Content-sensitive Approach for Video Browsing and Retrieval in the Context of Video Delivery: *VBaR* Framework

Phooi Yee Lau and Sungkwon Park
Media Communications Laboratory, Hanyang University, Seoul 133-791, Republic of Korea
E-mail: {laupy,sp2996}@hanyang.ac.kr

Keywords: browsing behavior, corner detection, image processing, video analysis, shot detection, low bit-rate channel, video delivery

Received: August 27, 2009

Information is doubling at a rate of once every few months and the rate of increase is growing. Video and media transport plays an important part of that growth. It has spurred the development of broadband access and slowly gained increasing prominence especially for multimedia-rich Internet contents. Operators are now under pressure to efficiently use available resources for delivering targeted contents in a reliable and consistent manner. The present work proposes a new strategy to select intra-coded frames that best represent the entire video. The proposed approach, tri-step approach, uses low-level image features to select representative key frames which enable random access to any part of the video sequence while avoiding the overhead incurred to transmit periodic intra-coded frames. The first step discriminates non-transition frames by checking adjacent frame characteristics using low level features. The second step refines our selection of key frame by dropping non-commonly browsed frames, being the non-informative frames. The final step verifies the remaining frames if they are far apart in time to maintain the efficiency of video delivery over low bit rate channel. Thirteen video sequences, obtained from MPEG-4 industry forum, are used in the experiments. A framework, named the Video Browsing and Retrieval (VBaR), is developed and it allows user to analyze input videos at their full control. Experimental results, using the proposed framework, show the ability to effectively select intra-coded frames in video delivery.

Povzetek: Predstavljeno je kontekstno odvisno indeksiranje in iskanje video vsebin.

1 Introduction

Today, streaming videos became a popular medium of entertainment and advertisement, with many websites providing direct links to videos from all over the internet. Viewers can now stream videos through a simple and searchable interface. What actually attracts the viewer's interest? Research shows that it is important to track the browsing behavior of viewers, both within and across videos, to obtain important clues to the effectiveness of a videos e.g. to an advertisement or to a trailer video preview [1-3]. Browsing behavior can be revealed by the choice of control used, e.g. fast-forwarding or pause, or by the enhanced user area which allows viewers to rate their video links and states their favorites.

Viewers, nowadays, have the privilege to download videos according to their preferences which sometimes, may require constant interactivity. Among the existing pressing issues resulted from the interactivities are: 1) the ability to playback or pause a video, and 2) the ability to guarantee transmittable and playable video using available resources. Because a video is essentially a collection of still images, presumably long (e.g. 120 minutes), one cannot tell what it is about without watching the whole video. Nonetheless, there has already been a wide spread of research interest in the delivery of selected content to any users, e.g. content-based video analysis or segmentation which has been intensively studied since the past decade [1-6]. In 1999, He et al. proposed a simple tracking of video usage by using server logs, which keeps information about the segments watched by viewers, to generate the summary of viewing behavior [5]. In 2001, Syeda-Mahmood presents a framework to continuously track viewers through HMMs, by observing their interaction with video based on deducing interest of viewers, a rather unusual approach [1]. Both methods studied the viewer's interest, irrespective of the video content or scene. At the content level, Zhu et al. present a system developed for content-based video browsing and retrieval, integrating audio-visual as well as text information and natural language understanding techniques analysis to extract scenes and content information of video documents, and to organize and classify video scenes [2]. In 2002, Chen and Yang presented an MPEG4 simple profile compatible approach for video browsing and retrieval over low bit rate channel, whereby, a new stream is generated from the video server to enable random access [6]. This method is oriented towards the viewers' browsing requirements, i.e. according to video content and the channel characteristics, so that transmission overhead could be concealed. These two later methods, though studied video contents, ignore the viewer's interest with respect to the contents. Most importantly, these works show that there is a need to study viewer's interest and video content to enable video to be streamed across in the shortest possible time.

But, some questions still arise, such as, how to best represent the underlying content? Due to the need to have an efficient playback system, extracting representative key frames to describe the contents of a video will play a fundamental role for many video applications. Key frames are often used to define the starting and ending point of smooth transition; i.e. frames that best represent the underlying content. Key frames can be sampled randomly or uniformly at some definite time intervals. The main drawback of uniform sampling, while easy to implement, is that it may cause some important short clips without representative key frames while longer clips might have multiple frames with similar content, thus failing to represent the actual video [7]. One of the most popular ways to extract key frames is by adapting to the dynamic video content. Shot-based key frame extraction segments a video within a continuous period and uses the first frame of each shot as key frames. It heavily depends on the temporal dynamics in the scene. In this case, though the selected key frames can represent the entire video, it may miss the important part of video as it is not possible to select key frames that can represent the video content well [8-10]. On the other hand, we know that content-based video analysis has been studied intensively for the past decade to support a variety of applications, including video indexing and browsing [11-13]. For example, shot-based video segmentation is used to provide abstraction and delineation for video indexing, browsing and retrieval [14].

But, in video delivery, not only do we need to consider the above stated requirements, we also need to consider how to deliver videos in the shortest time in order to maximize available resources, especially videos that are simultaneously viewed by many subscribers or high quality multimedia contents such as HDTV and 3DTV. Therefore, video delivery services need to be natural, intuitive, and guided by user's viewing interest. It is, thus, important to optimally select intra-coded frames, e.g. incorporating viewer's browsing patterns and viewer's interest into the selection process, since the largest part of traffic growth over the next decade will be associated with video delivery [11, 15]. Failure to do so will prove to be very capital-intensive as service operator may be force to purchase new infrastructure components. One of the solutions is to represent the content of video using key frames and these key frames should be able to 1) index videos to help search for a particular scene 2) automatically identify user preference through preference modeling, 3) facilitate automatic movie content summarization.

In this paper, a new strategy, tri-step approach, is used to select intra-coded frames that best represent and describe the entire video well. The first step uses low-level image processing techniques used for shot detection (scene change); the second step, uses low-level image processing techniques to classify key frames into *informative* and *non-informative* (common browsing behavior); and the third step, uses temporal constraints to enable distinct distribution of key frames spanning the entire video (decoder limitations). The method is tested under two experimental set-ups: 1) different video sequence: sports video (with motion) and news (less motion), and 2) different scenario: multiple-scene video (abrupt scene change) and single-scene video (gradual scene change). The purpose is to allow user randomly access any part of the video sequence while avoiding the overhead incurred in transmitting periodic intra-coded frames, thus, maximizing the resources available.

The remainder of this paper includes: Section 2 that describes and discusses the proposed approach and algorithms for selecting intra-coded frames; Section 3 that outlines the Video Browsing and Retrieval (*VBaR*) framework; Section 4 that evaluates several experimental results; Section 5 that discusses and concludes the paper with future work.

2 Analysis of Content-sensitive Frames for Video Streams – Tri-Step Approach

A video can be considered as being made up of numerous snapshots, called frames or picture. The volume generated by digitizing all frames is too large for the video delivery channel. Among the much used video compression standards, aimed to reduce the amount of data required to store or transmit video while maintaining an acceptable level of video quality on low-bit-rate channels, are the ISO MPEG4 Part 10 of MPEG4 and ITU-T H.264. Low bitrate channels are constraint by a few important characteristics: 1) prevent transporting video frames that takes up resource, and 2) eliminate redundant data to be delivered to prevent channel congestion. Let us look at the role of image coding in video delivery. Intra-coding, often used to enable random access, refers to the individually compressed image without any reference to the other frames. On the other hand, the compression performance could be further improved when the temporal redundancy in video sequences is exploited. Known as the inter-coded frame, this coding refers to a frame that is coded based on some relationship with adjacent frames, i.e. proposed to exploit the interdependencies between adjacent frames. In reality, transmitting intra-coded frames (I-frames), compared to inter-coded frames (P-frames or B-frames), will increase the bit-rates greatly. Therefore, for video streaming in low bitrate channel, transmitting inter-coded frames are more favorable. In general, these three pictures types (I-, P-, and B-frames) are encoded with a group of pictures (GOP) length in the general reference encoder [16].

In a video itself, there are two potential issues which will affect the quality of video received. One is the delay in packet delivery which may prevent the video being played smoothly, often referred as jitter or frame reversal. If jitter exceeds the buffer size in a device, video quality will degrade noticeably. The second is dropped frames. During congestions, significant numbers of packets are dropped; inter-coded frames are dropped first, followed by intra-coded frames, and this may also cause a noticeable degradation in video quality. In reality, network bandwidth is usually time-varying. If a user constantly searches a video for interesting video clips, then adapting a video to the start of all interesting clips could minimize video traffic as "watch" only frames are delivered. For example, Lee's work [15] adaptively assigned intra-coded frames by considering the scene changes and rapid motion in video sequences. Such type of strategy, i.e. placing intra-coded frames strategically to improve coding efficiently, is receiving increasing attentions in the video research community [17-18]. These works, for example, discuss how to efficiently place inter-coded frames for variety of video clips, especially those that do not contain frequent scene change. It shows that there is a need to reduce bandwidth consumption, especially crucial during peak-hours. It could be accomplished by avoiding the delivery of "non-watched" video data units to the set-top-box (STB), especially if users often quit video sessions prior to its completion. As we know, contents that are requested on demand, i.e. stored video contents, has recently emerged as a new business model. This business model redefines the subscriber-provider relationship, i.e. delegating content selection to the customer while the service provider manages the content distribution. As roughly 40% of sessions contain some interactivity, certainly there is a pressing need to deliver video data units efficiently in order to reduce performance bottlenecks, long delays and poor user experience for subscribers..

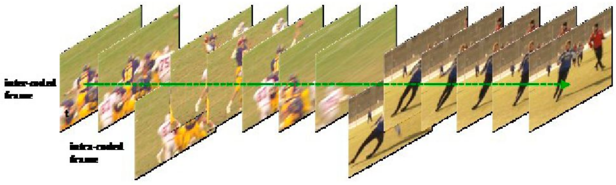


Figure 1: Input sequence: illustration of key frames for a video sequence.

The primary objective of this paper is to present a tool for selecting intra-coded frames that best represent and describe the entire video well. The proposed approach, using a tri-step approach, are able to select intra-coded frames by analyzing the video content for interesting scene that would attract viewers – see Figure 1. Our motivation is to enable user to retrieves requested video clips efficiently, without delivering too many redundant frames. We satisfy the following criteria in our approach: 1) reduce redundant frames that viewers will receive due the start of the clip is far from the periodically intra-coded frames; and 2) allow viewers to make browsing selection because a full video clip can themselves be long and needs to be segmented; and 3) adjust the video sequence to fully grasp the scene change, depending on the content, thus saving bandwidth and storage.

The proposed technique is divided into 3-step. The first step provides an efficient discrimination of input videos to select scene change frames.. The second step proceeds with a further evaluation of the selected frames in *Step 1*, verifying if these frames correspond to a set of common browsing behavior, i.e. *informative* frames. The final step studies the decoder limitation and verifies if the remaining frames are far apart in time to maintain the efficiency of video delivery over a low bitrate channel, i.e. determining the start frame of interesting video segments to enable distinct distribution of these frames spanning the entire video.

2.1 Step 1: Discriminate non-shot transition frames as candidate key frames

There are two type of scene change: 1) abrupt transition, which corresponds to a sudden change between two consecutive frames, and 2) gradual transition, which corresponds to a small change throughout a number of frames, detecting a transition frame could mean detecting the precise frame when the changes happen. The simplest feature that indicates a scene change is with low level features. They can be reliably used to indicate the starting position of a change in video sequences for shot boundary studies. The low level features applied in this paper are hue, saturation, value, and corners.

2.1.1 Preliminary Analysis

As discussed above, most images present high relativity with regards to some of its basic features except when scene change occurs, due to the presence of a new scene. This paper extracts the hue (H_F), saturation (S_F), value (V_F), and corner (C_F), as the set features to determine a scene change. Color saturation, hue and value can be easily obtained by converting the input videos to the HSV color space. The RGB color space is fundamentally different from the HSV color space as it separates the luminance from the color information (chromaticity) – see experimental results in Figure 2. Therefore, RGB color space image has to be converted to HSV color space by normalizing the RGB values – see equation (1). The H component, S component and V component can be obtained using equation (2), equation (3), equation (4) and equation (5), respectively.

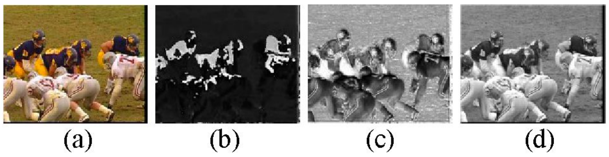


Figure 2: An input image from 'football' sequence: (a) Original CIF (352 x 288) video frame; (b) HSV - Hue feature; (c) HSV – Saturation feature; and (d) HSV – Value feature.

$$r = \frac{R}{R+G+B},\quad g = \frac{G}{R+G+B},\quad b = \frac{B}{R+G+B} \qquad (1)$$

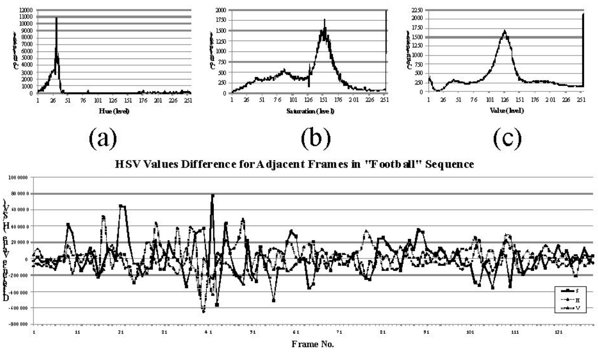
$$H = \left(\cos^{-1}\left\{ \frac{0.5\times\left[(r-g)+(r-b)\right]}{\left[(r-g)^2+(r-b)(g-b)\right]^{\frac{1}{2}}} \right\} \right)\times\frac{180}{\pi}, if\ b \le g \qquad (2)$$

$$H = \left(2\pi - \cos^{-1}\left\{ \frac{0.5\times\left[(r-g)+(r-b)\right]}{\left[(r-g)^2+(r-b)(g-b)\right]^{\frac{1}{2}}} \right\} \right)\times\frac{180}{\pi}, \text{if b > g} \qquad (3)$$

$$S = \left(1 - 3\times\min(r,g,b)\right)\times 255 \qquad (4)$$

$$V = \left(\max(r,g,b)\right)\times 255 \qquad (5)$$

The hue (H) varies from 0° to 360°, and is here quantized into 12 color intervals, each spanning 30°, [red, orange, yellow, yellow-green, green, green-cyan, cyan, cyan-blue, blue, blue-magenta, purple and magenta-red]. Saturation, S, is the intensity of specific hue, whereby, highly 'attractive' areas typically have vivid colors; therefore the color saturation is high. The V, also known as value, is also used as it allows selecting the highest pixel values and visually corresponding to brighter image areas. The HSV value seems to be a good cue to discriminate the non-short transition frames as they almost do not change with respect to small scene change. The HSV value that is computed for each video frame and the difference of these value based on adjacent frames are plotted onto chart – see Figure 3 (a/b/c) and Figure 3 (d), respectively. Later, a threshold is applied for each frame to determine if there are the scene changes.

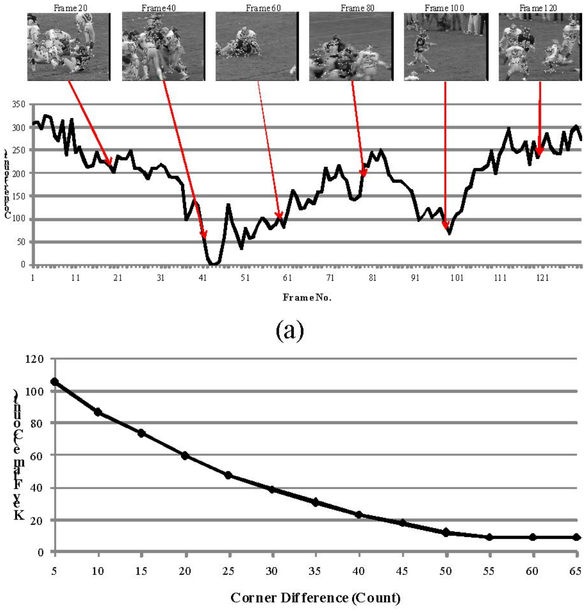


(a) (b) (c)

(d)

Figure 3: An input image (Figure 2) pixel count for: (a) Hue-, (b) Saturation-, (c) Value-level, and (d) Difference between adjacent frames for HSV value.

Corners have been traditionally used to detect or track motion. Here, we used it to assist us in tracking transition frames. This paper adopts a curvature-based corner detector which detects both fine and coarse features accurately at low computational cost [5]. It utilizes global and local curvature properties and balances their influence when extracting corners, allowing different parameters to be automatically determined for different images, different curves, and different kinds of corner candidates. The corner detector's step-by-step details can be found in [5] and experimental results are shown in Figure 4. Figure 4 (a) shows the corners count for "Football" sequence with sample frames. The number of corners detected for each consecutive frame is later threshold to determine if there is a scene change. Figure 4 (b) shows the number of key frames selected using different threshold values for the "Football" sequence.



(a)

(b)

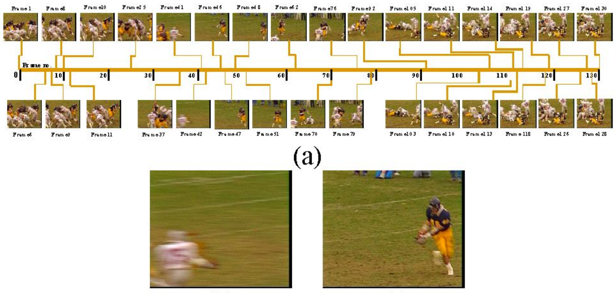
Figure 4: (a) Corners count for "Football" sequence with sample frames (b) Total key frames selected with different corner difference value.

2.1.2 Shot detection and elimination of frames

From exhaustive tests, it was observed that a scene change incurs more abrupt changes in the HSV values and the corner count, and is illustrated in Figure 3 (d) and Figure 4, respectively. In practice, one feature alone could not identify clearly the position of a scene change. Due to this, a more reliable shot detection can be obtained by combining the results coming from a set of features, discussed earlier in subsection 2.1.1. Votes are taken from each feature which favours the scene change detection and decisions are made based on a majority vote, according to equation (6). Experimental examples are shown in Figure 5.

$$F_{Step1} = \begin{cases} Cut: & \text{> two vote from } \mathrm{H_F S_F V_F C_F} \\ Non-cut: & otherwise \end{cases} \qquad (6)$$

Video frames that are classified as transition frames will be kept for further processing. This initial discrimination is conducted to allow fast analysis of a complete video and to discard non-scene transition key frames. Figure 5 shows key frames selected in *Step 1* for "Football" sequence. Notice, however, some frames shown in Figure 5 (a) are either appearing too close together in time or do not provide sufficient information about the scene, even though they represent a distinctive scene change. These frames will be further verified in the subsequent step and its initial classification revised.



(a)

(b) (c)

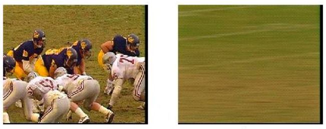
Figure 5: (a) Candidate key frames selected in *Step 1* (b) Frame 42 (c) Frame 47.

2.2 Step 2: Discriminate *non-informative* candidate key frames

Tracking the browsing behavior of viewers are valuable, not only for forecasting future visit patterns for content-sensitive e-commerce, but are also useful in the generation of fast previews of videos for easy pre-download [20-22]. If we could learn the center of interest to which the viewers are attracted, it could be used to help viewers find and select appropriate content from the vast amount of data available. Viewers often look for ways to quickly grasp the content by visual fast-forwarding and the ability to determine each interesting clip would enable distinct browsing states, distributed over the entire video. Whole video, which is sometimes too long, needs to be segmented into shorter and more interesting segments. These segments are often determined by the statistical inference over extensive historical samples, i.e. preferred content or preferred video clips. The basic idea is to evaluate the content using low-level features to enable the user to grasp the potential knowledge about the content to be browsed. But, what content would represent an interesting video clips?

Syeda-Mahmood's work [1] grouped three browsing behaviors by summarizing various potential states that viewers could be in: 1) curious, 2) aimless browsing, and 3) explicit queries/search [1]. The first two states are viewers with no specific agenda and they generally do not capture the actual browsing behavior i.e. passive viewers. The third state requires urgency and viewers tend to look for something intriguing, which often state the browsing patterns – active viewers (age under 40s). To assist these active viewers, the authors provide video abstracts, i.e. representative key frames, to represent the video's content (aka summary shots), to denote each shot. They then reclassify these shots into two categories – interesting and mundane. So, the question is how we can classify interesting shot, beyond the browsing behavior? Rich content shots, e.g. shots which include many details and garner most viewers' attention are considered interesting – see Figure 6.

The selection of interesting shots may be associated with the search for content which could elicit viewer's behavioral patterns in browsing. We relate rich content shots, i.e. interesting video frames, with image details which could be obtained through low-level image processing techniques. For example, we obtained the rough contour of objects, which strongly relates to the image content itself, by studying the edges and the corners of an input image. But analyzing the whole image could not specifically represent the content details. As such, here, we adopted the block based approach. The following describes how interesting frames (aka *informative*) can be extracted.



(a) (b)

Figure 6: Sample images – (a) rich content shot, and (b) and mundane shot.

At first, a block-based analysis of the image's edge and corner is performed. Each image is divided into 16 x 16 pixel blocks, and for every block, B_i, the edge, $I_E(B_i)$ and corner, $I_C(B_i)$ are calculated. The images are discriminated into *informative* and *non-informative* using a 5-level classification system. Each level is represented using a gray level code, as shown in Figure 7. The decision on the *informative* level can be expressed as follows:

- Level 0 (Non-*informative* block):
  If $I_E(B_i)| I_C(B_i) =0$
- Level 1 (Low *informative* block):
  If $0 < I_E(B_i) <= n$ or $0 < I_C(B_i) <= m$
- Level 2 (Average *informative* block):
  If $n < I_E(B_i) <= 2n$ *or* $m < I_C(B_i) <= 2m$
- Level 3 (High *informative* block):
  If $2n < I_E(B_i) <= 3n$ or $2m < I_C(B_i) <= 3m$
- Level 4 (Extreme *informative* block):
  If $I_E(B_i) > 3n$ or $I_C(B_i) > 3m$

Informative level	0	1	2	3	4
Gray level	■	▓	▒	░	□

Figure 7: Representation of *informative* level using 5 different gray levels.

m and *n* is a margin of safety, here set to 1 and 10, based on extensive experimental testing. *k* is the total block in a video frame. The image is then discriminated into the *informative* (*non-informative*) if more than (less than) 10 blocks are *Level 3* - see equation (7). Candidate key frames from *Step 1* are further classified into *informative* and *non-informative* frames. *Informative* frames are selected as *Step 2* candidate key frames and will be further classified and analysed – see Figure 8.

$$F_{Step\,2} = \begin{cases} Informative: & if(sum(B_i > Level2) > 10) \\ Non-informative: & otherwise \quad \mathrm{i} = 1,2,3\ldots\mathrm{k} \end{cases} \qquad (7)$$

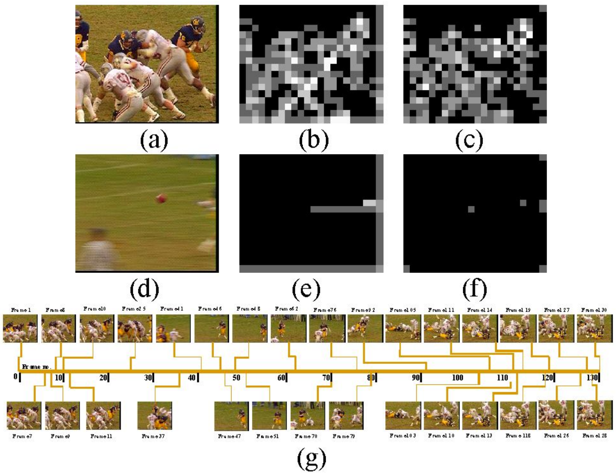


Figure 8: Input images (a and d) with its respective *informative* level - edge blocks (b and e) and - corner blocks (c and f), and (g) *Step 2* candidate key frames after discriminating non-*informative* frame.

2.3 Step 3: Applying temporal constraints to verify selected key frames

Only frames labelled as *informative* (in *Step 2*) are further analyzed in *Step 3*. This final step verifies if the remaining frames occur far apart in time to maintain the efficiency of video delivery over low bitrate channel. This is because there is a significant overhead associated with the transmission of periodic intra-coded frames, as intra-coded frames typically require 5-10 times as many bits as inter-coded frames [23-24]. For videos, during normal speed playback, intra-coded frames do not provide additional functionality, and this overhead should be avoided. Conversational applications, on the other hand, do not require frequent transmission of intra-coded frames, and often placed infrequently to allow bandwidth saving.

It is typical to have at least two intra-coded frames per each second of video in order to allow decoder to begin decoding with sufficient frequency. For example, we have an intra-coded frame every 15th frame on 29-30Hz systems, or every 12th frame on 24-25Hz systems, insinuating that a transition frame could take place in a window of 10-30 frames. If an intra-coded frame (after *Step 2*) surpasses this window, which may represent a gradual transition between the two shot, and if channel error propagation occurs, the inability to correct the error due to the unavailability of intra-coded frame will degrade the video quality at the receiver. On the other hand, more intra-coded frames are needed for a more frequent and pronounced scene change activity in a video sequence.

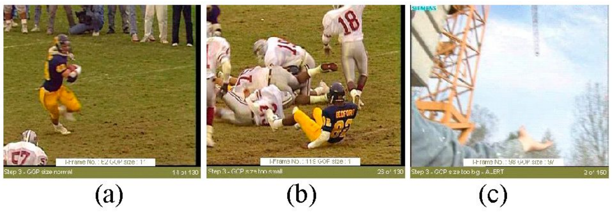


Figure 9: Examples of *Step 3* final key frames selection: (a) select (b) dropped (c) insert additional key frames.

Here, we proposed using temporal constraint to limit the GOP size (or window), a minimum of five to a maximum of thirty, using one-per-window (intra-coded frame) policy. This window allow us to select representative key frames that describe the content of the video, whether in abrupt scene transition or in gradual scene transition. Figure 9 (a) shows that frame number 14 (*Step 2*) or video frame number 62 has a GOP of 11 while Figure 9 (b) shows that frame number 26 (*Step 2*) or video frame number 119 has a GOP of 1 – being too small, and Figure 9 (c) shows that frame number 2 (*Step 2*) or video frame number 98 has a GOP of 97 – being too big. In *Step 3*, when the GOP size is too small - as shown in Figure 9 (b) - the intra-coded frame will be dropped. On the other hand, when the GOP size is too big – as shown in Figure 9 (c) – additional intra-coded frame will be inserted based on one-per-window policy. This is to ensure the videos have an intra-coded frame that are placed a second apart in order to control random accesses, at least, to every second – see experimental results in Figure 10. The remaining frames are named key frames.

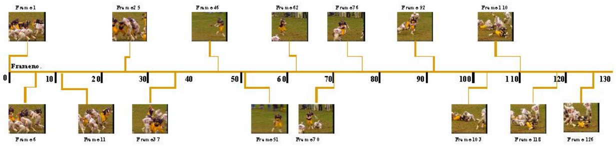


Figure 10: Selected key frames after temporal constraints in *Step 3*.

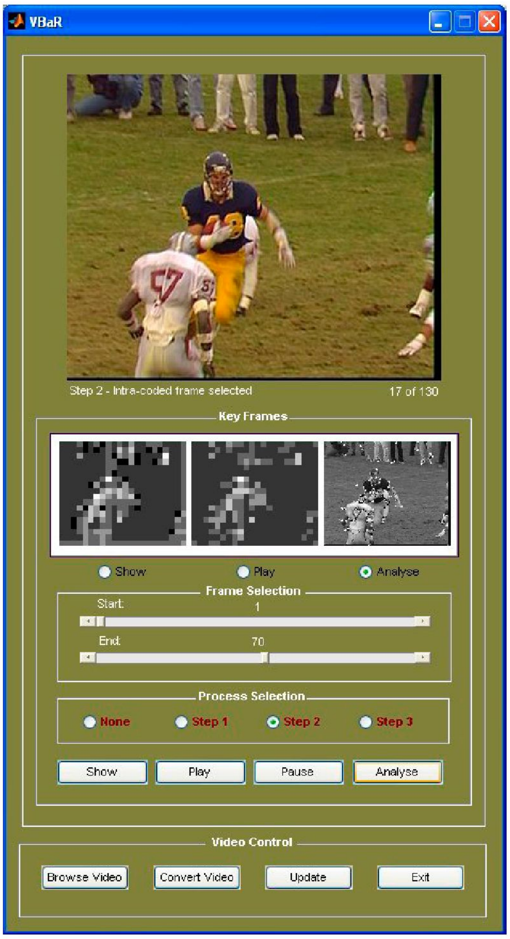
3 Content-sensitive Video Delivery System

3.1 Materials

The analysis was run on a PC platform, using thirteen (13) YUV format test video sequence from MPEG-4 industry forum ("http://www.m4if.org/resources.php"). The dataset is composed of thirteen sequences with different scenarios ranging from news, foreman, tennis, soccer, football, hall, coastguard, harbour, mobile, city skyline, crew, bus, and multiple scenarios - as shown in Table 2. Each selected video sequence has different number of frames and with 352 x 288 (CIF) resolutions.

3.2 *VBAR* Framework

The analysis framework, Video Browsing and Retrieval (*VBaR*) framework, is a research-oriented framework developed in the Media Communication Laboratory at Hanyang University to analyze video delivery performances. The framework was developed using MATLAB®GUIDE tools to achieve a user-friendly interface, as shown in Figure 11. At the moment, the system can analyze targeted videos, i.e. to select a set of representative key frames for a video



Step 2 - Intra-coded frame selected

17 of 130

Figure 11: Graphical user interface for *VBaR* framework.

3.3 Using the GUI

Table 1: Standard operating procedure for *VBaR*

Video Control: Select input video
1) ***Browse video*** for the *.yuv video
2) ***Convert video*** to extract the video

Frame Selection: Select the start and end frame for analysis
1) Select ***Start*** frame by clicking at the bar
2) Select ***End*** frame by clicking at the bar

Control Key Selection: Select the control options
Options: *Show/Play/Pause/Analyze*
Show : Show an image
Play : Play a video sequence from *Start* frame to *End* frame
Pause : Pause the execution
Analyze : Analyze the selected process – *Step 1/Step 2/Step 3*

Analyze Selection: Select the process option
Option: *None/Step 1/Step 2/Step 3*
None : No process
Step 1 : To select shot detection
Step 2 : To eliminate non-*informative* frames
Step 3 : To dropped or insert frames

**Results are shown in the main window and stored to file

The user interface supports the configuration of the desired task in a user friendly way. The input images or video, and the analysis results are shown in Figure 11. The software is able to analyze grayscale and color video sequences. The user interaction with the analysis software using the GUI follows the main steps listed in Table 1.

4 Experimental Results

The proposed algorithm is evaluated using thirteen selected video sequences, each with different number of video frames. The selected video sequences evidence a range of scenarios. Experimental results obtained using *VBaR* framework is presented. In these videos, experimental results are evaluated: 1) key frame analysis, 2) GOP analysis, 3) performance – speed, 4) comparison among different sequences, and 5) performance comparison.

Table 2: Video sequence details

Video 1: News – camera static, slow motion and few scene changes
Video 2: Foreman – camera moving, slow motion and few scene changes
Video 3: Stefan – camera moving, fast motion and many scene changes
Video 4: Soccer – camera moving, fast motion and many scene changes
Video 5: Football – camera moving, fast motion and many scene changes
Video 6: Hall – camera static, slow motion and few scene changes
Video 7: Coastguard – camera moving, slow motion and few scene changes
Video 8: Harbour – camera static, fast motion and few scene changes
Video 9: Mobile – camera moving, fast motion and few scene changes
Video 10: City – camera moving, fast motion and many scene changes
Video 11: Crew – camera moving, slow motion and few scene changes
Video 12: Bus – camera moving, fast motion and many scene changes
Video 13: Multiple scenes – combination of Stefan/Football/Soccer/News

4.1 Key Frame Analysis

This key frame analysis counts the number of frames selected/dropped in each process, e.g. *Step 1*/*Step 2*/*Step 3* – as shown in Table 3, VideoFile 1, VideoFile 2, and VideoFile 3.[1] For example, in *Video 5*, a total of 15 frames were selected as key frames, i.e. that is average GOP size of about 9, compared to *Video 1*, with 10 key frames and average GOP size of about 30. The reason is because *Video 5* has frequent scene change (usual for sports video). Notice that when there are few scene changes – e.g. *Video 1* and *Video 6*, only few key frames are selected in *Step 1* and *Step 2*. *Step 3* compensates this problem by inserting appropriate key frames based on one-per-window policy. In *Video 1*, a total of 6 key frames are inserted in *Step 3* - as shown in Table 3. Scene changes for sports video are more frequent. Due to this, quite a number of frames were selected as candidate key frames in *Step 1* and *Step 2*; with many having a GOP size as small as 1. In this case, the framework automatically dropped candidate key frames when the GOP is smaller than 5. For example, for *Video 3*, out of the 20 candidate key frames, 12 frames are selected as

[1] Please contact first author for VideoFile*.

key frames in *Step 3*; i.e. 8 candidate key frames are dropped.

We notice that *Video 3* and *Video 8* have the smallest average GOP size, even though both videos observe different camera settings and scene change details. In *Video 13*, for example, where multiple scenes change occurs ("Stefan"→ "Football"→ "Soccer"→ "News"), our framework is robust in selecting key frames – see VideoFile4.

Table 3: Step-by-step key frame(s) selection

Video No.	Total Frame	*Step 1*	*Step 2*	*Step 3*	Ave GOP size
Video 1	300	4	4	10	30
Video 2	150	9	9	10	15
Video 3	90	20	20	12	7
Video 4	150	12	12	9	16
Video 5	130	31	30	15	8
Video 6	300	4	4	10	30
Video 7	300	22	22	18	16
Video 8	150	31	31	19	7
Video 9	150	17	17	12	12
Video 10	150	22	22	13	11
Video 11	150	31	31	15	10
Video 12	75	18	18	9	8
Video 13	300	34	34	21	14

To summarize, in order to meet bandwidth and buffer size limit, especially if service providers want to guarantee transmittable and playable video, there exist two important contribution of our work: 1) for video sequences with few scene change, the system compensates by inserting key frames, and 2) for video sequences with frequent scene change, key frames are dropped.

4.2 GOP Analysis

Table 4: GOP size for each video sequence

Video GOP No.	1	2	3	4	5	6	7	8	9	10	11	12	13
1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	30*	30*	6	15	5	30*	8	5	17	21	14	6	14
3	30*	30*	8	13	5	30*	30*	5	8	8	16	15	12
4	30*	30*	7	6	14	30*	9	14	12	14	6	8	10
5	30*	9	5	25	12	30*	12	5	30*	14	6	5	5
6	30*	9	7	24	9	30*	7	15	9	15	30*	12	6
7	30*	7	7	12	5	30*	7	14	8	9	23	6	12
8	30*	9	8	20	11	30*	17	5	14	13	6	14	15
9	30*	13	8	30*	8	30*	30*	8	9	9	10	8	5
10	30*	8	5		6	30*	27	5	21	28	6		10
11			13		16		30*	7	7	5	9		13
12			15		11		17	9	14	8	5		6
13					7		10	5		5	7		9
14					8		7	7			5		8
15					8		10	14			6		5
16							30*	5					16
17							30*	5					30*
18							18	14					30*
19								6					30*
20													13
21													30*
FIX GOP (15):	20	10	6	10	9	20	20	10	10	10	10	5	20
Difference	-10	0	+6	-1	+6	-10	-2	+9	+2	+3	+5	+4	+1

*key frames inserted

We analyze the GOP size for all video sequences using our proposed *VBaR* framework. We compared the GOP size and its frequency for all video sequences, and experimental results are shown in Table 5. To begin with, there are more key frames selected for *Video 3*, *Video 5*, *Video 8* and *Video 11* (GOP size small) compared to *Video 1* and *Video 6* – due to the number of scene change detected. More significantly, we manage to show the robustness of our proposed framework for all types of video: sequences that have few scene change (*Video 1*) or sequences that have frequent scene change (e.g. *Video 13*) – as shown in Table 4. Our proposed method, i.e. variable GOP, compared to fixed GOP leads to significant reduction in the overhead involved in the transmission of intra-coded frames – shown in Table 4 (*Video 1* and *Video 6*). The GOP size, for all thirteen video sequences, range from 5 to 18, with few intra-coded frames of higher than 18 – see Figure 12. Those GOPs that is higher than 20 are mainly for *Video 1*, *Video 2* and *Video 6* - having fewer scenes change. Experimental results show that our proposed approach (variable GOP) is very useful to save bit rate for video with fewer scene change (e.g. romance or documentary film) and to compensate frequent scene change video (e.g. action film) with lower GOP.

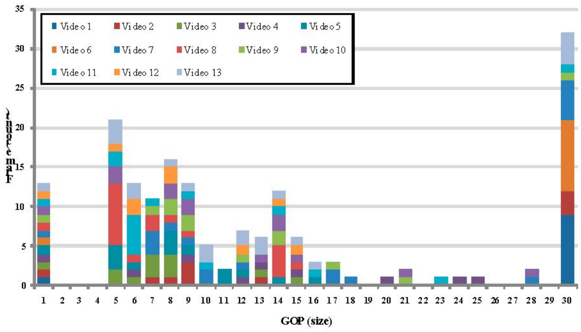


Figure 12: GOP statistics for thirteen different video sequences.

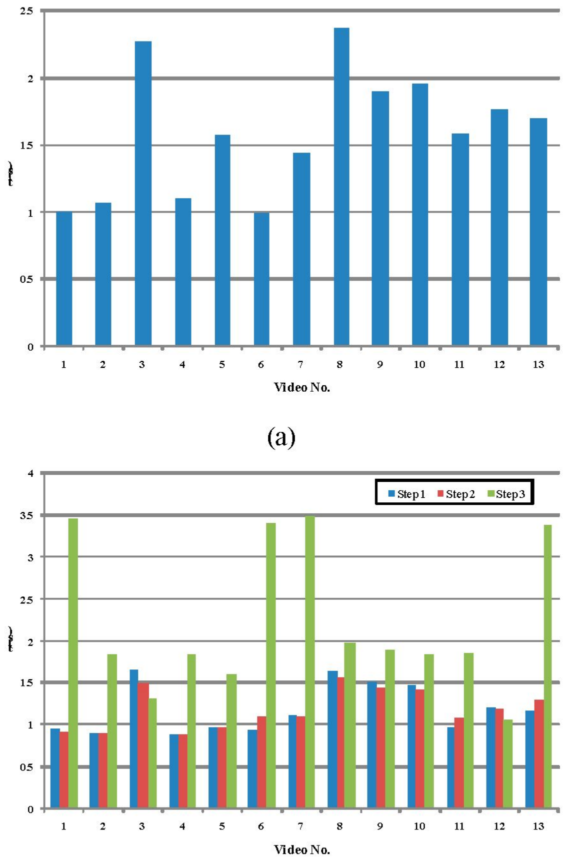
4.3 System Performances

The performance of the proposed framework is evaluated for all thirteen selected video sequences. Comparison of computational speed on the PC platform helps to identify the potential implementation of the method proposed. The algorithms are tested on an Intel E7300 Core 2 Duo 2.66GHz with 2GB of RAM – PC platform, using MATLAB®. The computation time is calculated for every *Step* – shown in Table 5. The time durations are represented in second (s).

Table 5: Comparison of Test Speed (in seconds)

Process	*Step 1*	*Step 2*	*Step 3*	Total time	Average/frame
Video1	285.8	3.7	13.8	303.3	1.01
Video2	135.5	8.1	16.6	160.2	1.06
Video3	149.3	29.8	26.1	205.2	2.28
Video4	133.0	10.6	22.1	165.7	1.10
Video5	126.5	30.3	48.5	205.3	1.57
Video6	279.4	4.4	13.6	297.4	0.99
Video 7	333.6	24.1	76.9	434.6	1.44
Video 8	245.9	48.8	61.4	356.1	2.37
Video 9	228.9	24.6	32.2	285.7	1.90
Video 10	221.0	31.6	40.4	29	1.95
Video 11	146.7	33.4	57.6	237.7	1.58
Video 12	91.5	21.5	19.2	132.2	1.76
Video 13	351.2	44.1	115.4	510.7	1.70

Video 6, which requires an average of 0.93 seconds/frame in *Step 1* and 1.1 seconds/frame in *Step 2*, while 3.4 seconds/frame in *Step 3*, has one the fastest processing speed – shown in Figure 13 (b). Figure 13 (a) also shows that complex video such as *Video 8* requires higher computation time. Nevertheless, experimental result shows that the time required for selecting key frame depends on the content in each video sequence, highest being 2.37 seconds/frame and lowest being 0.99 seconds/frame, relatively short.



(a)

(b)

Figure 13: Average processing time (seconds) per frame using *VBAR* for each (a) Video (b) *Step*

4.4 Comparison with different sequences

Table 6: Comparison among different types of video sequences

Features	# frame(s)	# key frame(s)	Average GOP
1. Camera setting			
a. Camera static	750	39	19
b. Camera moving	1645	133	12
2. Activities			
a. Slow motion	1350	63	16
b. Fast motion	1045	109	9
3. Scene change			
a. Few scene changes	1500	94	16
b. Many scene changes	895	78	11

We compared different type of video sequences characteristics to ascertain the potential of our proposed approach. We classified all videos into different settings: (1) camera settings, (2) activities, and (3) scene changes. We analyzed the number of key frame selected for each category. Experimental results are presented in Table 6. Experimental results show that fast motion video sequences have lower average GOP size, about 9, or more key frames, about 109. Also, the number of key frames selected for videos with frequent scene change and moving camera are much lower. Overall, in terms of video representation efficiency, the experiment results demonstrate the performance of the proposed *VBAR* framework.

4.5 Comparison with other work

We compared four other key frames selection methods to ascertain the potential of our proposed approach – shown in Table 7.

Table 7: Comparison with other key frames extraction methods

Features	Andreas [25]	Liu [11]	Chau [26]	Ouyang[27]	*VBAR*
1. Key frame selection					
	Temporal(T)	Color(C)	Visual (V)	Motion	T / C/ V
2. Application					
	Skimming	Segmentation	Efficiency	Skimming	Skimming
3. Test Videos					
	ND**	9 video	3 videos	18 videos	13 videos
4. Total Frames					
	ND**	1397	ND**	3912	2395
5. Analysis***					
	FrB	FrB	BlB	FrB	FrB/BlB
6. Temporal Constraint					
	Yes	No	No	No	Yes
a. Min (Reported)					
	ND**	None	1	None	5
b. Max (Reported)					
	ND**	None	13	None	30

*Color/Visual/Temporal **ND -Not described ***FrB–Frame-based/BlB–Block-based

Our proposed approach not only outperforms these methods, but also has three other advantages. Firstly, it applies temporal constraints for key frame selection (*Step 3*). Our proposed temporal constraints prevent key frames from occurring too far apart in time or too close together in time. Although Andreas's work [25] uses temporal constraints to filter out unsuitable clusters, its method only uses temporal constraints to cluster the frames in a video and to select a representative frame for each cluster in order to prevent selecting key frames which appear too close together in time. Chau's work [26], on the other hand, gives a set of key frame based on an objective model of visual content flow, but it failed to consider the temporal constraints in its method. As such, there exist many video shots having a GOP size as small as 1 to a maximum size of 13. Secondly, Liu's work [11] which extracts key frames based on parametric Gaussian Mixture Model (GMM) that are associated with video objects but did not report the computation time required. In comparison, the *VBAR* has lower computational complexity, as reported in Table 5, and a simpler representation. It is easy to implement and comes with a user-friendly interface. The *VBAR* system is able to generate summary for quick browsing of video content, i.e. dynamically generate flexible and effective summary.

Though Ouyang's work [27] proposed a similar interactive model of key frame selection, the graphical user interface does not describe and display how the parameters are selected and managed for the key frame selection.

5 Conclusions

In this paper, we assume that the intra-coded frame assignment and the intra-quantization parameter is taken care by the decoder. We focus on selecting intra-coded frames that best represent and describe the entire video well. We studied viewer's browsing patterns and incorporate viewer's interest into the key frame selection. Our major contributions are threefold. First, our proposed method manage to reduce the number of intra-coded frames by optimally selecting key frames using the proposed tri-step approach, being a simpler and faster alternative. Second, we proposed to limit the GOP size to allow bandwidth saving and to avoid video degradation due to unavailability of key frames. Third, we implement our framework on *VBaR* on a software platform to enable users to analyze selected video using a user-friendly graphical user interface.

The proposed *VBaR* framework is evaluated based on four important aspects: 1) key frame analysis, 2) GOP analysis, 3) system performance, and 4) performance comparison. The framework has been tested under various scenarios: sports video sequences, news video sequence, and other motion-filled sequences. The results were nevertheless promising: a consistent ability to divide a video into different clips using representative key frames. We anticipate that by delivering interesting video clips to subscriber, i.e. select representative key frames for all targeted content, we could minimize interactivity and eventually, reduce performance bottlenecks, long delays and poor user experience for subscribers, especially when roughly 40% of sessions contain some interactivity.

Future plans include testing the framework by using videos taken from different formats and scale. For the specific case of the intra-frame selection, where quantization parameter (QP) selection and transcoding technique selection can further reduce channel bandwidth utilization within a guaranteed picture quality, there is still considerable amount of work ahead. Nonetheless, the usage of *VBAR* in the present conditions is possible, providing a means to divide an entire video into video data units using representative key frames, and deliver only "watch" video data units to the STB, thus, enabling tremendous savings in bandwidth.

Acknowledgement

This work was supported by research fund of Hanyang University (HYU-2006-I).

References

[1] T. Syeda-Mahmood and D. Ponceleon, "Learning video browsing behavior and its application in the generation of video previews", in *Proceedings of the Ninth ACM International Conference on Multimedia*, Ottawa, Canada, 30 September – 05 October, 2001, pp. 119-128.

[2] Y. Zhu and D. Zhou, "Video Browsing and Retrieval Based on Multimodal Integration", in *Proceedings of the 2003 IEEE/WIC*, 13 – 17 October, 2003, pp.650.

[3] H. J. Zhang, J. Wu, D. Zhong and S. W. Smoliar, "An integrated system for content-based video retrieval and browsing", *Pattern Recognition*, vol. 30, no. 4, pp. 643-658, 1997.

[4] T. Syeda-Mahmood, S. Srinivasan, A. Amir, D. Ponceleon, B. Blanchard, D. Petkovic , "CueVideo: a system for cross-modal search and browse of video databases," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, vol.2, no., pp.786-787 vol. 2, 2000

[5] L. He, E. Sanocki, A. Gupta, and J. Grudin, "Auto-summarization of audio-video presentations", in *Proceedings of the Seventh ACM international Conference on Multimedia (Part 1)*, Orlando, Florida, United States, 30 October – 05 November 1999, pp. 489-498.

[6] C. Chen and Z. Yang, "MPEG4 Compatible Video Browsing and Retrieval over Low Bitrate Channel", in *Proceedings of the Third IEEE Pacific Rim Conference on Multimedia: Advances in Multimedia information Processing*, 16 – 18 December, 2002, pp. 1221-1226.

[7] M. Mills, "A magnifier tool for video data", in *Proceedings of ACM Human Computer Interface*, pp. 93-98, May 1992.

[8] W. Wolf, "Key frame selection by motion analysis", in *Proceedings of the 21st International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 1228-1231, 1996.

[9] H. Zhang, Z. Liu, H. Zhao, G., Cheng "Recognizing Human Activities by Key Frame in Video Sequences", *Journal of Software*, vol. 5, no. 8, pp. 818-825, Aug 2010

[10] A. Divakaran, K. A. Peker, R. Radhakrishnan, Z. Xiong and R. Cabasson, "Video Summarization Using MPEG-7 Motion Activity and Audio Descriptors", *Video Mining*, Rosenfeld, A.; Doermann, D.; DeMenthon, D., October 2003 (Kluwer Academic Publishers)

[11] L. Liu and G. Fan; , "Combined key-frame extraction and object-based video segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.15, no.7, pp. 869- 884, July 2005.

[12] X. Song, G. Fan, , "Joint Key-Frame Extraction and Object-Based Video Segmentation", in *Proceedings of IEEE Workshop on Motion and Video Computing*, vol.2, pp.126-131, Jan. 2005.

[13] X. Song, G. Fan, "A New Video Analysis Approach for Coherent Key-frame Extraction and Object Segmentation", in *Proceedings of IEEE 7th Workshop on Multimedia Signal Processing*, pp. 1-4, Oct. 30 2005-Nov. 2 2005

[14] P. Aigrain, H. J. Zhang, D. Petkovic, "Content-

Based Representation and Retrieval of Visual Media: A State-of-the-Art Review", *MultToolApp*, no. 3, pp. 179-202, November 1996.

[15] J. Lee, I. Shin H. W. Park, "Adaptive intra-frame assignment and bit-rate estimation for variable GOP length in H.264", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 10, pp. 1271-1279, Oct 2006.

[16] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard", *IEEE on Transactions Circuits and Systems for Video Technology* , vol. 13, no. 7, 2003, pp. 560-576.

[17] R. Hammoud and R. Mohr, "A probabilistic framework of selecting effective key frames for video browsing and indexing," in *Proceedings of International Workshop on Real-Time Image Sequence Analysis*, pp. 79–88, 2000.

[18] A. Girgensohn and J. Boreczky, "Time-constrained keyframe selection technique," *Multimedia Tools Application*, vol. 11, pp. 347–358, 2000.

[19] X. C. He and N. H. C. Yung, "Corner detector based on global and local curvature properties", *Opt. Eng.*, vol. 47, no. 5, 2008, pp. 057 008–1–057 008–12,.

[20] C. Y. Wei, M. B. Evans, M. Eliot, J. Barrick, B. Maust, and J. H. Spyridakis, "Influencing web-browsing behavior with intriguing and informative hyperlink wording", *J. Inf. Sci.*, vol. 31, no. 5, 2005, pp. 433-445.

[21] T. Zhu, R. Greiner, G. Häubl, K. Jewell, and R. Price, "Using Learned Browsing Behavior Models to Recommend Relevant Web Pages", in *Proceeding of the 2005 International Joint Conference on Artificial Intelligence*, Edinburg, Scotland, 30 July–5 August, 2005, pp. 1589-1590.

[22] B. A. Huberman, P. L. T. Pirolli, J. E. Pitkow, and R. M. Lukose, "Strong Regularities in World Wide Web Surfing", *Science*, Apr 3, vol. 280, no. 5360, pp. 95- 97, 1998

[23] D. Tao, J. Cai, H. Yi, D. Rajan, L. T. Chia, and K. N. Ngan, "Dynamic Programming-Based Reverse Frame Selection for VBR Video Delivery Under Constrained Resources," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no.11, pp. 1362-1375, 2006.

[24] E. R. Iain, "Video Codec Design", *John Wiley & Sons*, April 2002.

[25] A. Girgensohn and J. Boreczky, "Time-Constrained Keyframe Selection Technique", *Multimedia Tools Application*, vol. 11, no. 3, 347-358, August 2000.

[26] W.-S. Chau, O. C. Au, T.-W. Chan, T.-S. Chong, "Optimal key frame selection using visual content metric," in *Proceedings of International Conference on Communications, Circuits and Systems*, vol. 1, pp. 551- 555, 27-30 May 2005.

[27] J. Ouyang, J. Li, and H. Tang, "Interactive key frame selection model", *Journal Visual, Communication and Image Representation*, vol. 17, no. 6, pp. 1145-1163, December 2006.