An integrated approach to summarization and adaptation using H.264/MPEG-4 SVC☆

Luis Herranz*, José M. Martínez

Video Processing and Understanding Lab, Escuela Politécnica Superior
Universidad Autónoma de Madrid, 28049 Madrid, Spain

Abstract

The huge amount of multimedia content and the variety of terminals and networks make video summarization and video adaptation two key technologies to provide effective access and browsing. With scalable video coding, the adaptation of video to heterogeneous terminals and networks can be efficiently achieved using together a layered coding hierarchy and bitstream extraction. On the other hand, many video summarization techniques can be seen as a special case of structural adaptation. This paper describes how some of them can be modified and included in the adaptation framework of the scalable extension of H.264/AVC. The advantage of this approach is that summarization and adaptation are integrated into the same efficient framework. The utility of this approach is demonstrated with experimental results for the generation of storyboards and video skims, showing that the proposed framework can generate the adapted bitstream of the summary faster than a conventional transcoding approach.

Key words: video summarization, scalable video coding, MPEG-4 SVC, H.264/MPEG-4 AVC, video adaptation

1. Introduction

Video summarization is a key technology required for efficient access and browsing of the huge amount of content available in multimedia repositories[1, 2]. It includes a number of techniques exploiting the temporal redundancy of video frames in terms of content understanding. These techniques try to provide the user with a compact representation in the form of a set of key images or shorter video sequences, but preserving the more informative parts[3, 4, 5]. Thus, the user can get an idea of what happens in the video without having to visualize the whole sequence. In general, a summarized sequence is built from the source sequence selecting frames guided by some kind of semantic analysis of the content.

Many algorithms have been proposed for keyframe selection, using different criteria and abstraction levels. At a low level, keyframe selection has been formulated as an optimization problem from both the set theory [3] and the rate-distortion[6] points of view. A widely used representation is the image storyboard, which abstracts the sequence in a set of key images. However, with video content, sometimes it is more useful and meaningful to present the content as segments of frames, instead of single frames. Segments provide information about the temporal evolution of the sequence, that isolated images cannot provide. This representation is often known as video skim, where significant segments are extracted from the sequence. Several approaches have been used in video skimming, including visual attention[7], image and audio analysis[8, 9] and high level semantics[2]. Between selecting single frames and selecting whole segments, there is still the possibility of selecting a variable amount of frames per segment. Fast forwarding the sequence at a constant rate can provide the user with a preview of the content, useful

☆This work is partially supported by the Spanish Government under project TEC2007-65400 (SemanticVideo) and by the Comunidad de Madrid under project S-0505/TIC-0223 (PROMULTIDIS).

*Corresponding author

Email addresses: luis.herranz@uam.es (Luis Herranz), josem.martinez@uam.es (José M. Martínez)

to browse it in a shorter time[10]. However, there are often less important parts that can be sped up, while more significant parts can be played at normal rate. Thus, a content based fast forwarding can be obtained if the skimming of frames is done in a frame basis guided by a semantic clue. Motion activity and camera motion have been used as clues to drive the selection of frames[11, 12]. A related technique is frame dropping aided by some low level features, where less important frames are discarded during the transmission of the sequence in case of network congestion. [13] uses MPEG-7 intensity motion descriptor to guide the frame dropping and [14] uses the perceived motion energy.

Besides these widely extended representations, there is an increasing interest on the development of more intuitive abstractions. In this direction, comics and posters have inspired several works[1, 15, 16] where the key images are presented with variable size in a layout where the temporal order of frames is replaced by a spatial scan order. Edited video is structured into more abstract units such as shots and then scenes, which usually contain several related shots. This hierarchical structure can be also exploited for summarization[17] and browsing[5, 4].

In modern multimedia systems, there are many possible ways to search, browse, retrieve and access multimedia content, through different networks and using a wide variety of heterogeneous terminals. Content adaptation[18] is a main requirement to effectively bring the content from service providers to the actual users, using different terminals and networks and enabling the so called Universal Multimedia Access (UMA). Especially important is the case of mobile devices, such as Personal Digital Assistants (PDAs) and mobile phones, where other issues such as limited computational resources and low power consumption requirements become very important. The MPEG-21 standard specifies a number of tools and concepts in a standard framework to enable advanced multimedia applications in heterogeneous usage environments. Particularly, MPEG-21 DIA[19] tackles the adaptation for universal access, with metadata tools to describe the usage environment, including terminal capabilities, network and user characteristics.

Considering a source bitstream and its adaptation and delivery as a modified bitstream, the whole process often implies decoding, adaptation to the target usage environment and encoding of the adapted content. This approach to adaptation is known as transcoding[20], and it can be computationally very demanding. Scalable video coding tackles the problem of adaptation at the encoding stage, in a way that simplifies the adaptation process. A scalable video stream contains embedded versions of the source content that can be decoded at different resolutions, frame rates and qualities, simply selecting the required parts of the bitstream. Thus, scalable video coding allows a very simple, fast and flexible adaptation framework to a variety of terminals and networks, with different capabilities and characteristics. The numerous advantages of this coding paradigm have motivated an intense research activity in the last years. Recently, the Joint Video Team (JVT) has worked in a scalable extension of the successful H.264/MPEG-4 AVC[21] standard, supporting multiple scalabilities, notably temporal, spatial and quality scalabilities. This new specification is known as SVC[22]. In this paper, the term SVC refers to this specific H.264/AVC standard extension.

As video is distributed in a compressed format, the use of coding parameters has been also explored for efficient video summarization and adaptation. The analysis stage can benefit from these compressed data, as they can provide low level semantic information but avoiding most of the burden of decoding the bitstream[23]. Motion vectors have been used for motion activity and camera based summarization[11, 12]. Besides, the coding structure can also be used for lightweight customization of the bitstream, directly operating with the compressed data. Specifically, for H.264/MPEG-4 AVC in the framework of MPEG-21 DIA, [24] proposes a content based adaptation system using a shot-based approach, while [14] uses a similar approach for frame dropping.

Usually, summarization and adaptation are considered separately in two independent stages. First, the content is decoded, analyzed and summarized into a new video sequence or a collection of images, that can be adapted to the usage environment constraints. However, summarization can be considered as a specific type of content based adaptation in which the main objective is to obtain a version of the content with a modified structure, where only the most relevant frames are kept in the adapted version.

Recently, some authors have reported systems which use the specific features of scalable video codecs for efficient analysis and summary generation. [25] proposes a content based adaptation en-

gine which integrates content analysis and wavelet-based scalable video for efficient content based adaptation. [16] exploits the hierarchy of motion information in scalable videos to extract a measure of motion activity, which is subsequently used to generate a comic-like summary.

In this paper, we describe a video summarization framework focused on efficient generation and adaptation of summaries. It uses the layered approach of SVC including a variety of video summarization approaches in a generic summarization model. Thus, video summarization is integrated into the same framework of SVC, enabling the provision of efficient and adapted summaries, taking advantage of hierarchical coding structures and bitstream scalability. Although the paper is not focused on the semantic performance of summarization algorithms, it also describes the application of the framework to the generation of storyboard and video skim summaries. The efficiency of the proposed approach is compared with a conventional solution for summary generation and adaptation based on transcoding.

The rest of the paper is organized as follows. Section 2 briefly overviews the SVC standard. A suitable model for describing summaries of temporal scalable bitstreams is presented in Section 3. Section 4 describes how video summarization is integrated into the SVC adaptation framework. Section 5 describes some examples of summarization algorithms. Experimental results are shown in Section 6. Finally, Section 7 draws some conclusions.

2. H.264/MPEG-4 Scalable Video Coding

The SVC standard[22] is built as an extension of H.264/AVC, including new coding tools for the generation of scalable bitstreams. SVC is based on a layered scheme, in which the bitstream is encoded into a base layer,H.264/AVC compliant, and one or more enhancenment layers. Each enhancement layer improves the video sequence in one or more of the scalability modes. There are different scalabilty modes, with temporal, spatial and quality ones being the most important.

Spatial scalability is achieved by using interlayer prediction from a lower spatial layer, in addition to intralayer prediction mechanisms such as motion-compensated prediction and intra prediction. The same mechanism of interlayer prediction for spatial scalability can provide also Coarse Grain Scalability (CGS) for quality scalability. Quality scalability can be also achieved using Medium Grain Scalability (MGS), which provides quality refinements inside the same spatial or CGS layer. Temporal scalability in SVC is provided using hierarchical prediction structures, already present in H.264/AVC. Each temporal enhancement layer increases the frame rate of the decoded sequence.

As in H.264/AVC, the recommendation specifies two conceptual layers: a Video Coding Layer (VCL), which deals with the efficient representation of the video content, and a Network Abstraction Layer (NAL), which deals with the format and header information in a suitable manner to be used by a variety of network environments and storage media. The bitstream is composed of a succession of NAL Units, each of them containing payload and header with several syntax elements. An Access Unit (AU) is a set of consecutive NAL units which results in exactly one decoded picture. In SVC, the versions at different spatial and quality resolutions for a given instant form an AU, and it can contain both base layer and enhancement layer NAL units.

Each NAL unit belongs to a specific spatial, temporal and quality layer. This information is stored in the header of the NAL unit in the syntax elements *dependency_id*, *temporal_id* and *quality_id*. This length of the NAL unit header in H.264/AVC is extended to include this information. In SVC, the base layer is always H.264/AVC compatible. However, the extended NAL unit header would make the bitstream non compliant with H.264/AVC. For these reason, each base layer NAL unit has a non extended header, but it is preceded by an additional NAL unit containing the SVC related information. These units are called prefix NAL units. If the stream is processed by a H.264/AVC decoder, these prefix NAL units and the other enhancement layer NAL units are simply ignored, and the base layer can still be decoded.

3. Summarization model for H.264/AVC with hierarchical structures

Video summarization includes a number of techniques which analyze the semantics of the source sequence and then generate a summary according to this analysis. For convenience, we separate the whole summarization process into two stages: analysis of the input bitstream and generation of the summarized bitstream. Actually, analysis is completely detached from the generation and it could be performed in a previous stage and stored as metadata.

A summary is usually generated by the concatenation of frames. Here, the basic unit for summarization is the frame. In the case of uncompressed video (e.g. in YUV format), it is possible to select each frame independently and build a new sequence just concatenating the values of the samples of each selected frame. A summary can be described with the indices of the frames of the source sequence that must be included. The analysis stage only needs to provide these indices to the generation stage.

Usually in compressed video streams, frames are coded in groups rather than individually using specific coding units, so it is more convenient to refer the output of the analysis stage to coding units rather than to single frames. This section introduces a suitable model for describing the summary in the case of video codecs with temporal scalability, such as H.264/AVC.

3.1. Summarization units and hierarchical coding structures

In H.264/AVC each frame is coded in an integer number of NAL units. For simplicity, we will consider that each frame is coded into one slice and one NAL unit, and it corresponds to a single AU. However, concatenating the NAL units of the frames belonging to the summary will probably generate a non-decodable bitstream, as most of them are encoded predictively with respect to previous frames in the source bitstream. If these reference frames have been removed from the output bitstream, predicted frames will not be decodable. For this reason it is more convenient to refer the results of the analysis to coding oriented structures rather than to single frames, taking into acount the prediction dependencies between them.

If we consider a sequence with N frames coded with T temporal decompositions (that means $T+1$ temporal levels), then the frame index can be notated as $n \in \{0, 1, \ldots, N-1\}$ and the temporal level as $t \in \{0, 1, \ldots, T\}$. For each temporal level, a subsampled version in the temporal axis can be decoded, as there are no breaks in the prediction chain. In this case, we use an alternative representation that describes the summary using group of frames related by prediction rather than frame indices. In this alternative representation, the basic unit for summarization is the Summarization Unit (SU). We define the Summarization Unit as a set of consecutive AUs at certain temporal level related by the prediction structure and that can be decoded independently from the other AUs in the sequence. The sequence is then partitioned into M Summarization Units. Fig. 1 shows an example of a hierarchical coding structure and its corresponding SUs. The SU at the highest temporal level is the one formed by an Instant Decoding Refresh (IDR) frame and three B frames which are coded predictively. Obviously, it is not possible to include a B frame in the summary without including its reference frames, as it would not be decoded by the user's decoder. However, there are more SUs in the bitstream, at different temporal levels, as the one composed by the IDR and B frames at the first temporal level, and the one composed only by the IDR frame. In SVC, the concept can be extended, in order to include the additional versions given by spatial and quality scalabilities. Thus, there is possible to define more SUs, with only the NAL units from the base layer, or including also NAL units from enhancement layers. The only requirement for these groups of NAL units is that they must be decodable independently of the rest of the bitstream (except other required non VCL NAL units such as parameter sets) and that their decoding results exactly in a set of consecutive frames at a certain temporal level.

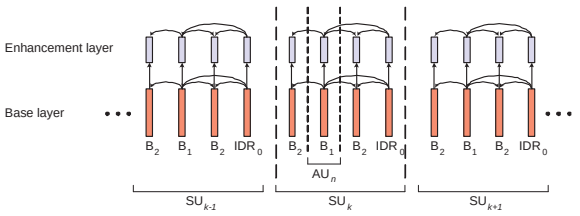


Figure 1: Coding structures and summarization units in H.264/AVC (base layer) and SVC.

Within this framework, the summaries are built by concatenating SUs, resulting directly in the output bitstream. All the frames in the bitstream can be decoded with a suitable H.264/AVC or SVC decoder. Each SU must be decoded independently from other SUs, as the summarization process could eventually remove some of them. In order to guarantee that each SU can be decoded independently it is important to provide each of them with a random access point. In H.264/AVC, the simplest method is the use of an Instant Decoding Refresh (IDR) Access Unit for each SU, as an IDR Access Unit signals that the IDR Access Unit and all the following Access Units can be decoded without decoding any previous picture. An additional advantage of using IDR Access Units is the limited error propa-

gation. An eventual transmission error would only propagate until the next IDR Access Unit.

The selection based on SUs has the drawback of losing some accuracy in the selection of frames. This accuracy depends on the length of the SU and it is given by the coding structure (e.g. the SU of Fig. 1 has an accuracy of 4 frames).

Besides the concept of SUs we define the summarization constraint $tlevel\,(m)$ as the maximum temporal level for each summarization unit SU_m. This function describes how to generate the summaries, as the indices of the frames do in the case of uncompressed bitstreams. If the value of $tlevel\,(m)$ is set to -1 for a certain m it means that SU_m is not included in the summary. The objective of the analysis stage of a summarization algorithm in this framework is to determine the summarization constraint for each video sequence, based on a certain content analysis.

3.2. Modalities of video summaries

There are different video summarization modalities that can be easily adapted to the proposed scheme. Depending on the values that $tlevel\,(m)$ takes for the SUs, we distinguish several modalities of video summaries (see Figure 2):

- Storyboard: built by selecting a few independent and separated frames to represent the content in few images. Within the proposed model, for convenience, we restrict the potential selected frames to be I frames belonging to the lowest temporal level. We also assume that the lower temporal resolution has only one I frame. In practice there is no noticeable difference for practical applications, and actually most storyboard summarization algorithms use temporal subsampling to speed up the analysis. With this assumptions, the storyboard is characterized as follows

$$tlevel\,(m) = \begin{cases} 0 & keyframe\ in\ SU_m \\ -1 & otherwise \end{cases}$$

- Video skim: the adapted sequence is shorter than the input sequence, obtained by selecting certain segments of the input sequence. In this case, the valid options for each SU are either not constraining its temporal level or skipping it. Thus if the maximum temporal level is t_{max} the video skim can be characterized as follows

$$tlevel\,(m) = \begin{cases} t_{max} & SU_m \in skim \\ -1 & otherwise \end{cases}$$

- Content based fast playback: this summarization modality is based on the acceleration and decceleration of the sequence controlled by a certain content based criteria, in order to visualize it in a shorter time. In this case, the number of frames of each SU is variable depending on the required frame rate at each SU.

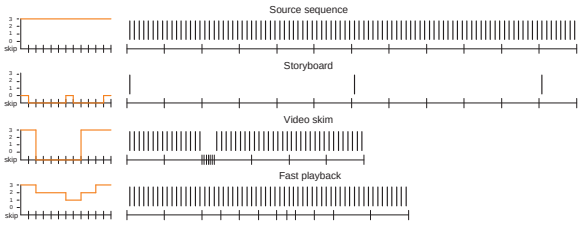


Figure 2: Examples of the function $tlevel\,(m)$ (left) and frames selected (right).

4. Adaptation and summarization in SVC framework

Most of video summarization techniques can be formulated as a special case of video adaptation, where the adaptation is performed in the temporal axis, and the adapted version is composed by the selection and concatenation of frames from the original sequence. For these reason it is very convenient to describe the summarization process using tools similar to those used for video adaptation.

The advantage of SVC relies on its efficient adaptation scheme. With SVC the adaptation engine is a simple module, known as extractor, which modifies the bitstream selecting only the parts required according to some constraints (see Figure 3). The constraints (resolution, bitrate, etc.) are imposed by the usage environment. The Usage Environment Description (UED) tools of MPEG-21 DIA can be used to describe, among others, the terminal capabilities and network characteristics with a standardized specification. The extractor selects the appropriate layers of the input bitstream satisfying the constraints. The output bitstream is also conforming to the SVC standard and it can be decoded with a suitable SVC decoder.

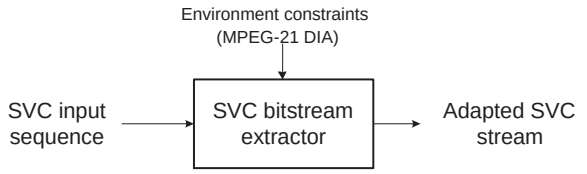


Figure 3: Adaptation in the SVC framework.

4.1. Extraction process in SVC

The extraction process in SVC is non-normative, with the only constraint that the output bitstream, obtained from discarding enhancement layers, must also be compliant with the SVC standard. However, the JVT provides the Joint Scalable Video Model (JSVM), including a software implementation of SVC. In this section we briefly describe the basic extraction process of SVC in the JSVM.

The extractor processes the NAL units using the syntax elements *dependency_id*, *temporal_id* and *quality_id* to decide which must be included in the output bitstream. Each adaptation decision is then taken for each access unit AU_n, where n is the temporal instant. Each layer (base or enhancement) in AU_n can be denoted as $L(d,t,q;n)$. An operation point $OP_n = (d_n, t_n, q_n)$ is a specific coordinate (d,t,q) at temporal instant n, representing a particular resolution (spatial and temporal) and quality, related, respectively, to the syntax elements *dependency_id*, *temporal_id* and *quality_id*. If we denote the extraction process as $\mathcal{E}(OP, AU)$, the result of adapting an Access Unit AU_n with a particular operation point OP_n can be defined as the adapted Access Unit $\tilde{AU}_n = \mathcal{E}(OP_n, AU_n)$, and it contains all the layers and data necessary to decode the sequence at this particular resolution and quality. For each AU_n, the extractor must find the operation point OP_n satisfying the constraints and maximizing the utility of the adaptation. In a typical adaptation scenario, the terminal and the network impose constraints that can be fixed (*display_width*, *display_height* and *display_supported_rate*) or variable (*available_bits*(n) related to the instantaneous network available bitrate). Thus, the adaptation via bitstream extraction can be formulated as an optimization problem:

for each instant n find $OP_n^* = (d_n^*, t_n^*, q_n^*)$ maximizing $utility\left(\tilde{AU}_n\right)$
subject to

$$frame\_width(d) \leq display\_width$$
$$frame\_height(d) \leq display\_height$$
$$frame\_rate(t) \leq display\_frame\_rate$$
$$bitsize\left(\tilde{AU}_n\right) \leq available\_bits(n)$$

In this formulation, $utility\left(\tilde{AU}_n\right)$ is a generic measure of utility or quality of the resulting adaptation. It should be computed or estimated for all the possible adapted AUs, in order to select the most appropriate. Depending on the application, it can be estimated by the extractor or be available as external metadata, which can be specified, for example, using the MPEG-21 Adaptation Quality of Service (AQoS) description tool[26]. The actual values of resolution and frame rate can be obtained indirectly from d and t, and the size of any Access Unit can be obtained just parsing the bitstream.

The JSVM extractor solves the problem using a prioritization approach. The NAL units in an AU are ordered in a predefined order and selected in this order until the target bitrate or size is achieved. In Figure 4 each block represents a NAL unit containing a layer $L(d,t,q;n)$. The base quality layer $(q=0)$ of each spatial and temporal level are placed first in the priority order. Then, NAL units including quality refinements are placed in increasing order of their temporal level. Spatial enhancement layers are placed next. The extractor just drops the NAL units with a priority lower than the required one.

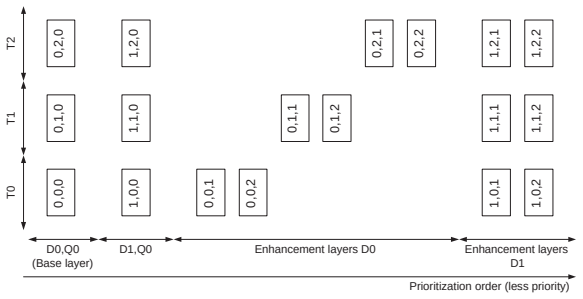


Figure 4: Prioritization of NAL units in the JSVM extractor.

However, this prioritization scheme does not ensure the optimality of the extraction path in terms of utility. For this reason, besides the basic extraction method, SVC provides additional tools for improved extraction, namely the optional syntax element *priority_id,* which signals explicitly the priority of each NAL unit, based on any other (non-normative) criteria[27].

4.2. Including summarization in the framework

In the previous framework, the constraints imposed to the adaptation engine are external, due to the presence of a constrained usage environment (*environment constraints*). The adaptation modifies the resolution and quality of the bitstream, but the information in the content itself does not change. However, there is no restriction on the nature of the constraints. Summarization can be seen as a modification of the structure of the bitstream based on the information in the content itself, in order to remove semantic redundancies in the temporal axis, in a constrained situation where the number of frames must be reduced considerably. For this reason, we reformulate the video summarization problem (typically, the selection of a suitable set of keyframes or segments) into the problem of finding the appropriate constraints such that the extractor generates a suitable summary. In this context, we call them *summarization constraints*. These constraints can modify the value of the temporal resolution. If both environment and summarization constraints are used together in the extraction, the result is an integrated summarization and adaptation engine which can generate summaries adapted to the usage environment using only SVC tools (see Figure 5).

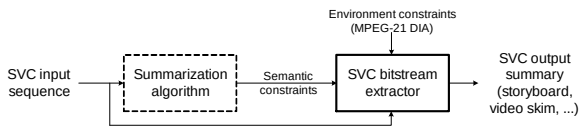


Figure 5: Integrated summarization and adaptation of SVC.

The adaptation process, as described previously, is performed on an AU basis. However, in the proposed summarization model, the summaries are referred to the SU index with the summarization constraint $tlevel\,(m)$, so it must be harmonized with the adaptation process. When a sequence is partitioned into SUs, each of them contains one or more AUs and, for simplicity, we assume that each AU belongs only to a single SU. Then we define a new summarization constraint $\widetilde{tlevel}\,(n)$ for each AU_n associated to a certain SU_m:

$$\begin{aligned} \forall n \quad &\in \quad \{0,\ldots,N-1\}, \\ &\widetilde{tlevel}\,(n) \equiv \quad tlevel\,(m), \quad AU_n \in SU_m \end{aligned}$$

The problem of adaptation in the extractor, including the new summarization constraint, can be now expressed as

for each instant n find $OP_n^* = (d_n^*, t_n^*, q_n^*)$ maximizing $utility\,(\mathcal{E}\,(OP_n, AU_n))$
subject to
$frame\_width\,(d) \leq display\_width$
$frame\_height\,(d) \leq display\_height$
$frame\_rate\,(t) \leq display\_frame\_rate$
$bitsize\,(\mathcal{E}\,(OP_n, AU_n)) \leq available\_bits\,(n)$
$t \leq \widetilde{tlevel}\,(n)$

The last constraint makes the extraction process content-based, constraining directly the temporal level. The problem can be solved using the same tools described in the previous section, including the prioritization scheme of the JSVM. Implicitly d, t and q are assumed to be positive (or zero). Thus, if $\widetilde{tlevel}\,(n)$ takes a negative value for a certain n, the problem has no solution, as the new summarization constraint cannot be satisfied. In that case, we assume that the extractor will skip that Access Unit not including any of its NAL units in the output bitstream. The summarization algorithm can take advantage of this fact to signal when a certain Summarization Unit must not appear in the output bitstream.

As in the model for H.264/AVC, all the SUs must be independently decodable for all the possible adapted versions. Again, the simplest solution is the use of IDR Access Units. In SVC, IDR Access Units only provide random access points for a specific dependency layer. For this reason, enhancement layers must also have an IDR Access Unit at the beginning of each SU, in order to guarantee the independence of the SUs for all layers.

5. Summarization algorithms

In order to demonstrate the utility of this approach to generate summaries in practical applications, we describe two algorithms for storyboard and video skim summarization, based on widely used approaches and adapted to the model described previously. We can further exploit scalability to increase the system efficiency, discarding those layers that do not improve significantly the results, and their processing increases the decoding burden and the amount of data to process (e.g. pixels). Thus, for analysis, we do not use spatial and quality enhancement layers, as, for our purpose, it is enough to process only the H.264/AVC base layer. For simplicity, we use the Group of Pictures (GoP) of the base layer as the basic analy-

sis unit. Therefore each Summarization Unit SU_m must correspond to a GoP g_m in the base layer and the results of analysis for g_m will be used as constraint in the adaptation of SU_m.

5.1. Storyboards

Storyboards are generated using spectral clustering[28]. Only I frames are analyzed and considered as candidates for keyframes in the storyboard. Thus, there is no possible mismatch between analysis and generation of the summary, as in the previous model only I frames (lowest temporal version) are selected when a storyboard is generated.

The first stage consists of the selection of a number of candidates from all the GoPs. Let consider M GoPs forming the input sequence V. The GoPs are initially parsed and grouped into R shots. In order to do it efficiently, the shot detection algorithm is computed directly over the compressed domain, using a modification of the metric proposed in [29], which is based on the variation of the 16x16 and 4x4 intra prediction modes between consecutive I frames. In contrast to [29], in our algorithm the analysis is performed at GoP level rather than at frame level, and using an adaptive threshold rather than a constant one. For each I frame, the macroblock map is partitioned into 3x3 blocks B_i (see Fig. 6), and the difference between two successive GoPs g_{m-1} and g_m is given by

$$\delta\left(g_{m-1}, g_m\right) = \sum_{\forall i} \left| \sum_{j \in B_i} \text{mode\_4x4}^j_{g_{m-1}} - \sum_{j \in B_i} \text{mode\_4x4}^j_{g_m} \right|$$

with $\text{mode\_4x4}^j_{g_m} = 1$ if the prediction of the macroblock j of the I frame of g_m is 4x4, and $\text{mode\_4x4}^j_{g_m} = 0$ if the prediction is 16x16. A shot change is declared if $\delta\left(g_{m-1}, g_m\right) > th_\delta$, where the adaptive threshold is computed as $th_\delta = \bar{\delta} + \beta\sigma_\delta$ making use of the mean $\bar{\delta}$ and standard deviation σ_δ of a sliding window (15 GoPs in the experiments). After the shot boundaries are determined, only those shots longer than a minimum length are considered relevant. Each remaining shot is sampled linearly in time, and a number of keyframes are selected from the I frames of its GoPs (from one up to three keyframes for long shots). Each keyframe τ_p is characterized by the MPEG-7 color layout descriptor[30] after decoding the frame.

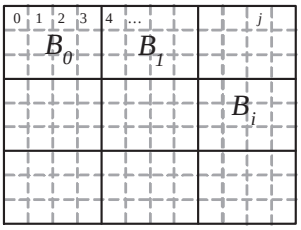


Figure 6: Macroblock map and partitions for shot change detection.

The video sequence is structured into R shots and represented by the MPEG-7 color layout features of P keyframes. Redundancies in the set of keyframes are exploited using a clustering algorithm. A popular algorithm is spectral clustering[28], which is based on graph theory and on the concept of similarity rather than distance. The nodes in the graph are the set of keyframes, and an affinity matrix A collects the similarity between any two nodes in the set of keyframes. The matrix A is computed as:

$$A_{ij} = e^{-\frac{d^2(\tau_i, \tau_j)}{2\sigma^2}}$$

where $d\left(\tau_i, \tau_j\right)$ is the distance between the features of τ_i and τ_j and σ is a scale parameter. We adapted the spectral clustering algorithm proposed in [28], following these steps:

1. Form the degree matrix D, defined as a diagonal matrix with $D_{ii} = \sum_j A_{ij}$, and the laplacian matrix $L = D^{-1/2} A D^{-1/2}$.
2. Perform Singular Value Decomposition and estimate the number of clusters K using the criterion proposed in [15].
3. Select the K eigenvectors $\{e_1, e_2, \ldots, e_K\}$ of L associated with the K largest eigenvalues.
4. Form the matrix Z by stacking the eigenvectors in columns and compute Y from Z by normalizing each row as $Y_{ij} = {}^{Z_{ij}}/\left(\sum_j Z^2_{ij}\right)^{1/2}$.
5. Cluster the rows of Y into K clusters using the K-means algorithm.
6. For each cluster c_k, select the closest keyframe τ^* to its centroid as its representative. This representative τ^* belongs to a certain GoP; the index of this GoP is denoted by m^*_k.
7. Build a list $\texttt{list}_{\texttt{sb}} = \left\{m^*_0, m^*_1, \ldots, m^*_{K-1}\right\}$ with the indexes of the GoPs including the representative keyframes of the clusters.

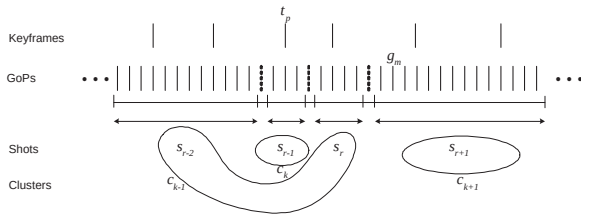


Figure 7: Structuring the sequence for summarization.

After spectral clustering, the sequence has been structured hierarchically into clusters, shots and GoPs (see Figure 7), and the indices saved in $\texttt{list}_{\texttt{sb}}$ represent a storyboard summary. The summarization constraint for extraction is

$$tlevel\,(m) \;=\; \begin{cases} 0 & m \in \texttt{list}_{\texttt{sb}} \\ -1 & otherwise \end{cases} \qquad m = 0, 1, \ldots, M-1$$

5.2. Video skims

Modifying slightly the algorithm used for storyboard summarization, it can also be used to obtain video skim summaries. A simple but effective video skim can be obtained concatenating short excerpts instead of single frames. Each excerpt $\boldsymbol{b}_k = \left\{m_k^* - \frac{N_{exc}}{2}, \ldots, m_k^*, \ldots, m_k^* + \frac{N_{exc}}{2} - 1\right\}$ represents a segment of fixed length of N_{exc} GoPs (with N_{exc} even), and it is centered around m_k^*. Note that m_k^* is selected in the middle of a shot or sufficiently far from its boundaries. Therefore, for resonable values of N_{exc} (e.g. the equivalent in GoPs to 1-2 seconds) there is no risk of selecting GOPs beyond the boundaries of the shot. The last step in the previous algorithm is then replaced by

7. Build a list $\texttt{list}_{\texttt{sk}} = \{\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_{K-1}\}$ with the indices of the GoPs representing segments around the representative keyframes of the clusters.

In this case, the summarization constraint is

$$tlevel\,(m) \;=\; \begin{cases} t_{max} & m \in \texttt{list}_{\texttt{sk}} \\ -1 & otherwise \end{cases} \qquad m = 0, 1, \ldots, M-1$$

6. Experiments

The main advantage of the proposed framework is the efficient generation of adapted summaries. Thus, the experiments are directed to evaluate the performance in terms of efficiency. For comparison, we also provide experimental results with an alternative approach based on transcoding. We used the reference software JSVM 9.8 in the simulations. The test sequence is an excerpt of 10001 frames from the sequence *news12* (CIF resolution and 25 frames per second) from the MPEG-7 Content Set[31]. The sequence was encoded in SVC with 2 spatial levels and 2 quality levels, using CGS for quality scalability, with one base layer and three enhancement layers. The details of these layers are shown in Table 1. Dyadic hierarchical structures were used for temporal scalability with GoP lengths ranging from 1 to 32 frames (1 to 6 temporal levels). In order to compare the approach with a non scalable approach, the sequence was also encoded with H.264/AVC with the same settings of the layer 3 in Table 1.

The framework was studied for two modalities of summaries: storyboards and video skims. For each summary, we first computed the summarization constraint to be used, using the summarization algorithms described previously. For a given summary, the same summarization constraint was used for every method tested, in order to have the same selected frames. Figure 8 shows the summarization constraints, for both storyboard and video skim, in the test sequence coded with a GoP length of 8 frames.

6.1. Test conditions

In the experiments we considered two target conditions to test the performance of the framework:

- *CIF@25.* Both spatial and temporal resolutions do not change with respect to the original bitstream. Therefore, neither spatial nor temporal adaptation will be required, and only efficiency in the generation of summaries is studied.
- *QCIF@12.5.* In this scenario there is adaptation in both spatial and temporal resolutions. Both generation of the summary and adaptation to the target conditions are studied.

The summaries are generated and adapted to the test conditions with the following methods (see Table 2):

- *AVC transcoding.* In this approach, the sequence is first decoded to YUV format. The

Layer Number	Spatial resolution	Temporal resolution	Quality resolution (QP)
0	QCIF	25 Hz	38
1	QCIF	25 Hz	32
2	CIF	25 Hz	38
3	CIF	25 Hz	32

Table 1: Settings of the layers for SVC encoding

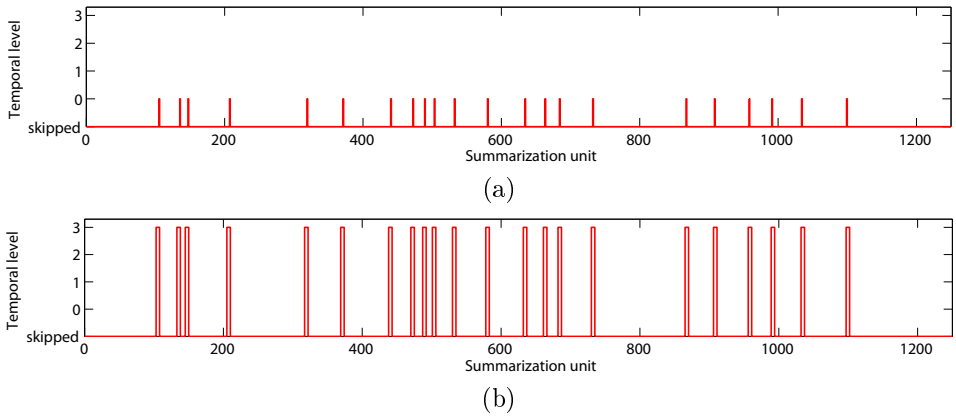


Figure 8: Examples of summarization constraint $tlevel\,(m)$ for the test sequence: a) storyboard, b) video skim.

summary is generated and adapted (if required) into another YUV sequence, which is finally encoded to H.264/AVC.

- *SVC extraction.* It uses the SVC bitstream extractor to select the required packets and to generate the adapted summary.

- *AVC extraction.* The same bitstream extractor is used for this case. This method can be used only when neither spatial nor quality adaptation are required.

- *AVC hybrid.* This method complements the previous one, as the summary is first generated using AVC extraction, and then it is transcoded. Note that, compared to transcoding, only a few frames (depending on the length of the summary) are transcoded, as most of them were discarded during extraction.

For AVC transcoding and AVC hybrid methods, the settings of the encoder were modified to reduce significantly the computational burden due to encoding, and specifically, due to motion estimation. Thus, a fast search method was used with a smaller search range (8 pixels).

6.2. Results

In the first experiment, the different methods are compared for several modalities and summary lengths, ranging from the empty to the whole sequence. The test sequence was encoded with a GoP of 8 frames. As expected, transcoding is much slower than methods based on extraction. As expected, SVC extraction and AVC extraction have good performance. The latter is about two times faster, as the extractor needs to parse and process packets from a single layer, instead of the four layers as in SVC extraction.

Methods based on extraction also have an almost constant performance for all the summary lengths. Methods based on transcoding are more sensitive, but still very constant, with the length of the summary. Most of the processing time in transcoding is due to decoding, as encoding complexity was reduced and only a few frames are encoded in contrast to the decoding of all frames. However, it can be noticed a significant increment of the processing time for longer summaries. The hybrid method performs very fast for storyboards with few frames, but the processing time increases rapidly for longer summaries, because of transcoding.

The generation of a summary with the 0% of the

Method (resolution)	Spatial resolution (input/output)	Temporal resolution (input/output)	Description
AVC transcoding (CIF@25)	CIF/CIF	25/25 Hz	Decoder + Adapter +
AVC transcoding (QCIF@12.5)	CIF/QCIF	25/12.5 Hz	+ Encoder
AVC extraction (CIF@25)	CIF/CIF	25/25 Hz	Extractor
AVC hybrid (QCIF@12.5)	CIF/QCIF	25/12.5	Extractor + Decoder + + Adapter + Encoder
SVC extraction (CIF@25)	CIF/CIF	25/25 Hz	Extractor
SVC extraction (QCIF@12.5)	CIF/QCIF	25/12.5 Hz	
SVC extraction (CIF@25 low)	CIF/CIF	25/25 Hz	
SVC extraction (QCIF@12.5 low)	CIF/QCIF	25/12.5 Hz	

Table 2: Methods and cases used in the experiments.

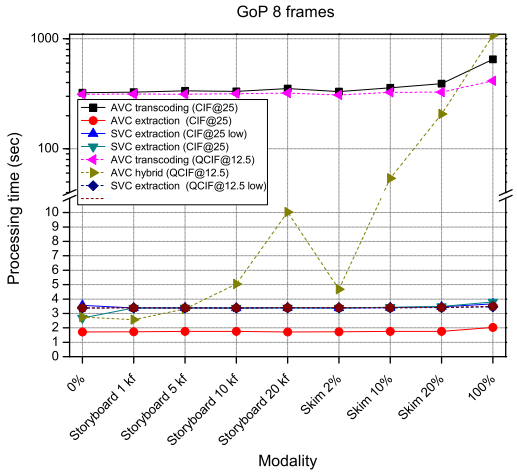


Figure 9: Processing time for different modalities. Note that half of the vertical scale is linear and the rest is logarithmic.

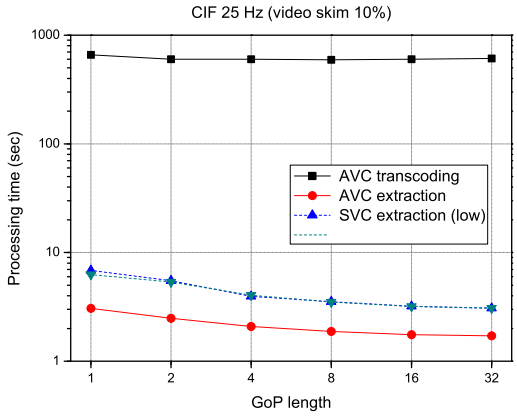
frames in the sequence (an empty summary) is very useful to have a reference of the time used in initialitation and other processes independent of the length of the summary. In transcoding, this time is due to the decoding of all frames, and it is the most important contribution to the overall processing time. In the case of extraction, the JSVM extractor performs the extraction in two passes. In the first pass, all the NAL headers are parsed in order to obtain a description of the bitstream, which is then used to perform the actual extraction. As it can be seen in the figure, most of the extraction time is used in this first pass.

An important parameter in the framework is the SU length (GoP length in the experiments), as it is related to the precision in analysis, coding efficiency and also processing time in the generation of the bitstream. Fig. 10 shows the results for a video skim (10% of the total length) with the different methods tested. It shows that the processing time is almost constant with the GoP length, for both transcoding and extraction.

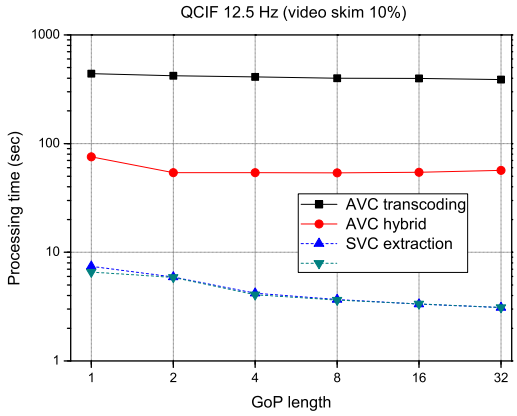
As experiments have shown, a simple solution based on extraction has better performance than others based on transcoding, for the purpose of video summarization. A hybrid solution based on both extraction and transcoding can also be useful when no spatial nor quality scalability are available, specially for short summaries such as storyboards. It must be noted that both transcoder and extractor are not optimized, so the results shown in the experiments can be further improved with optimized software. However, we think that they still provide a fair comparison, and show better results with extraction, which is inherently more efficient than transcoding. In the case of the JSVM extractor, an important amount of processing time is spent in an initial parsing of the whole bitstream. Storing previously this information as a bitstream description can improve significantly the efficiency of the extractor[32].

7. Conclusions

This paper discussed the use of SVC for summarization purposes, based on the consideration of video summarization as content adaptation, where the structure of the frames in the sequence is modified. With some restrictions due to prediction relationships between frames, the same SVC extraction approach can be used for this structural adaptation.



(a)



(b)

Figure 10: Dependency of the processing time with the GOP length : a) CIF 25 Hz, b) QCIF 12.5 Hz.

The summarization is then included in the adaptation framework via additional constraints in the extraction process. These additional constraints enable the description of a summary in terms of summarization constraints. This model of summarization constraints is flexible enough to cover several summarization modalities, such as storyboard and video skims.

The use of an integrated approach has several advantages. One is that the generation of the summary has all the advantages of bitstream extraction in terms of efficiency. Another advantage is the adaptation, in the same process, of the summaries to a specific usage environment, using the layered approach of SVC. Experimental results confirmed these advantages in two different adaptation cases.

References

[1] M.M. Yeung and B.-L. Yeo. Video visualization for compact presentation and fast browsing of pictorial content. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(5):771–785, Oct. 1997.

[2] S. Pfeiffer, R. Lienhart, S. Fischer, and W. Effelsberg. Abstracting digital movies automatically. *Journal Of Visual Communication And Image Representation*, 7(4):345–353, December 1996.

[3] Hyun Sung Chang, Sanghoon Sull, and Sang Uk Lee. Efficient video indexing scheme for content-based retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(8):1269–1279, Dec. 1999.

[4] N. Dimitrova, H.-J. Zhang, B. Shahraray, I. Sezan, T. Huang, and A. Zakhor. Applications of video-content analysis and retrieval. *IEEE Multimedia*, 9(3):42–55, July-Sept. 2002.

[5] X. Zhu, A.K. Elmagarmid, X. Xue, L. Wu, and A.C. Catlin. Insightvideo: toward hierarchical video content organization for efficient browsing, summarization and retrieval. *IEEE Transactions on Multimedia*, 7(4):648–666, Aug. 2005.

[6] Z. Li, G.M. Schuster, A.K. Katsaggelos, and B. Gandhi. Rate-distortion optimal video summary generation. *IEEE Transactions on Image Processing*, 14(10):1550–1560, Oct. 2005.

[7] Y.-F. Ma, X.-S. Hua, L. Lu, and H.-J. Zhang. A generic framework of user attention model and its application in video summarization. *IEEE Transactions on Multimedia*, 7(5):907–919, Oct. 2005.

[8] M.A. Smith and T. Kanade. Video skimming and characterization through the combination of image and language understanding. In *Content-Based Access of Image and Video Database. Proceedings of IEEE International Workshop on*, pages 61–70, 3 Jan. 1998.

[9] Y. Li, S.-H. Lee, C.-H. Yeh, and C.-C.J. Kuo. Techniques for movie content analysis and skimming: tutorial and overview on video abstraction techniques. *IEEE Signal Processing Magazine*, 23(2):79–89, 2006.

[10] B.M. Wildemuth, G. Marchionini, Meng Yang, G. Geisler, T. Wilkens, A. Hughes, and R. Gruss. How fast is too fast? evaluating fast forward surrogates for digital video. In *Digital Libraries. Proceedings of Joint Conference on*, pages 221–230, 27-31 May 2003.

[11] K.A. Peker, A. Divakaran, and H. Sun. Constant pace skimming and temporal sub-sampling of video using motion activity. In *Proceedings of International Conference on Image Processing*, volume 3, pages 414–417, 7-10 Oct. 2001.

[12] J. Bescos, J. M. Martinez, L. Herranz, and F. Tiburzi. Content-driven adaptation of on-line video. *Signal Processing: Image Communication*, 22:651–668, 2007.

[13] O. A. Lotfallah, M. Reisslein, and S. Panchanathan. Adaptive video transmission schemes using mpeg-7 motion intensity descriptor. *IEEE Transactions On Circuits And Systems For Video Technology*, 16(8):929–946, August 2006.

[14] Z. Gang, L.-T. Chia, and Y. Zongkai. MPEG-21 digital item adaptation by applying perceived motion energy to H.264 video. In *Image Processing, 2004. International*

Conference on, volume 4, pages 2777–2780, 24-27 Oct. 2004.

[15] J. Calic, D.P. Gibson, and N.W. Campbell. Efficient layout of comic-like video summaries. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(7):931–936, 2007.

[16] M. Mrak, J. Calic, and A. Kondoz. Fast analysis of scalable video for adaptive browsing interfaces. *Computer Vision and Image Understanding*, 113(3):425—434, 2009.

[17] C.-W. Ngo, Y.-F. Ma, and H.-J. Zhang. Automatic video summarization by graph modeling. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, volume 1, pages 104–109, 2003.

[18] S.-F. Chang and A. Vetro. Video adaptation: concepts, technologies, and open issues. *Proceedings of the IEEE*, 93(1):148–158, Jan 2005.

[19] A. Vetro. MPEG-21 digital item adaptation: Enabling universal multimedia access. *IEEE Multimedia*, 11(1):84–87, 2004.

[20] I. Ahmad, Xiaohui Wei, Yu Sun, and Ya-Qin Zhang. Video transcoding: an overview of various techniques and research issues. *IEEE Transactions on Multimedia*, 7(5):793–804, Oct. 2005.

[21] G. J. Sullivan and T. Wiegand. Video compression - from concepts to the H.264/AVC standard. *Proceedings Of The IEEE*, 93(1):18–31, January 2005.

[22] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the H.264/AVC standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(9):1103–1120, 2007.

[23] H. L. Wang, A. Divakaran, A. Vetro, S. F. Chang, and H. F. Sun. Survey of compressed-domain features used in audio-visual indexing and analysis. *Journal of Visual Communication and Image Representation*, 14(2):150–183, Jun 2003.

[24] S. De Bruyne, D. De Schrijver, W. De Neve, D. Van Deursen, and R. Van de Walle. Enhanced shot-based video adaptation using mpeg-21 generic bitstream syntax schema. In *Computational Intelligence in Image and Signal Processing, 2007. CIISP 2007. IEEE Symposium on*, pages 380–385, 1-5 April 2007.

[25] L. Herranz. Integrating semantic analysis and scalable video coding for efficient content-based adaptation. *Multimedia Systems*, 13(2):103–118, August 2007.

[26] D. Mukherjee, E. Delfosse, J. G. Kim, and Y. Wang. Optimal adaptation decision-taking for terminal and network quality-of-service. *IEEE Transactions on Multimedia*, 7(3):454–462, June 2005.

[27] I. Amonou, N. Cammas, S. Kervadec, and S. Pateux. Optimized rate-distortion extraction with quality layers in the scalable extension of H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(9):1186–1193, 2007.

[28] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14, 2001.

[29] S. De Bruyne, W. De Neve, K. De Wolf, D. De Schrijver, P. Verhoeve, and R. Van de Walle. Temporal video segmentation on H.264/AVC compressed bitstreams. In *Advances in Multimedia Modeling*, volume 4351 of *Lecture Notes in Computer Science*, pages 1–12. Springer-Verlag Berlin, 2006.

[30] E. Kasutani and A. Yamada. The mpeg-7 color layout descriptor: a compact image feature description for high-speed image/video segment retrieval. In A. Yamada, editor, *Proc. International Conference on Image Processing*, volume 1, pages 674–677 vol.1, 2001.

[31] Description of MPEG-7 content set. Technical report, ISO/IEC JTC1/SC29/WG11 N2467, October 1998.

[32] M. Eberhard, L. Celetto, C. Timmerer, E. Quacchio, and H. Hellwagner. Performance analysis of scalable video adaptation: Generic versus specific approach. In *Proc. Ninth International Workshop on Image Analysis for Multimedia Interactive Services WIAMIS '08*, pages 50–53, 2008.