSet* description scheme (DS) which allows standardized communication of audiovisual data between all components on the way to the client (proxies, routers, and gateways) so that they understand each other [6]. The *VariationSet* DS is used to represent the associations between different variations of multimedia resources. The major objective of the *VariationSet* DS is to allow the selection of the most suitable variation which can be used instead of the original to adapt to the different capabilities of terminal devices, network conditions and user preferences [2]. The variations may result from various types of multimedia processing such as summarization, reduction, transcoding, etc. The important descriptions include fidelity (the resulting quality), data size of the source and the variations, priority and type of relationship (such as temporal or spatial reduction). During delivery, the information contained in the MPEG-7 document will help the system to identify the most appropriate variation that meets the required QoS. They can also be transmitted to an adaptation engine or to end-users so that they may carry out the adaptation more efficiently. We realized that MPEG-7 lacks descriptors to specify the variation methods applied on each segment in segment-based variation. Hence, new descriptors were identified and defined. We plan to propose an input to MPEG-7 so that they are incorporated into the standard.

VI. Implementation

The variation products defined in this paper are implemented in a prototype system module for variation creation called the Variation Processing Unit (VaPU). It is a server component in charge of generating variations and the corresponding MPEG-7 documents. It logically lies between the media- and meta-databases (see Fig. 2). These databases respectively store the media and the metadata. VAPU is loosely coupled with the rest of the system in the sense that variations are created offline and that it does not participate in variation selection or in real-time operations for adaptation. Its task is to prepare variations and metadata for later adaptation. It is developed in Java under Java Media Framework (JMF).

JMF provides an application programming interface (API) that supports the integration of audio and video playback into Java applications and applets [11]. It provides access to the decompressed audio and video data by means of plug-ins. Through plug-ins, effects such as spatial or color reduction can be applied. Information such as color distribution for constructing color histograms can be collected or other algorithms such as shot detection can be implemented. The variation methods are implemented as independent plug-ins. This provides modularity and has two major advantages: (1) each variation product can be modified or replaced when better algorithms are developed without affecting the others; and (2) new variation products can be defined and independently implemented.

VaPU has four major components. The block diagram in Fig. 2 shows the components of VaPU and the interaction and information flow between them. A screenshot of the entry screen of VaPU is also shown in Fig. 3.

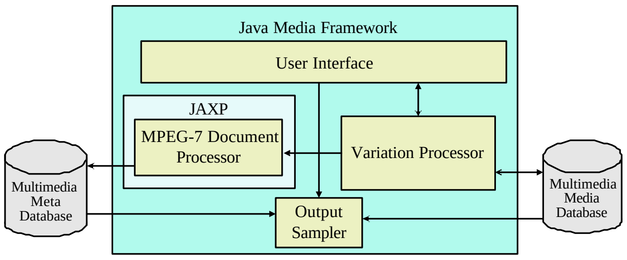

Figure 2. Block diagram of components of VaPU.

- The *User Interface* communicates with the user to get inputs. These inputs are passed as parameters to the *Variation Processor* and the *MPEG-7 Document Processor*. Input is entered through a user-friendly GUI built using Java Swing. The user interface has various modules applicable to the variation product selected.
- The *Variation Processor* creates the variations. It also extracts and gathers information about the source and the variations and avails it to the MPEG-7 Document Processor.
- The *MPEG-7 Document Processor* creates MPEG-7 documents with the appropriate MPEG-7 descriptors for the source and the variation videos. It uses JAXP (Java API for XML Processing), which is in general used to parse and transform XML documents. First, the MPEG-7 Document Processor constructs a DOM (Document Object Model) tree. Then, the DOM tree is parsed and an XML text file is produced.
- The *Output Sampler* is used to play/display variation processing results. It is an auxiliary and temporary facility to check the results of variation creation and is not part of the process of variation creation.

VII. Summary

In this paper, we defined variation products and introduced a novel framework architecture called the variation factory. Two new variation methods, viz., object-based and segment-based variations were introduced. Variation equivalence and order invariance were defined. Metadata description was explained and new descriptors identified and defined. The implementation of the variation products in a prototype system module for variation creation called the Variation Processing Unit is described.

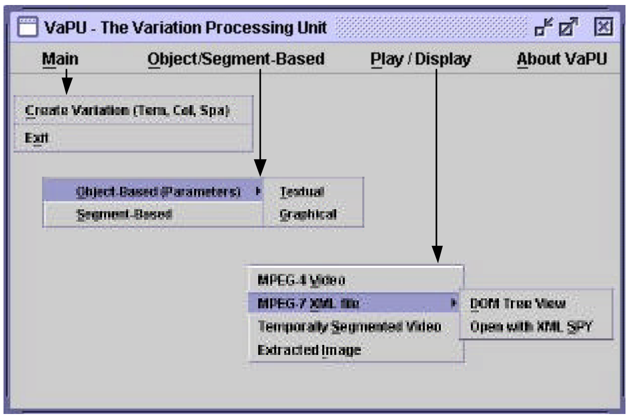

Figure 3. Entry screenshot of VaPU.

References

[1] R. Mohan, J. R. Smith, and C.-S. Li, "Adapting multimedia Internet content for universal access," *IEEE Transactions on Multimedia*, Vol. 1, No. 1, 1999, pp. 104-114.

[2] MPEG MDS Group, "MPEG-7 Multimedia Content Description Interface - Part 5: Multimedia Description Schemes (Final Draft International Standard)," *ISO/IEC N4242*, July 2001.

[3] O. Steiger, D. M. Sanjuán, and T. Ebrahimi, "MPEG-based personalized content delivery," in *Proceedings of the IEEE International Conference on Image Processing, ICIP 2003*, Barcelona, Spain, September 2003, pp. 45-48.

[4] L. Sampath, A. Helal, and J. R. Smith, "UMA-based wireless and mobile video delivery architecture," in *Proceedings of SPIE Photonics East: Internet Multimedia Management Systems*, Vol. 4210, Boston, November 2000, pp. 103-115.

[5] S. Dogan, A. H. Sadka, and A. M Kondpz, "MPEG-4 video transcoder for mobile multimedia traffic planning," in *Proceedings of the IEEE Second International Conference on 3G Mobile Communication Technologies, 3G'2001*, London, March 2001, pp. 109-113.

[6] L. Böszörményi, H. Hellwagner, H. Kosch, M. Libsie, and S. Podlipnig, "Metadata driven adaptation in the ADMITS project," *Signal Processing: Image Communication*, Vol. 18, No. 8, September 2003, pp. 749-766.

[7] B. Zheng and M. Atiquzzaman, "TSFD: two stage frame dropping for scalable video transmission over data networks," *IEEE Workshop on High Performance Switching and Routing*, Dallas, TX, May 2001, pp. 43-47.

[8] B. Shen and S. Roy, "A very fast video spatial resolution reduction transcoder," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2002.

[9] S. Acharya and B. Smith, "Compressed domain transcoding of MPEG," in *International Conference on Multimedia Computing and Systems (ICMCS 1998)*, June 1998, pp. 295–304.

[10] A. Vetro, H. Sun, and Y. Wang, "Object-based transcoding for adaptable video content delivery," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 11, No. 3, March 2001, pp. 387-401.

[11] R. Gordon and S. Talley, *Essential JMF: Java Media Framework*, Prentice Hall, New Jersey, 1999.