Efficient Key-frame Extraction Based on Unimodality of Frame Sequences

Vasileios T. Chasanis, Antonis I. Ioannidis and Aristidis C. Likas
Department of Computer Science and Engineering
University of Ioannina
Ioannina, Greece
{vchasani, aioannid, arly}@cs.uoi.gr

***Abstract*—Keyframe extraction for shot representation is the most common video summarization approach. Any reliable keyframe extraction algorithm should automatically detect the number of keyframes, while extracting non-repetitive keyframes that can efficiently summarize the video content. Moreover, it is important that key-frame extraction is performed in reasonable time. The proposed method is based on a moving window of successive frames that slides over the whole frame sequence (shot). The set of frames included in each window is tested for content homogeneity using an appropriate unimodality test. Thus, each window is characterized as unimodal or not and the frame sequence of each non-unimodal window is splitted into two (possibly unimodal) segments. In this way, each video shot is segmented into unimodal segments and the key-frames are computed as the representative frames (medoids) of each unimodal segment. An important aspect of the above method is that it does not require the number of keyframes to be specified in advance, since the number of segments is computed automatically. Numerical experiments demonstrate that our method provides reasonable estimates of the number of ground-truth keyframes, while extracting non-repetitive keyframes that efficiently summarize the content of each shot.**

I. Introduction

The most popular indexing and summarization method for video sequences is based on key-frame extraction. More specifically, each video is first segmented into shots, which are subsequently sufficiently using their most representative frames, which are called key-frames. Any key-frame extraction algorithm should fulfil some requirements. First, the key-frames should adequately represent the whole video content without missing important information and second, these key-frames should be non-repetitive, in terms of video content information. Finally, it is highly desirable that the number of keyframes should be specified automatically without any prior knowledge about the video content.

There are two major categories of methods for the keyframe extraction problem. The first one considers keyframe extraction as a clustering problem, thus frames are clustered into groups and the cluster representatives (e.g. medoids) are selected as keyframes. For example, in [1] the keyframes are extracted using clustering based on the visual variations in shots. A variant of this algorithm is presented in [2], where a threshold parameter defining whether two frames are similar controls the final number of key-frames. In [3], consecutive frames are clustered into groups with a split-merge approach based on the mutual information. A different technique for key-frame selection is described in [4], where the key-frame positions in the video sequence is taken into account.

The second major category of key-frame extraction methods is based on the detection of abrupt changes in the similarity between successive frames [5], [6]. In [7], three principles (Iso-Content Distance, Iso-Content Error and Iso-Content Distortion) are employed. Selected keyframes are equidistant in the video content curve with respect to these three principles. In [8], a keyframe selection framework based on keypoints is presented. A keypool of unique keypoints extracted from all frames based on SIFT descriptors [9] is generated and those frames that best cover the keypool are selected as keyframes. However, this method is relatively slow due to the cost for SIFT extraction and matching.

In the herein approach, we propose a novel keyframe extraction algorithm that belongs to the second category. The method is based on a unimodality test on a set of frames. This test essentially decides on the content homogeneity of frames in this set. We check for unimodality using the recently proposed *dip-dist criterion* [10], which is based on Hartigans' dip-test [11] for unimodality.

In our method the test is applied on a moving window of successive frames that slides over the whole shot. If at any point the test on the window declares multimodality, we assume that video content inside this window changes, thus we try to find the best split of the frames in this window into two (possibly unimodal) segments. The aforementioned procedure finally divides any shot into segments that are unimodal with respect to video content, and provides unique non-repetitive keyframes by selecting one representative frame (medoid) for each segment. A key aspect of our method, contrary to existing methods, is that it can provide automatically the number of keyframes of a video shot. Moreover, taking into consideration consecutive windows that declare multimodality, it is possible to identify multimodal frame sequences, that usually correspond to the transition period between two actions in a shot. It must be noted that we do not select keyframes from multimodal segments. Another important characteristic of our method is that it is very fast, thus it can be used for online video summarization.

The rest of the paper is organized as follows. In Sections 2 and 3 we describe the dip-dist criterion for deciding unimodality and our key-frame extraction method, respectively. In Section 4 we describe the evaluation procedure and provide numerical experiments. Finally, in Section 5 we provide

978-1-4799-2186-7/14/$31.00 ©2014 IEEE

conclusions and suggestions for further study.

II. The Dip-dist criterion for unimodality

The *Dip-dist* criterion has been proposed in [10] to evaluate the cluster structure of a set of data objects. The basic intuition behind dip-dist is that a set of objects is homogeneous if the underlying density distribution is unimodal. However, unimodality is not checked in the original data space, but it is tested using only the pairwise distance values between data objects (i.e. the distance matrix).

More specifically, in the dip-dist computation, each object of the dataset is considered as *viewer* that decides on the unimodality of the set in the following way: we consider the set of the pairwise distances from the viewer to all other data objects. Then, the density of this set of distances is tested for unimodality using Hartigans' dip test [11] and is characterized as either unimodal or multimodal. If the percentage of viewers suggesting multimodality exceeds a given threshold (e.g. 1%), then the set of objects is characterized as multimodal.

As proposed in [11], given a set of n values F_n, the dip-test computes the *dip value* of F_n ($dip(F_n)$) which is the departure from unimodality of the empirical distribution (cdf) of F_n:

$$dip(F_n) = \min_{G \in \mathbb{U}} \rho(F, G), \tag{1}$$

where $\rho(F, G)$ is the distance between the two distributions F, G and $\mathbb{U}$ the class of all unimodal distributions. In [11], it is also argued that uniform distribution U is the most appropriate for the null hypothesis. Thus the computation of the p-value for a unimodality test for a set F_n uses b bootstrap sets of n samples U_n^r, $r = 1, \ldots, b$ from U[0,1] and expresses the probability of $dip(F_n)$ being larger that $dip(U_n^r)$:

$$P = \#[dip(F_n) \leq dip(U_n^r)]/b, r = 1, \ldots, b. \tag{2}$$

The null hypothesis H_0 that F_n is unimodal, is accepted at significance level α if p-value$> \alpha$, otherwise hypothesis H_1 indicating multimodality, is accepted.

In our key-frame extraction method, we work with a moving window containing w successive video frames $V = \{f_1, \ldots, f_w\}$. Let Vf_i is the feature vector (image descriptor) of frame f_i and *Dist* the $w \times w$ matrix with the pairwise Euclidean distances:

$$Dist(f_i, f_j) = \sqrt{(Vf_i - Vf_j)^2}. \tag{3}$$

We use the dip-dist criterion to decide whether a window (set V) is unimodal or not, ie. if its visual content, as specified by the selected image descriptor, is homogeneous or not. Multimodality indicates that the video content changes in this set, thus the window should be splitted. The viewers suggesting multimodality are called split-viewers [10].

Summarizing, given a window of of w successive video frames and the corresponding pairwise distance matrix *Dist*, the dip-dist criterion is applied as follows:

1) Create b sets U_w^r of w values sampled form $U(0, 1)$ and compute the dip values $dip(U_w^r), r = 1, \ldots, b$, for those sets. Once the window w is fixed this can be done in a preprocessing step and the values can be stored for later use.
2) Compute the dip values $dip(i)$ for every frame/viewer f_i, $i = 1, \ldots, w$ using the matrix *Dist*.
3) Estimate the p-values $P(i), i = 1, \ldots, w$, based on Eq.2 using a significance level α and the percentage of frames/viewers identifying multimodality. If the percentage is higher than a threshold (in our case 1%) then the window is characterized as multimodal, otherwise it is considered as unimodal. Note that since in our method the maximum considered window size is 50, only one viewer that observes multimodality is required to characterize a window as multimodal.

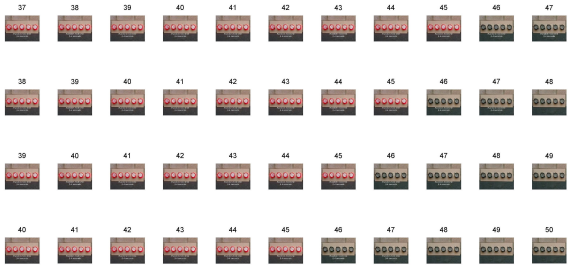

Fig. 1: Four consecutive windows of successive frames of an example video sequence.

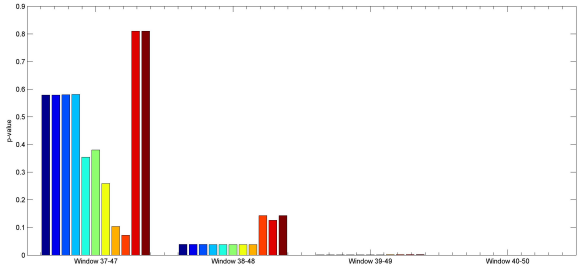

Fig. 2: Corresponding p-values of the frames of each window of Fig. 1.

In Fig. 1 we present four consecutive windows of successive frames of an example video sequence each containing 11 frames. In Fig. 2 we present the corresponding p-values of the frames/viewers of each window in Fig. 1. Note that in the first sequence only the last two frames differ in content, thus all viewers have high p-values (unimodality). In the next sequences the content variability inside the window increases and the corresponding p-values decrease.

It can be observed that in the first window both last "different" frames have high p-values, thus view the rest of the window as unimodal. However, as we progress in time and more different frames are added in the window, all frames start to act as split-viewers. Finally, in the fourth window, all frames/viewers propose multimodality. This example is the hard case, where visual content changes suddenly. In a similar way when visual content changes smoothly, we wish to find at least on split viewer to declare multimodality.

III. Keyframe Extraction

As mentioned in the previous Section, we can use the dip-test [10] to decide on whether a set of frames is unimodal (homogeneous content) or multimodal with respect to

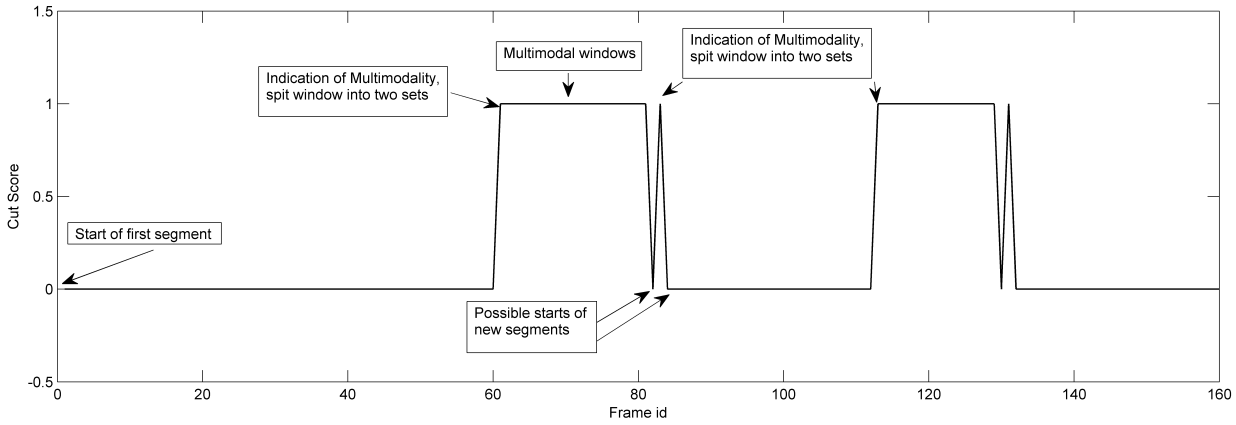

Fig. 3: Cut score sequence of our keyframe extraction algorithm for an example sequence of 160 frames.

a given image descriptor (eg. HSV histogram). Our method exploits this convenient and effective test in order to identify homogeneous frame subsequences (called unimodal segments). Note that such unimodal segments should be maximal in the sense that if we combine two adjacent unimodal segments, the resulting segment is no longer unimodal. This segmentation task is actually the major computational part in our method.

In order to extract the unimodal segments, we consider a moving window with w successive frames that slides over the frame sequence. At each position, the window of frames is tested for unimodality using dip-dist criterion and a binary *cut score* zero or one is assigned if unimodality is detected or not. More specifically, a window is characterized as multimodal if the percentage of frame/viewers that vote for multimodality (split viewers) is greater than a threshold t_v=0.01 as proposed in [10]. In Fig.3, we present the sequence of *Cut Score* values (as the window moves along the frame sequence) computed for a shot of 160 frames. The segmentation process works as follows: the first unimodal segment of the shot starts at the first frame, whose corresponding window is unimodal, eg. frame #1 in Fig.3. The algorithm then proceeds until the first multimodal window is encountered; such window corresponds to frame #61 in Fig.3. In this case, the set of frames in the multimodal window is splitted into two segments of successive frames as described later in this Section. Let s be the splitting position inside the multimodal window of frames. Then we can define the boundaries of first unimodal segment as $S_1 = \{1, s\}$. Note that the first frame of the next segment is not necessary the next frame $s+1$, since it must also fulfil the requirement that its corresponding window of frames is unimodal. If not, we seek for the first unimodal window which corresponds to frame #82 in Fig.3. This frame is considered as the begining of the new segment. The method proceeds analogously until all windows have been tested for unimodality. For the example in Fig.3, the extracted segments are $S_1 = \{1, 71\}$, $S_2 = \{82, 83\}$, $S_3 = \{84, 120\}$, $S_4 = \{130, 133\}$ and , $S_5 = \{132, 160\}$. To avoid very small segments like S_2 and S_4, we discard segments containing less than five frames. It must be noted that there may exist intervals such as 72-83 and 121-129 that have not been assigned to any segment. These frames usually correspond to transition intervals (e.g. moving from one room to another), thus we do not select keyframes from such multimodal segments.

In general, we consider that a unimodal segment starts at the first unimodal window (that follows a multimodal window) and lasts for several consecutive frames whose corresponding windows remain unimodal, until a multimodal window is encountered. This window is then splitted into two segments at a splitting point s, and the frame at position s is considered as the end of the unimodal segment. The next unimodal segment then starts at the first unimodal window that is encountered after the end of the previous segment. This segmentation procedure continues until the last window of the sequence has been examined.

After the unimodal segments have been extracted, the medoid frame of each segment is selected as key-frame. Thus, the number of key-frames is equal to the number of extracted segments. Note that the medoid is defined as the frame of a segment with minimum average dissimilarity to all other frames of its segment.

A. Splitting a multimodal window

Suppose that a multimodal window of frames (with size w) has been characterized as multimodal and we wish to optimal split this sequence into two segments. Let *Dist* the $w \times w$ matrix of pairwise distances of the frames of the window. To find the optimal split, we consider all splitting positions j ($j = 2, \ldots, w-1$) in this sequence. For each splitting position j, we obtain two segments $S_1(j)$ (containing frames $1, \ldots, j$) and $S_2(j)$ (containing frames $j+1, \ldots, w$). The quality of this split is computed by finding the medoids of $S_1(j)$ and $S_2(j)$, computing the sum of distances from the medoid for each segment (clustering error) and summing the two clustering errors to obtain the total clustering error $E(j)$. We consider that the best split at frame s for which the total clustering error $E(j)$ is minimum:

$$s = \min_j (\min_k \sum_{i=1}^{j} Dist(i,k) + \min_l \sum_{i=j+1}^{w} Dist(i,l)), \quad (4)$$

with $k = 1, \ldots, j$ and $l = j+1, \ldots, w$.

IV. Experiments

In the following Section, we present the dataset we have use to evaluate our algorithm, the evaluation process and the experimental results.

A. Dataset

We have used two video datasets for evaluation. The first consists of 13 videos, where the first ten videos are also used in [8] and have been taken from Open Video Project (http://www.open-video.org). Moreover, two additional videos and the widely known Foreman video sequence were used. Details and screenshots from all videos tested, are given in Table I and Fig.4, respectively. The second video dataset consists of eleven video sequences taken for TRECVID 2008 Test Data [12]. Details and screenshots from all videos tested, are given in Table II and Fig.5, respectively.

TABLE I: First Video Dataset.

Video Name	Start Frame	End Frame	# of Frames
v25 A New Horizon, segment 02	664	900	237
v28 A New Horizon, segment 05	3223	3440	218
v33 Take Pride in America, segment 03	540	650	11
v39 Senses And Sensitivity, Introduction to Lecture 4 presenter	1838	1934	97
v40 Exotic Terrane, segment 01	1790	1989	200
v49 America's New Frontier, segment 07	150	500	351
v57 Oceanfloor Legacy, segment 04	1600	1800	201
v58 Oceanfloor Legacy, segment 08	540	633	94
v63 Hurricane Force - A Coastal Perspective, segment 03	867	1012	146
v66 Drift Ice as a Geologic Agent, segment 05	766	977	212
Foreman sequence	1	400	400
video 1	1	271	271
video 2	1	175	175

TABLE II: Second Video Dataset (Trecvid).

Video Name	Start Frame	End Frame	# of Frames
MRS145918 v1	2160	2990	830
MRS145918 v2	9690	10770	1080
MRS145918 v3	12600	13600	1000
MRS157444 v1	726	1331	605
MRS157444 v2	9076	11294	2218
MRS157469 v1	2200	5000	2800
MRS158013 v1	3370	8258	4888
MS0237650 v1	768	953	285
MS0237650 v2	1050	2341	1991
MS0237650 v3	3900	4570	670
MRS148800 v1	7740	12150	4410

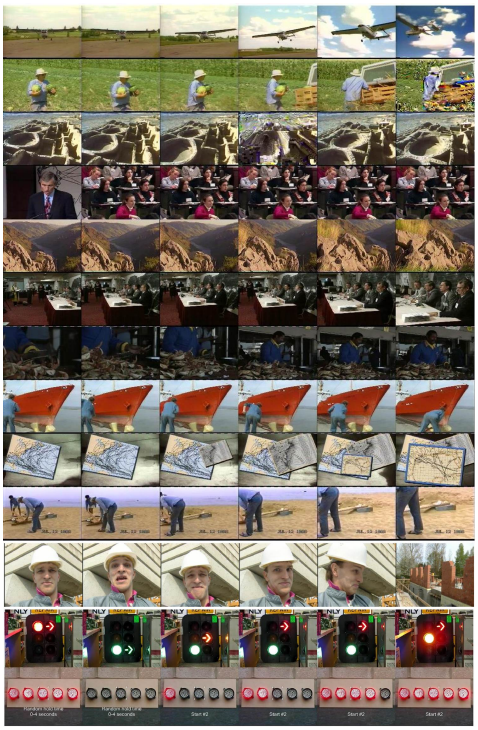

Fig. 4: Screenshots of the video dataset.

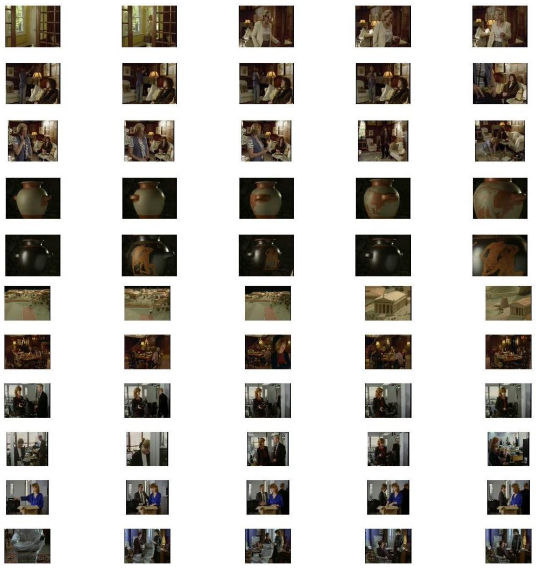

Fig. 5: Screenshots of the second video dataset.

B. Evaluation

The evaluation of keyframe extraction algorithms is non-trivial, due to subjectivity imposed when selecting the ground-truth (GT) keyframes and also due to the difficulty in comparing the extracted with the ground-truth keyframes. We evaluate our keyframe extraction algorithm in three ways. At first we test its ability to estimate the correct (as specified in the ground-truth) number of keyframes. For this reason we compute the average absolute difference between the number of extracted keyframes and the number of the keyframes in the ground-truth (D_k). Furthermore, in order to compare our solution with the ground truth solution, we do not directly compare the corresponding keyframes, but follow an alternative more robust approach that exploits the fact that our method also provides a segmentation of the video sequence into segments: we assign each ground truth keyframe to the extracted segment where it belongs to. Let N_S the number of extracted segments. We form a histogram vector with each element K_j, $j = 1, ..., N_S$ indicating the number of ground-truth keyframes assigned to segment j. Obviously, the

histogram vector $\hat{K}$ of a reference (perfect) solution would have $\hat{K}_j = 1$, for all j, implying an one-to-one correspondence between ground-truth keyframes and extracted segments. It is obvious that the closer the distribution K to the $\hat{K}$, the better the performance of the algorithm. Thus we define the quantity:

$$D_d = \frac{1}{N_S}||K - \hat{K}||_2, \quad (5)$$

as a second performance measure, where lower values indicate better performance.

To be more clear, we provide below some typical example cases. Suppose we are given a shot with four ground truth keyframes:

- Case 1: The method segments the shot into four segments, thus four keyframes are extracted. The reference histogram is $\hat{K} = [1111]$. If all ground truth keyframes are distributed uniformly in the four extracted segments then $K = [1111]$ and $D_d = 0$.
- Case 2: The method segments the shot into four segments and keyframes are not distributed uniformly, for example $K = [2011]$. In this case $D_d = 0.5$.
- Case 3: The method segments the shot into three segments, thus underestimates the number of ground truth keyframes. The reference histogram is $\hat{K} = [111]$ and suppose for example that $K = [211]$. In this case $D_d = 0.33$.
- Case 4: The method segments the shot into five segments, thus overestimates the number of ground truth keyframes. The reference histogram is $\hat{K} = [11111]$ and suppose for example that $K = [11101]$. Then $D_d = 0.2$.

The third way we used to assess the performance of our method was visual evaluation. Similarly to [8], the evaluation of the results were based on the visual comparison of the key-frames extracted from the experiments against the ones in the ground truth set. Three persons with video-processing background participated in the evaluation process and the cross-section of their evaluations was used. $F1$ measure has been used for evaluation provided from the following equation:

$$F1 = \frac{2 \times P \times R}{P + R}, \quad (6)$$

where P and R are Precision and Recall, respectively, and are computed form the following equations:

$$P = \frac{N_c}{N_c + N_m}, R = \frac{N_c}{N_c + N_{ms}}, \quad (7)$$

where N_c, N_m and N_{ms} are the number of correct, multiple and missed detections of ground-truth keyframes, respectively. Note that as multiple detection we consider the case where a ground-truth keyframe is found similar to more than one of the extracted keyframes.

For the second video dataset, TRECVID has provided the ground truth inclusion. In Table III we present the ground truth for video sequence "MRS145918 v1" as provided by TRECVID. In Fig. 6 we provide frames that describe the corresponding ground truth. We have asked three persons with video-processing background to evaluate the key-frames extracted from the experiments against the ground truth provided by TRECVID. Recall, Precision and F_1 measures are employed to evaluate the performance of the algorithms under comparison.

TABLE III: Ground truth for MRS145918 v1 video sequence as provided by TRECVID.

woman in white jacket enters house through white door
woman in white jacket walks into room past man and woman
woman in white jacket takes off her sunglasses
woman in white talks, looking at glass case
woman in white turns away from wall

Fig. 6: Frames corresponding to the MRS145918 v1 video sequence.

C. Experimental Results

HSV normalized color histograms were used to describe video frames, with 8 bins for hue and 4 bins for each of saturation and value, resulting in a 16 (8+4+4) dimensional feature vector in case of 1D histogram (named HSV1D) and in a 128 (8×4×4) dimensional feature vector in case of 3D histogram (named HSV3D). The parameters of the dip-dist criterion are set to α=0 for significance level of dip test and b=1000 for the number of bootstraps. It is worth mentioning that the dip values $dip(U_w^r), r = 1, \ldots, b$, for all Uniform sample distributions U_w^r are the same for each window, thus they can be precomputed once and stored for later use.

In Tables IV and V we present the performance of our keyframe extraction algorithm using HSV1D and HSV3D histograms, respectively. The values of the three performance measures defined in the previous section are reported for various window sizes.

It is clear that the proposed method yields very good performance for windows of size 30 to 50, with peak performance at w=45 for both histograms. When HSV3D histograms are employed, our algorithm estimates the number of keyframes very well (mean absolute difference is 0.81), while the visual evaluation of the extracted keyframes (F_1=89.03%) indicates that the extracted keyframes are non-redundant.

In Table VI, we provide comparative results of our algorithm with the method proposed in [8], where two parameters, *Coverage* and *Redundancy*, are used to guide the keyframe extraction process. It must be noted that the number of keyframes is not automatically estimated, but it is controlled by the *Coverage* parameter. We have tried different values of *Coverage* and in Table VI we present the best performance. The D_d value is not available for this approach, since no segments are extracted using this method.

In Table VII, we provide comparative results of our algorithm with the method proposed in [8] on the second video dataset taken from TRECVID. We use a window of size

TABLE IV: Performance results of our algorithm using HSV1D on the first video dataset.

	HSV1D		
Window	D_d	D_k	F_1
5	0,7021	2,94	48,69%
10	0,6006	2,31	57,30%
15	0,5211	2,13	66,06%
20	0,3821	1,25	71,88%
25	0,3779	1,25	69,30%
30	0,3863	1,38	76,63%
35	0,3683	1,38	80,03%
40	0,3544	1,31	75,11%
45	**0,2932**	1,31	**81,86%**
50	0,2901	**1,13**	77,81%

TABLE V: Performance results of our algorithm using HSV3D on the first video dataset.

	HSV3D		
Window	D_d	D_k	F_1
5	0,6341	2,88	53,41%
10	0,5243	2,06	67,90%
15	0,5391	2,19	61,08%
20	0,4279	1,50	70,14%
25	0,3934	1,50	66,70%
30	0,3085	1,13	75,91%
35	0,3136	1,38	79,72%
40	0,3165	1,13	81,55%
45	**0,2234**	**0,81**	**89,03%**
50	0,2416	1,13	86,59%

TABLE VI: Comparative results on the first video dataset.

	D_d	D_k	F_1
HSV1D w = 45	0,2932	1,31	81,86%
HSV3D w = 45	**0,2234**	**0,81**	**89,03%**
Method in [8]	-	1,92	64,78%

$w = 45$ for our method. It is clear that our algorithm surpasses the method under comparison, while providing a very good inclusion of ground truth.

It must also be noted that this method is very slow compared to our approach. For example, for a video shot with 300 frames, the computational time of then method in [8] is 150 sec, whereas in our approach it is less than 3 seconds. It is worth mentioning that our algorithm does not require the entire video shot to extract each keyframe, thus it can be used for online video summarization. Since unimodal segments are extracted progressively, keyframes are extracted immediately after their corresponding segment is detected.

TABLE VII: Comparative results on the second video dataset.

	R(in%)	*P(in%)*	F_1*(in%)*
MRS145918 v1	80.00	66.67	72.73
MRS145918 v2	100.00	80.00	88.89
MRS145918 v3	100.00	100.00	100.00
MRS157444 v1	66.67	100.00	80.00
MRS157444 v2	100.00	77.78	87.50
MRS157469 v1	100.00	100.00	100.00
MRS158013 v1	100.00	61.54	76.19
MS0237650 v1	100.00	66.67	80.00
MS0237650 v2	100.00	72.73	84.21
MS0237650 v3	100.00	75.00	85.71
MRS148800 v1	100.00	73.33	84.62
HSV3D w = 45 (Average)	**95.15**	**79.43**	**85.44**
Method in [8] (Average)	77.03	69.44	63,34

V. Conclusions

In this paper, a novel keyframe extraction algorithm has been proposed. A moving window of successive frames that slides over the whole frame sequence is tested for unimodality using dip-test criterion on the distribution of their pairwise distances. In case of non-unimodality, frames are splitted into two disjoints sets. Finally, each video shot is segmented into unimodal segments with respect to video content that provide the final non-repetitive keyframes. Performance results on several video sequences demonstrate that our method efficiently estimates the correct number of keyframes, while extracting non-repetitive keyframes that efficiently summarize the video content of each shot. In future work, we plan to employ additional descriptors to capture different aspects of video frames.

Acknowledgment

The work described in this paper is co-financed by the European Regional Development Fund (ERDF) (2007-2013) of the European Union and National Funds (Operational Programme Competitiveness and Entrepreneurship (OPCE II), ROP ATTICA), under the Action "SYNERGASIA (COOPERATION) 2009".

References

[1] Y. Zhuang, Y. Rui, T. Huang, and S. Mehrotra, "Adaptive key frame extraction using unsupervised clustering," in *IEEE International Conference on Image Processing*, vol. 1, oct 1998, pp. 866 –870.

[2] Z. Rasheed and M. Shah, "Detection and representation of scenes in videos." *IEEE Transactions on Multimedia*, vol. 7, no. 6, pp. 1097–1105, 2005.

[3] Z. Cernekova, I. Pitas, and C. Nikou, "Information theory-based shot cut/fade detection and video summarization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 1, pp. 82–91, 2006. [Online]. Available: http://dx.doi.org/10.1109/TCSVT.2005.856896

[4] A. Girgensohn and J. S. Boreczky, "Time-constrained keyframe selection technique." *Multimedia Tools Applications*, vol. 11, no. 3, pp. 347–358, 2000.

[5] W. Wolf, "Key frame selection by motion analysis," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, ser. ICASSP '96, 1996, pp. 1228–1231.

[6] G. Ciocca and R. Schettini, "An innovative algorithm for key frame extraction in video summarization." *Journal of Real-Time Image Processing*, vol. 1, no. 1, pp. 69–88, 2006.

[7] C. Panagiotakis, A. D. Doulamis, and G. Tziritas, "Equivalent key frames selection based on iso-content principles." *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 3, pp. 447–451, 2009.

[8] G. Guan, Z. Wang, S. Lu, J. D. Deng, and D. D. Feng, "Keypoint-based keyframe selection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 4, pp. 729–734, 2013.

[9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[10] A. Kalogeratos and A. Likas, "Dip-means: an incremental clustering method for estimating the number of clusters." in *Proceedings of the Neural Information Processing Systems*, ser. NIPS '12, 2012, pp. 2402–2410.

[11] J. A. Hartigan and P. M. Hartigan, "The dip test of unimodality," *The Annals of Statistics*, vol. 13, no. 1, pp. 70–84, 1985.

[12] P. Over and A. F. Smeaton, Eds., *Proceedings of the 2nd ACM Workshop on Video Summarization, TVS 2008, Vancouver, British Columbia, Canada, October 31, 2008*. ACM, 2008.