REAL-TIME KEYFRAME EXTRACTION TOWARDS VIDEO CONTENT IDENTIFICATION

Maria Chatzigiorgaki and Athanassios N. Skodras

School of Science and Technology, Hellenic Open University, 26222 Patras, Greece
mhatzi@ieee.org, skodras@ieee.org

ABSTRACT

Keyframe extraction constitutes a fundamental unit in many video retrieval-related applications. In the emerging research field of content-based video copy detection, efficient representation of video content at keyframe level is crucial, due to the fact that similarity search is mainly performed between content-representative frames. In this paper a sequential search algorithm that bypasses the process of temporal video segmentation is proposed for keyframe extraction in MPEG videos. We aim at providing an efficient, real-time and fully-automatic way of extracting keyframes in videos, where not only the laborious task of offline video database indexing is avoided, but also query video processing is performed in the same manner as the reference video database. Significant reduction in computational cost is achieved by exploiting DCT coefficients in feature extraction. The effectiveness of the proposed scheme is evaluated in terms of quality and speed on the manually annotated TRECVid 2007 test video dataset.

Index Terms— Keyframe extraction, DCT, MPEG, video copy detection

1. INTRODUCTION

Keyframe extraction is the basis for many video-retrieval related applications, being one of the well-investigated subjects that still gains research community's attention. In the emerging research field of *Video Copy Detection* (VCD), the selection of keyframes is an important issue. However, a lot of effort has been devoted to feature extraction and similarity matching algorithms, due to their major contribution to the performance of such an application. Most of the existing keyframe extraction methods are not suitable for video copy detection, as they don't meet specific requirements. As a result, simple approaches of low complexity are usually preferred [1].

Representation of video content plays an important role in VCD applications. Several such approaches have been reported up to now, i.e. frame-based, uniform sampling, shot-based and content-based keyframe representation [2]. In *frame-based representation*, all the frames of a video are considered, leading to a huge amount of information, hard to manipulate. This no information loss representation has been adapted in ViCopT application [3]. In *uniform (or temporal) sampling*, the total number of content-representative frames is constrained by a predefined sampling rate, where a frame is selected depending on it's temporal position [4]. One of the well-investigated research areas is the *shot-based representation*, where a video is segmented into shots, used to represent its content. However, the most popular and efficient is the *keyframe representation*, where a compact set of keyframes is selected, depending on the content variation along a video. A video is temporarily segmented into shots and usually the middle frame of each shot is selected as keyframe. Alternatively, the first and/or the last frame of each shot is selected [5].

Generally, keyframe extraction techniques can be roughly categorized into *Sequential* and *Cluster-based methods* [6]. In sequential methods, as the name implies, consecutive frames are compared in sequential manner and keyframes are detected depending on the similarity either with previous frames or the previously detected keyframe. In cluster-based methods, the frames are grouped into a finite set of clusters in the selected feature space [7]. The frames of each shot form a cluster and the frame closest to the cluster's center is usually selected as keyframe. In this way, a compact set of keyframes is produced, capturing adequately the content variation along a video. However, the computational complexity is a major issue.

Another popular category is the *Compressed-domain techniques*, where features are extracted directly from the MPEG stream, making them preferable due to their real-time performance. Representative examples are the perceived motion energy (PME) model exploiting motion vector information [8] and the Discrete Curve Evolution (DCE) algorithm [9], where macroblock type information is exploited.

The requirements of a VCD application regarding the extracted set of keyframes, is that they must capture adequately the visual content variation along the video, while preserving their temporal position and being distinctive. An additional requirement is the real-time implementation of such a module

This work was supported by the project PENED (Contract no. $03E\Delta 832$) co-funded by the European Union - European Social Fund (75%), the Greek Government - Ministry of Development - General Secretariat of Research and Technology (25%) and the Private Sector in the frames of the European Competitiveness Programme (Third Community Support Framework).

as VCD applications are demanding from the perspective of other more important modules, like the feature extraction and the matching algorithm and further aggravation of the computational load is undesirable. Finally, the number of keyframes is a contributing factor in the performance of a VCD application, as an increased number of keyframes will have impact on the running time.

The rest of this paper is organized as follows. The proposed keyframe extraction algorithm is presented in Section 2, while experimental results are provided in Section 3. Finally in Section 4 conclusions are drawn.

2. THE PROPOSED ALGORITHM

The proposed algorithm consists of two main parts, namely the feature extraction module to represent every decoded frame, and the actual keyframe extractor, where the feature vectors of consecutive I-frames are compared in sequential manner in order to detect the content representative frames of a video, labeled as keyframes.

The design principle of the proposed scheme is that consecutive I-frames are examined on the basis of change detection in their feature vector representation. As in any sequential-type keyframe extractor, the process of shot determination through shot boundary detection (SBD) is bypassed, as shot changes are detected in terms of change captured in their feature vector behavior. In this way, the usually multi-pass and time consuming SBD process is avoided.

The underlying idea of the proposed scheme is that keyframes are selected from "stable" areas, i.e. areas whose frames belong to the same motionless shot or sub-shot. Stable frames are mostly preferred as content-representatives. However, contrary to relevant methods, two keyframes located at the begging and at the end of a stable frame area are selected instead of one, i.e. the one with the smallest variation along a group of frames. At this point it should be noted that keyframes are not selected "at the boundaries" but close to them, to avoid selection of frames belonging to a gradual transition (e.g. dissolve). It is well known that inside a gradual transition the content variation is very small and consequently there is a strong possibility for these frames to be miss-classified as stable.

Another aspect of the proposed scheme is that it takes advantage of any kind of transition in keyframe extraction process, regardless of shot segmentation. This means that whenever a transition takes place, either a shot cut or a transition from stable to motion frame areas and vice versa, a keyframe is selected. Considering that shots are usually unstable due to camera operation or object motion, one keyframe in such cases it is impossible to capture this kind of content variation. The flowchart of the proposed keyframe extraction algorithm is illustrated in Fig.1.

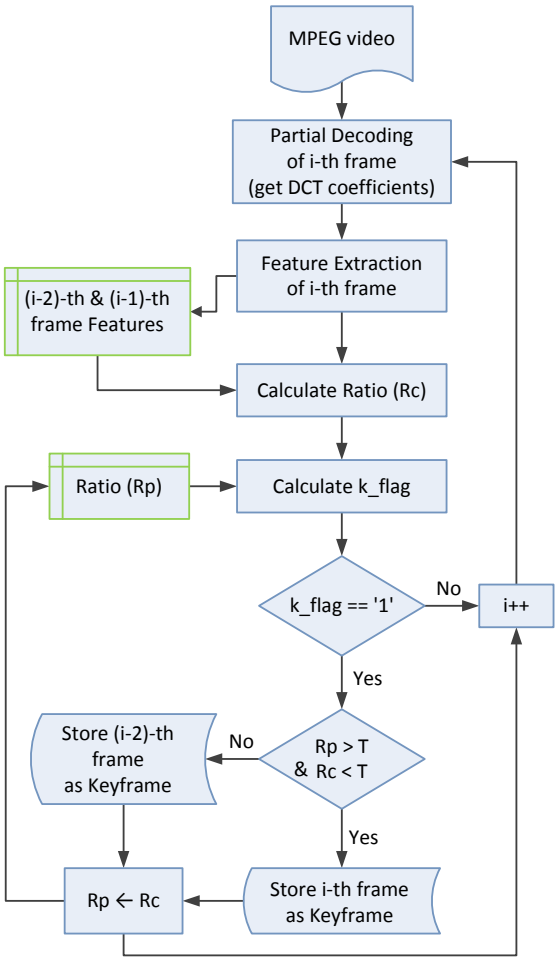


Fig. 1. Flowchart of the proposed keyframe extractor

2.1. DCT-domain feature extraction

As real-time implementation is one of the primary goals of the proposed keyframe extractor, the straightforward way to achieve this is by working directly in MPEG's compressed bitstream. To that purpose, every frame is partially decoded up to the stage of inverse quantization, resulting in its reconstructed DCT coefficients, which have been saturated to lie in the interval $[-1024, +1023]$. Further gain in speed can be accomplished considering only the Intra-coded (I) frame of each GoP. Based on these observations, to speed up the processing time the DC coefficients of every I-frame are used as the basis for feature representation. Two different features have been selected and evaluated in the proposed keyframe extraction algorithm.

Several compressed-domain features have been proposed, mainly for JPEG image retrieval, which can also be applied to MPEG videos. The DC difference-based feature proposed by Chang *et al.* [10] is such a feature. By taking into account only the DC coefficients of Y component in the feature extraction, the differences between every pair of consecutive

DC coefficients are calculated. In this way, a vector of K elements is obtained (where K denotes the total number of DC coefficients in Y component), each one corresponding to a DC difference value, except for the first element which is the first DC coefficient. For the construction of the feature vector, each of these K elements is examined and if it is greater than or equal to zero, the binary value '1'is assigned ('0'otherwise) to the difference being examined. This binary K-bits vector corresponds to the feature vector used to represent each I frame. This feature is characterized by compactness and real-time extraction, which makes it an ideal solution for the proposed keyframe extractor.

The similarity between two binary feature vectors f_1 and f_2 can be measured by applying a binary operation. Following the methodology described in [10], the binary xor operation is applied according to the following equation:

$$d(f_1, f_2) = \sum_{k=1}^{K} b_k^{f_1} \oplus b_k^{f_2} \tag{1}$$

where the $b_k^{f_1}$ and $b_k^{f_2}$ correspond to k-th bit of feature vectors f_1 and f_2 respectively.

Additionally, the YCbCr color layout (λ_Y) feature, recently proposed by Kim *et al.* [11] and used for frame-to-frame similarity in Video Linkage framework towards video copy detection, was selected for the proposed algorithm. In our proposed scheme, this feature was slightly modified in order to be extracted directly in the DCT-domain.

According to the originally proposed method described in detail in [11], each frame's Y, Cb and Cr components is divided into 16x16 blocks of pixels. Next, during a color averaging process, each block is represented by one color value, calculated as follows:

$$YCbCr_{avg} = \frac{2}{3} \cdot Y + \frac{1}{6} \cdot Cb + \frac{1}{6} \cdot Cr \tag{2}$$

This results in a feature vector of length M, where M corresponds to the total number of 16x16 blocks of each component.

In order to make this feature applicable to the proposed keyframe extraction algorithm, the DC coefficients of Y, Cb and Cr at macroblock level are taken into consideration. In 4:2:0 chrominance format, each macroblock consists of six 8x8 blocks, i.e. 4 Y, 1 Cb and 1 Cr. Each pair of Cb and Cr is used four times in combination to the four corresponding Y values in Eq.2, resulting in four $YCbCr_{avg}$ values per macroblock. Thus, the extracted feature vector consists of K values $YCbCr_{avg}$, where K denotes the total number of 8x8 blocks in Y component. At this point it should be noted that initially the DC coefficients are normalized to lie in the interval $[0, 255]$.

As a measure of similarity between two frames represented by feature vectors f_1 and f_2, the *cosine distance* is used, defined by the following equation:

$$sim_{\lambda_Y}(f_1, f_2) = \frac{\sum_{k=1}^{K} v_{1k} \cdot v_{2k}}{\sqrt{\sum_{k=1}^{K} v_{1k}^2 \cdot \sum_{k=1}^{K} v_{2k}^2}} \tag{3}$$

where $\{v_{1k}\}$ and $\{v_{2k}\}$ denote the $YCbCr_{avg}$ values of f_1 and f_2 respectively.

2.2. Keyframe extraction

Given an MPEG video sequence S_i $(i = 0, 1, \ldots N-1)$ of N frames as input, in the first step every decoded frame should be represented in feature-space. Let f_i be the feature vector representing the i-th frame, where i denotes the index of the currently processed frame.

At every instance where a new frame is decoded, a feature vector f_i is calculated to represent its visual content and stored for latter use where a new frame will be decoded. Every three consecutive frames, i.e. the current frame and the two previously decoded frames form a triplet, whose feature vectors are used for the decision making in the keyframe selection process. Based on this triplet of features (f_{i-2}, f_{i-1}, f_i), a quantity called *change Ratio* R_x, $x \in \{c, p\}$, as a measure of the content variation is calculated, according to the following equation:

$$R_x = \frac{\max\{d(f_{i-2}, f_{i-1}), d(f_{i-2}, f_i)\}}{\min\{d(f_{i-2}, f_{i-1}), d(f_{i-2}, f_i)\}} \tag{4}$$

In the above equation, d corresponds to distance (or similarity) between a pair of feature vectors. Depending on the feature, either Eq.1 or Eq.3 is used for the comparison of two feature vectors. As for the index x, c denotes the current ratio R_c, while with p is indicated the previously calculated and stored value of the ratio, R_p.

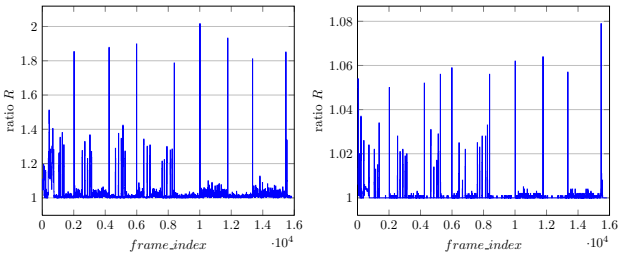
According to the flowchart of Fig.1, the keyframe decision depends on the value of k_flag, which is derived from the following logical function:

$$k\_flag = (R_c)_{bin} \oplus (R_p)_{bin} \tag{5}$$

where $(R_c)_{bin}$ and $(R_p)_{bin}$ are binary variables corresponding to R_c and R_p respectively. According to Eq.6, the logical '1' is assigned to the above variables if the ratio exceeds a predefined *Threshold* T.

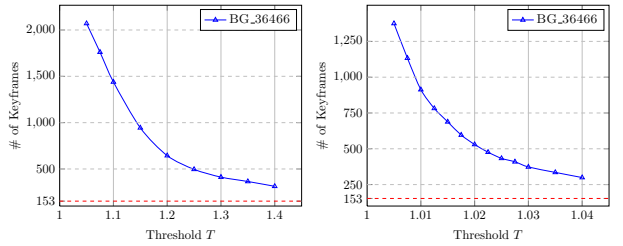
$$(R_x)_{bin} = \begin{cases} 1, & \text{if } R_x > T \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

As a result of the logical xor operation, the k_flag is reset (assigned to '0') if both ratios R_c and R_p follow the same behavior, i.e. either both have values greater, or less than T. Therefore, at least one keyframe is detected only if k_flag is set, corresponding to the occurrence of a transition. At the end of this procedure, the R_p is replaced by R_c and stored, until the next frame is decoded. Also the feature vectors f_{i-2} and f_{i-1} are updated ($f_{i-2} \leftarrow f_{i-1}$ and $f_{i-1} \leftarrow f_i$) and stored.



(a) Chang *et al.* feature [10] (b) Kim *et al.* feature [11]

Fig. 2. Ratio R variation along the video BG_37770



(a) Chang *et al.* feature [10] (b) Kim *et al.* feature [11]

Fig. 3. Number of keyframes as a function of the Threshold T using the test video BG_36446

2.3. Threshold selection

The threshold T plays an important role in the performance of the proposed keyframe extractor, as it constitutes a decisive factor to the determination of the number of keyframes that will be extracted. The optimal way to determine threshold's value in general is by using statistical methods. However, in this paper a simple methodology for threshold selection is followed. Two videos from TRECVid 2007 test dataset [12], namely BG_37770 and BG_36466, are used for the determination of the two thresholds.

Initially a range of threshold values must be defined for each algorithm, as ratio R values differ for each feature. In Fig.2 the variation of R along the test video BG_37770 is illustrated. Large peaks correspond to shot cuts, while smaller peaks correspond either to gradual transition, or camera/object motion. Based on the variation of R shown in Fig.2, the threshold can not exceed the values of 1.4 and 1.04 for the two features respectively.

The optimal threshold for each algorithm will be defined by examining the variation of the number of extracted keyframes depending on the threshold. The dependency of the number of keyframes on the threshold T is illustrated in Fig.3. Due to limited space, only the graphs corresponding to video BG_36466 are presented.

As it was expected, small value of T results in a large set of keyframes, most of them being very similar to each other. On the other hand a large value of T leads to a compact set of keyframes, but with strong possibility of miss-detections due to the fact that only changes between shots or sub-shots with dissimilar content (strong variation) will be considered as "transition", in order keyframe to be extracted. Therefore, there is a trade-off between a compact set of keyframes and adequate video content representation. Based on these observations and taking into consideration the fact that VCD applications require efficient representation of video content, where false negatives (i.e. shots not represented by at least one keyframe - miss detections) should be eliminated, the values of 1.2 and 1.015 were selected by visual inspection as optimal thresholds for the two algorithms. From now on, we will refer to *C-Algorithm* and *K-Algorithm* as the algorithms using Chang *et al.* [10] and Kim *et al.* [11] features respectively.

3. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed keyframe extraction algorithm, eight MPEG-1 video sequences from TRECVid 2007 test video collection [12] where selected, namely BG_2402 (00:26:39)[1], BG_26788 (00:14:21), BG_35046 (00:26:01), BG_35841 (01:01:43), BG_36878 (00:25:06), BG_37322 (00:26:11), BG_37347 (00:25:12) and BG_37940 (01:31:11). The algorithm was implemented in C and experiments were conducted on a Windows XP system with a 3GHz P4 processor and 1GB RAM.

Before we proceed with the evaluation of the proposed algorithm, a typical example of how the algorithm works in a real-world application scenario is presented. For that purpose, the TV spot "*BMW 1-series commercial with Kermit*" was used as query and over 50 versions of it (copies and near-duplicates) available on the web were downloaded and converted to MPEG-1 format. Seven of them were selected (in particular those with the largest variations) for evaluation. Two kinds of variations regarding the format and the content of a video can be defined. Thus, videos of varying encoding formats (flv, H.264, wmv, etc), frame rates (29.97/25/15 fps), aspect ratios (16:9, 4:3, etc) and spatial resolutions (320x240, CIF, QCIF) in combination with content modifications like changes in contrast, brightness and gamma, strong re-encoding and overlay text, form a dataset of video copies. Regardless of keyframe's feature-based representation to identify video copies, visually similar videos should result in similar sets of keyframes in order to effectively identify video copies. Fig.4 depicts the last eleven keyframes extracted by the proposed method using C-Algorithm for seven video copies. Mainly due to frame rate variations, the temporal locations of I frames differ slightly. However the proposed method appears to be effective, resulting in sets of similar keyframes.

The evaluation of the proposed algorithm is carried out in terms of the number and the percentage of the extracted keyframes and the processing time. The comparative re-

[1]Time durations are expressed in the format `hh:mm:ss`

Fig. 4. Extracted keyframes of seven different versions of the video "*BMW 1-series commercial with Kermit*" using the C-Algorithm ($T = 1.2$)

sults are demonstrated in Fig.5. In comparison to *Ground Truth*[2] (GT), both algorithms result in a reasonable *number of keyframes*, with the exception of the video BG_37940, which is justified by the complexity of its content. At this point it should be noted that the annotation of the GT set was constrained to one keyframe-per-shot description. In particular, the middle *I*-frame of each shot was selected as keyframe. Therefore, shots involving motion are represented by a single keyframe, while adequate content representation requires more than one. However, the advantage of manual annotation over an automatic one, is that it ensures that every shot will be represented by at least one keyframe, while in automated methods the uncertainty of missing shots is a major concern. Another important factor for the evaluation is the *keyframe percentage*. As it is shown in Fig.5(b), it is limited to a maximum of 2.5%, while it reaches 1.5% on average. Finally, the *processing time*, which reflects the ability of an algorithm to run in real-time, constitutes one of the aspects of more importance in the proposed algorithm, along with adequate video content representation. As it was expected, C-Algorithm runs slightly faster than K-Algorithm, mainly due to the similarity measure used. It should be noted that the processing time corresponds to the total running time, including the time of partial decoding of MPEG stream. It is worth to mention that a video of duration 1(1/2)hours (BG_37940) can be processed on average in 90 sec (see Fig.5(c)). In the case of 25fps, this corresponds to 60x real-time processing! For qualitative evaluation, a segment of the video BG_37940 was selected (see Fig.6). For this particular segment, the actual number of keyframes extracted by C-Algorithm is 74, while K-Algorithm results in 69 keyframes. Due to space restrictions, 18 selected keyframes are presented for each algorithm. With green frame have been marked the keyframes that are not included in the GT set, while red frame is used to indicate false alarms (frames with strong motion). The former correspond mainly to sub-shot transitions, which are not captured by one keyframe per shot description as in GT. Also we have noticed that shots of very short duration are excluded from GT annotation. Such an example is the 17th keyframe, extracted by both of our algorithms.

The major drawback of the proposed algorithm is the fact that it is threshold-dependent. Upon the requirement of VCD applications for adequate video content representation, thresholds should be tuned in order to avoid miss-detections, leading in this way to a set of keyframes characterized by redundancy (groups of visually similar keyframes). Among the weaknesses is also the lack of control on the number of the extracted keyframes, like in rate-constrained methods [6].

4. CONCLUSIONS

In this paper, we have presented a simple yet effective sequential keyframe extraction algorithm, orientated towards VCD applications. Keyframes are detected on the basis of transition detection, not only when shot cut occurs, but also the proposed extractor is able to detect transitions between sub-shots that usually occur due to camera/object motion. Although the number of the extracted keyframes is determined

[2]Annotation by Multimedia Computing Group, Institute of Computing Technology, Chinese Academy of Sciences (MCG-ICT-CAS)

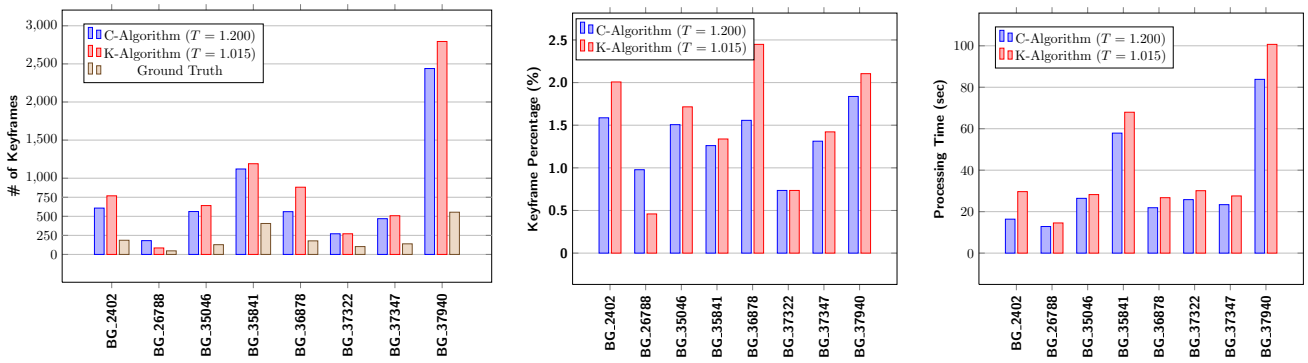
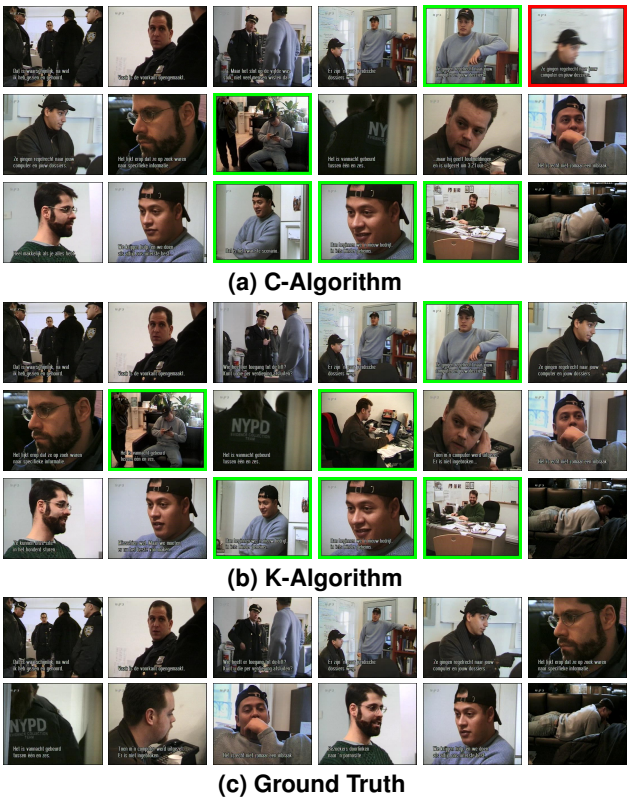


Fig. 5. Experimental results: a) # of extracted keyframes, b) Keyframe percentage (%) and c) Processing time (sec)

a posteriori, the proposed algorithm results in a reasonable set of keyframes (1.5% on average), while exhibiting robustness against numerous video content and format variations. In this way, a compact set of keyframes is extracted in real-time, capturing adequately the content variation along a video. The proposed algorithm is characterized by simplicity and flexibility. Therefore, it can easily be embedded into a VCD application, without impact on the computational cost. Further improvement of the proposed method includes a statistical-based approach for reliable determination of the threshold values, tested for numerous video format and content variations, as in a real-world VCD application scenario.



(a) C-Algorithm

(b) K-Algorithm

(c) Ground Truth

Fig. 6. Selected keyframes corresponding to a segment of the video BG_37940

5. REFERENCES

[1] N. Gengembre and S.-A. Berrani, "A probabilistic framework for fusing frame-based searches within a video copy detection system," in *Proc. CIVR'08*, Niagara Falls, Canada, July 7–9, 2008.

[2] X. Yang, P. Xue, and Q. Tian, "Automatically discovering unknown short video repeats," in *Proc. ICASSP'07*, Hawai, USA, 2007.

[3] J. Law-To, O. Buisson, V. Gouet-Brunet, and N. Boujemaa, "Robust voting algorithm based on labels of behavior for video copy detection," in *Proc. ACM Multimedia'06*, Santa Barbara, California, USA, Oct. 23–27, 2006.

[4] X.-S. Hua, X. Chen, and H.-J. Zhang, "Robust video signature based on ordinal measure," in *Proc. ICIP'04*, Singapore, Oct. 24–27, 2004.

[5] T. Can and P. Duygulu, "Searching for repeated video sequences," in *Proc. ACM MIR'07*, Augsburg, Germany, Sept. 28–29, 2007.

[6] H.C. Lee and S.D. Kim, "Rate-constrained key frame selection using iteration," in *Proc. ICIP'02*, Rochester, NY, USA, Sept. 22–25, 2002.

[7] X. Zeng, W. Hu, W. Li, X. Zhang, and B. Xu, "Key-frame extraction using dominant-set clustering," in *Proc. IEEE Int'l Conf. on Multimedia & Expo (ICME'08)*, Hannover, Germany, June 23–26, 2008.

[8] T. Liu, H.-J. Zhang, and F. Qi, "A novel video key-frame-extraction algorithm based on perceived motion energy model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 10, pp. 1006–1013, Oct. 2003.

[9] J. Calic and E. Izquierdo, "A multiresolution technique for video indexing and retrieval," in *Proc. IEEE Int'l Conf. on Image Processing (ICIP'02)*, Rochester, New York, USA, Sept. 22–25, 2002.

[10] C.-C. Chang, J.-C. Chuang, and Y.-S. Hu, "Retrieving digital images from a JPEG compressed image database," *Image and Vision Computing*, vol. 22, pp. 471–484, 2004.

[11] H. Kim, J. Lee, H. Liu, and D. Lee, "Video Linkage: Group based copied video detection," in *Proc. CIVR'08*, Niagara Falls, Canada, July 7–9, 2008.

[12] A.F. Smeaton, P. Over, and W. Kraaij, "Evaluation campaigns and TRECVid," in *Proc. of the 8th ACM International Workshop on Multimedia Information Retrieval (MIR'06)*, Santa Barbara, California, USA, 2006, pp. 321–330.