Video Summarization via Segments Summary Graphs

Mahmut Demir & H. Işıl Bozma
Intelligent Systems Laboratory, Electrical and Electronic Engineering
Boğaziçi University, Bebek 34342, İstanbul, Turkey
mahmut.demir@boun.edu.tr

Abstract

In this paper we propose a novel approach to video summarization that is based on the coherency analysis of segmented video frames as represented by region adjacency graphs. Similar segments across consecutive region adjacency graphs are matched and tracked using an efficient graph matching technique. Shot boundaries are detected based on a coherency score that measures the appearances and disappearances of tracked segments. As such, it is possible to form a compact representation of each detected shot based on prevalent segmented regions and their relations - referred to as the 'segments summary graphs'. Furthermore, the segments summary graph is amenable for further semantic analysis and understanding of the scene. Experiments on benchmark datasets demonstrate that our method outperforms the state of the art summarization approaches.

1. Introduction

Video summarization aims to generate a compact representation of a video sequence in order to have a more efficient storage and processing. The partitioning of the video is integral to its summarization. The most common approach to partitioning is shot boundary detection as it is intrinsically and inextricably linked to the way that video is produced [5]. Here, a shot is defined as the longest coherent sequence of frames between two cuts. Generally, shots are separated by one of the several motion picture effects such as cuts, fade, dissolve or camera motion such as rotating or zooming. Boundary detection can be performed by analyzing the dissimilarity of successive frames where high dissimilarity indicates the boundary. Several different methods have been proposed for comparing frames such as color histogram difference [11, 28], object tracking [26, 17], motion field [27, 3], event analysis [15] and graph similarity [14]. Although these approaches can easily detect abrupt shot changes such as hard cuts, other effects such as fading or dissolving are relatively hard to detect due to gradual shot changes that spread over the number of frames. Sliding window approaches and adaptive thresholding methods are used in various works to detect gradual shot changes at higher rates [21, 16, 2].

Once a shot is determined, the next step is to transform its data into a compact representation. This is a challenging problem as this representation needs to encode all noteworthy information in the shot. The two most common approaches are selecting a collection of static keyframes of video shots or composing shorter clips of shots. As such, some of the semantic content that are crucial may be missed or the representation may not be sufficiently compact. In mosaic based approaches [1, 23], panoramic images are created from several frames and dynamic scene contents are superimposed in a single panorama. However, this approach requires high computation power and only applicable in static background scenes. Furthermore, most approaches do not encode the semantic information however incorporating semantics would enhance the browsing experience and facilitate content based video retrieval and search. Encoding audio-visual cues [9, 8], using video annotations [25], object and event analysis [7, 24] are some of the several techniques used for including semantics. These methods, however, requires either manual annotation or computationally expensive content analysis.

In this paper, we propose a novel approach to shot boundary detection and graph based semantic representation of shots. Our approach is based on the coherency of segmented regions adjacency graphs extracted at each frame. The nodes (segments) of region adjacency graphs are connected temporally and tracked through the video sequence using a simple graph matching technique. The novelties of this approach are two-fold: First, shot boundaries correspond to low coherency regions that are determined based on the number of appearing/disappearing nodes inside a sliding window. As such, detecting gradual changes become possible – in contrast to previous graph-based approaches such [14] where boundaries are determined via comparing the similarity of consecutive frames. Second, it not only enables shot detection but also provides a compact representation - referred to as segments summary graph - for each shot. Each segments summary graph encodes the major entities and their spatio-temporal relations in that shot. As such, it differs from previous related work where such an encoding is not possible - as each shot is summarized based on only a single key graph. The outline of this paper is as follows: First, the overall approach is presented in Section 2. The construction of region adjacency graphs is explained in Section 3. This is followed by region adjacency graph matching method in Section 4. Shot detection based on region adjacency graphs is explained in Section 5. The formulation of segments summary graphs is introduced in Section 6. The proposed approach is evaluated experimentally in Open Video [6, 18] dataset and compared with those of state of the art approaches in Section 7. The paper concludes with a brief summary.

2. Overall Approach

Consider a sample video that is comprised of a sequence of frames f_k with $k \in \mathcal{K}$. The proposed approach consists of four steps as shown in Fig. 1. The first step is forming the region adjacency graph (RAG) of each image frame f_k. In the RAG, segmented regions in the image and their spatial relationships are expressed as nodes and edges respectively [22]. The next step is to match any the newly formed RAG with those that are associated with the previous frames as to identify nodes (segments) that have appeared previously and hence assign their labels accordingly. In the third step, coherency score is calculated based on number of appearing/disappearing nodes of RAGs through a sliding window. In the last step, frames associated with low coherency regions are assigned as shot boundaries.

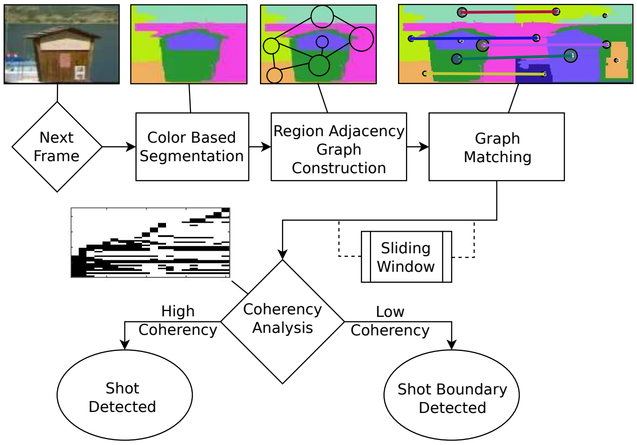


Figure 1: Shot boundary detection algorithm overview

Immediately after shot boundary is detected, a shot is defined by a set of frames between two shot boundaries. The frame associated with the highest coherency score is selected as the summary keyframe and segments summary graphs (SSG) is contructed by grouping only the long-life segments. The resulting SSG encodes segments and theirs spatial relations that are prevalent in the shot.

3. Region Adjacency Graphs

As the first step, each video frame f_k is represented as a RAG. This is done via segmenting the frame into N_k homogeneous color regions $\mathcal{S}^k = \{S_i^k\}_{i=1}^{N_k}$ using a segmentation method proposed in [10]. The segmented regions and their adjacency relationships are represented by the nodes and the edges of a region adjacency graph G^k. Thus, each RAG is an attributed graph that consists of $G^k = (\mathcal{N}^k, E^k, \mathcal{A}^k)$ where $\mathcal{N}^k$ is the set of nodes, E^k is the edge set and $\mathcal{A}^k$ is the attribute set that contains attributes related to vertices N_i^k and E_{ij}^k. Each segment $S_i^k \in \mathcal{S}^k$ is associated with a node N_i^k. If two segments S_i^k and S_j^k have common borders, edge relation E_{ij}^k between the respective nodes N_i^k and N_j^k is formed. A node N_i^k is associated with a set of attributes as given by a N_A-dimensional vector $a(N_i^k)$ as derived from the respective segment S_i^k such as its area, centroid and mean color. The edge attribute w_{ij}^k is set to a value that is inversely proportional to mean color difference between two segments S_i^k and S_j^k. The top three images in Fig. 1 illustrate RAG construction. For visualization purposes, the position, color and radius of nodes represent the center of mass, mean color and total area of segments respectively.

4. RAG Matching and Node Existence Matrix

In the second step, each newly formed RAG is matched with those that are associated with the previous frames to recognize nodes (segments) that have appeared previously and hence assign their node labels accordingly. In this way, the nodes of each RAG are related to nodes of previous RAGs and connected temporally. Then, the nodes of RAG at each frame is placed to the node existence matrix with their associated node label and frame number. Example node existence matrix is shown in Fig. 2.

The graph matching algorithm is an extended version of a method based on node signatures [13]. The node signature consists of node attributes, the number of incident edges $d(n_i)$ and the attributes of the edges of neighboring nodes $E(N_i^k)$:

$$s(N_i^k) = \{a(N_i^k), d(N_i^k), w_{ij}^k \text{ for } j \in E(N_i^k)\} \tag{1}$$

Given two RAGs G^k and G^l, $l > k$, cost matrix C with the corresponding elements c_{ij}^{kl} is defined based on node signatures as:

$$\begin{aligned} c_{ij}^{kl} &= \delta(s(\mathcal{N}_i^k), s(\mathcal{N}_j^l)) \\ &= \left\| s(\mathcal{N}_i^k) - s(\mathcal{N}_j^l) \right\| \end{aligned} \tag{2}$$

where δ defines a weighted Manhattan distance and c_{ij}^{kl} is the distance between two nodes $\mathcal{N}_i^k$ and $\mathcal{N}_j^l$. Calculated cost matrix, C is used as the basis for an attributed graph matching using Hungarian algorithm with $O(n^3)$ running time. The resulting permutation matrix P^{kl} defines the optimum matching between the nodes of two given graphs. However some of these matches may contain false or actually unrelated node-to-node assignments as node attributes associated with segments may change greatly as the frame changes. In order to ensure the correct assignments, the elements p_{ij}^{kl} of the permutation matrix P^{kl} are modified based on thresholding the cost matrix by τ_m so that they contain only the correct matches:

$$\overline{p}_{ij}^{kl} = \begin{cases} p_{ij}^{kl} & \text{if } c_{ij}^{kl} < \tau_m \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The result of the matching for each RAG across the frames $\mathcal{K}$ is encoded in the node existence matrix $\mathbf{M}$. Each column represents a frame with index $k \in \mathcal{K}$ while each row represents a segment. Thus, it evolves as the frames are processed. A sample node existence matrix is as shown in Fig. 2. In this case, for example, Node#10 has appeared throughout whole sequence. This is in contrast to some nodes that appear only for a very short period.

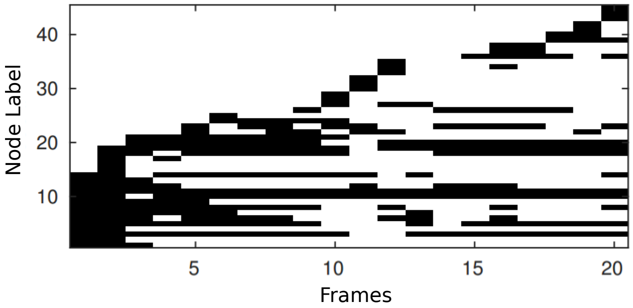


Figure 2: Node existence matrix encodes the states of presence or absence of nodes through time. Nodes are labeled with numbers and tracked through time. Black regions represent the nodes appeared at that particular frame.

5. Shot Boundary Detection

Generally, shots are separated by one of the following motion picture methods: cut, dissolve, fade, wipe or camera movements such as turning or zooming. Although some type of transitions causes abrupt changes between the disappearing and the appearing shot and can be accurately detected by straightforward frame-to-frame dissimilarity analysis, most of the changes are gradual and relatively hard to detect automatically. Here, we propose a coherency score metric based on the number of appearing and disappearing of nodes(segments) in the shots that can effectively identify any kind of transition at higher rates.

Coherency score is calculated at each frame through a sliding window of size τ_w. Value of τ_w is set depending on the frame rate, video resolution and segmentation parameters. Coherency is measured primarily based on the number of nodes emerging and disappearing within a window where each node is weighted with a parameter $\omega_i^{k^*}$ by how long it appeared across frames, how much its area and how positionally stable it is. These weights are updated accordingly at each frame. A coherency score φ_k based on these criteria is defined as in (4).

$$\varphi_k = \sum_{k^*=k-\tau_w}^{k+\tau_w} \sum_{i=1}^{|N^{k^*}|} \frac{1}{\omega_i^{k^*}(\alpha_i^{k^*} + \beta_i^{k^*})} \tag{4}$$

$$\alpha_i^k = \begin{cases} 1 & \text{if } i \in M^k, i \notin M^{k-1} \\ 0 & \text{otherwise} \end{cases}$$

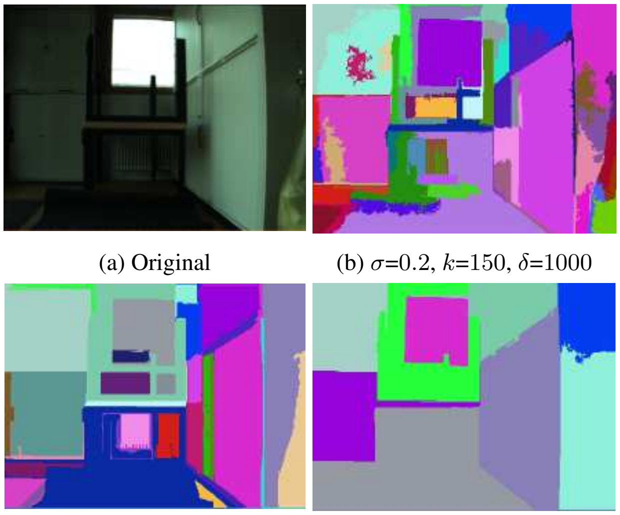
$$\beta_i^k = \begin{cases} 1 & \text{if } i \notin M^k, i \in M^{k-1} \\ 0 & \text{otherwise} \end{cases}$$

The validity of calculated coherency score depends on how accurately the nodes are tracked. Several factors as discussed in the experiments section is observed to affect tracking performance.

After the coherency score φ_k is calculated, it is used in deciding whether to start a new shot or to end the current shot or to continue with it. A simple reasoning is used in deciding what to do: A new shot is initiated if coherency is maintained consecutively τ_n times while the current shot ends if it cannot be observed τ_n times. Otherwise the current shot continues.

6. Segments Summary Graphs

Immediately after a shot boundary is detected, Segment Summary Graph is formed of the segments that are prevalent in that shot. This is determined based on the spatio-temporal coherence of the nodes of the RAGs associated with that shot. Spatial coherence is determined depending on the mean centroid and area of segments. Segments with small area or having high positional variance are deleted. Temporal coherence is determined depending on the temporal persistence of nodes and edges. Nodes are tracked throughout the sequence of frames and the ones that appear long enough thoughout the shot period, as specified by τ_n, are selected as candidate nodes. Similarly, edges that exist at least certain percentage τ_e are selected as candidate edges. Node and edge attributes of selected candidates are averaged and encoded in resulting SSGs. Fig. 5b shows constructed SSGs for each detected shots.



(a) Original

(b) σ=0.2, k=150, δ=1000

(c) σ=0.7, k=150, δ=1000

(d) σ=0.7, k=150, δ=5000

Figure 3: Graph based segmentation result for different parameters

7. Experimental Results

Our experimental results are obtained using a dataset that contains videos from the Open Video Project [18]. Videos are distributed among several genres and their duration varies from 1 to 4 min. The first step - namely the segmentation of video frames into homogenous color regions - is based on efficient graph based segmentation algorithm [10]. This method can produce coarse segments while keeping edge details in low variability image regions. The number and area of the segments depends on three parameters: smoothing factor σ, merging threshold k and minimum segment size δ. Using smaller σ produces jagged segments that are sensitive to small color variations as seen in Fig. 3b. It is observed from Fig. 3c and Fig. 3d that increasing δ reduces the total number of segments but creates oversimplified regions. As such, the parameters are adjusted as to generate segments that encircle prominent entities and coarse enough to omit insignificant and small objects with values $\sigma = 0.7$, $k = 150$ and $\delta = 1000$.

The validity of coherency analysis depends on accurately tracking the nodes of RAGs therefore matching the nodes of RAG correctly is crucial. The evaluation of graph matching performance is performed based on visual inspection. The average match ratio of two RAGs is calculated as 87% and correct match ratio is calculated as 80%. Here, the match percentage is the percentage of nodes across two consecutive RAGs that have been matched while the correct match percentage is the percentage of correctly matched nodes based on visual inspection. It is observed that match ratios are highly related to the consistency of segmentation and the visual difference between the contents of two frames. In other words, graph match ratio is higher in frames where the content is composed of stationary, less cluttered and textureless objects. Average node tracking performance based on visual inspection is shown in Fig. 4. It's already expected that the tracking accuracy to decrease as the number of frames increases. Several reasons can be stated: inconsistent segmentation, changing frame content and incorrect node matching. If the segmentation is not consistent between consecutive frames that means the objects are not segmented as same as in the previous frame, tracking will be implausible. Similarly, if the content between frames changes too much, maintaining an accurate tracking gets difficult.

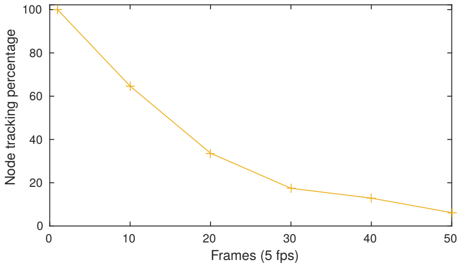


Figure 4: Node tracking accuracy

7.1. Sample Video Summarization Results

First, we consider the results of the proposed approach (SSG) on a sample video (5^{th} video from the Open Video dataset). Fig. 5 explains how the coherency based shot and boundary detection is performed. The node existence matrix is depicted in Fig. 5a where the horizontal axis is for the frame numbers and the vertical axis is for the node labels. Black stripes represent the nodes that appeared at the respective frames. It should be noted that nodes are tracked for the whole video even if they disappear and appear again. Coherency scores along with the detected shots and boundaries evolve as seen in Fig. 5c. Red regions indicate the shots and blue regions represent the shot boundaries. It is observed that in certain frames, the number of nodes increase suddenly while previous nodes disappear. Such frames are selected as the shot boundaries because the coherency score is below the threshold at these regions. The most coherent frame in each shot is selected as summary keyframe as shown in Fig. 5d in numbered circles. Finally, SSGs of each detected shot is illustrated in Fig.5b. Colored circles encode the nodes of SSG and they are selected based on several criteria within a set of RAGs related to each detected shots. In deciding which segments to include, temporal continuity, size and positional stability of segments are considered. In other words, intermittent, fast moving and small segments will be disregarded. Results demonstrate

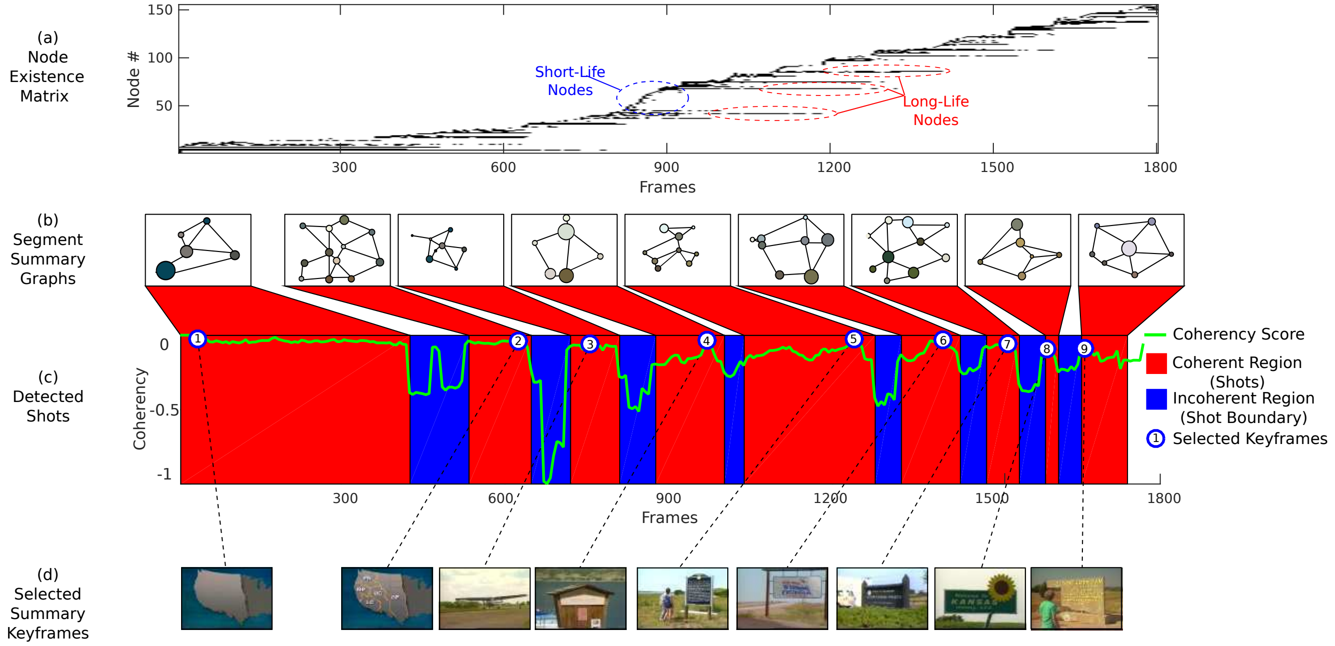


Figure 5: SSG: Detected shots

that SSG structure is suitable for encoding the prevalent entities and hence semantic analysis.

7.2. Comparative Results

In this section, the proposed SSG approach is compared with sparse dictionary (SD) based approach [4], VSUMM [6], Open Video Project storyboard (OVP) [18], Delaunay Clustering (DT) [20], STIMO [12] and On-line Minimum Sparse Reconstruction (OnMSR) [19] approaches. Recorded performance results of these approaches are adopted from [19]. For comparison purposes each video is downsampled at 5 fps and have 352x240 pixels resolution.

The evaluation is based on manually created user summaries where each video is summarized by 5 different users. In this process, they are oriented to select any number of frames to compose their summaries. Next, user generated summaries are compared with automatically generated summaries based on three metrics including precision, recall and F-score as defined in [19]:

$$Precision = \frac{n_{mAS}}{n_{AS}} \tag{5}$$

$$Recall = \frac{n_{mAS}}{n_{US}} \tag{6}$$

$$F - Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{7}$$

where n_{mAS} is the number of matching keyframes in an automatic summary, n_{AS} is the total number of keyframes in automatic summary and n_{US} is the total number of keyframes in user summary. Two frames are matched only if the visual content is similar and frame numbers are not apart from each other. Here, visual similarity of the frames is checked by visual inspection and maximum frame number difference is set to 60 frame (which corresponds to 2 seconds at 30fps) in order to be counted as matched. Precision reflects the percentage of matched keyframes over all automatically selected keyframes whereas recall shows the percentage of matched keyframes over all user selected keyframes. Good summarization should contain as many keyframes so that all important shots are represented and as few frames as possible so that there is no redundant keyframes that points to the same shot. F-score as defined in (6) is an effective metric as it balances the precision and the recall scores. The results are presented in Table 1. As the summarization results point out, our approach achieved the highest F-score rate over all other approaches.

Algorithms	Precision (%)	Recall (%)	F-score (%)
OVP	43	64	51.4
DT	47	50	48.5
STIMO	39	65	48.8
VSUMM	42	77	54.4
OnMSR	50	66	56.9
SSG	**56**	**75.9**	**64.4**

Table 1: Comparative summarization performances.

As a case study, we present the video summaries produced by all different approaches considered for compari-

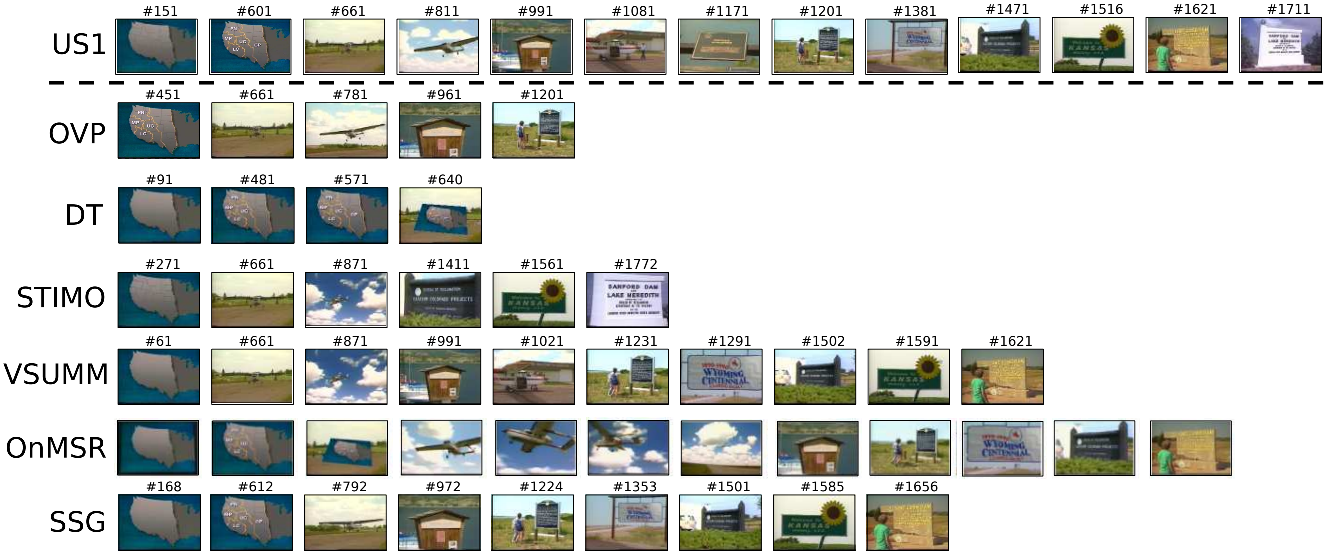


Figure 6: Comparative video summarization results.

Resolution	Frames/sec	Elapsed time per each step (sec/frame)		
		Segmentation	Graph Matching	Coherency Analysis
352x240	4.38	0.139 (92%)	0.0097 (6%)	0.0028 (2%)
176x120	9.51	0.085 (97%)	0.0019 (2%)	0.0011 (1%)

Table 2: Video Summarization Computation Time Results.

son in Fig. 6[1]. We have selected the 5^{th} video from the Open Video dataset for comparison. US1 shows the user summary keyframes. OVP and DT approaches selected the least number of frames as they represent only the beginning of the video. Our approach together with VSUMM and OnMSR approaches produced the closest summarization result to ground truth. All of the selected keyframes by our approach match with those of the user summary however it misses some of the user summary keyframes. This is because of the value of threshold tuned accordingly to achieve the best overall summarization performance.

The experiments were performed on a computer with 3.6 GHz Intel Core i7-4790. Average computation time per frame of each step is presented in Table 2. Segmentation is the most time consuming step whereas graph matching and coherency analysis spends only 2% and 1% of processing power, respectively. Segmentation process is directly related to frame resolution. For example, downsampling the video by 2 speeds up the processing by 4 times. In applications where the speed is a priority, higher frame rates can be achieved by downsampling the video while preserving the summarization performance. Original video can be processed at 4.89 frames/sec whereas video downsampled by 2 can be processed at 9.51 frames/sec which means that our algorithm is suitable real-time processing.

8. Conclusion

In this paper, we have introduced a novel approach to video summarization. The novelty of this approach is that shots are detected based on the coherency of consecutive region adjacency graphs as derived from respective video frames. Similar segments across consecutive region adjacency graphs are matched and tracked using an efficient graph matching technique. The coherency score associated with each region adjacency graph defines an effective metric to detect shots and any type of boundary with high accuracy. Experimental results with benchmark datasets demonstrate that the proposed method outperforms the state of the art approaches. Simultaneously, a novel shot representation model - referred to as the ‘segments summary graphs’- is introduced. The resulting segments summary graph encodes the segments and spatio-temporal relations that are prevalent in the shot and hence is amenable for further semantic analysis and understanding.

Acknowledgments

This work has been supported in part by TUBITAK EEEAG-115E380.

[1] This study could not include [14] as the associated codes are not available online.

References

[1] A. Aner and J. R. Kender. Video summaries through mosaic-based shot and scene clustering. In *Computer Vision*, pages 388–402. Springer, 2002.

[2] G. Boccignone, A. Chianese, V. Moscato, and A. Picariello. Foveated shot detection for video segmentation. *IEEE Transactions on Circ. and Sys. for Video Tech*, 15(3):365–377, 2005.

[3] E. Bulut and T. Capin. Key frame extraction from motion capture data by curve saliency. In *Computer Animation and Social Agents*, page 119, 2007.

[4] Y. Cong, J. Yuan, and J. Luo. Towards scalable summarization of consumer videos via sparse dictionary selection. *IEEE Trans. on Multimedia*, 14(1):66–75, 2012.

[5] C. Cotsaces, N. Nikolaidis, and I. Pitas. Video shot detection and condensed representation. a review. *IEEE Signal Proc. Mag.*, 23(2):28–37, 2006.

[6] S. E. F. de Avila, A. P. B. Lopes, A. da Luz, and A. de Albuquerque Araújo. Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pat. Rec. Letters*, 32(1):56–68, 2011.

[7] G. C. De Silva, T. Yamasaki, and K. Aizawa. Evaluation of video summarization for a large number of cameras in ubiquitous home. In *Proc. of the 13th annual ACM int. conf. on Multimedia*, pages 820–828. ACM, 2005.

[8] A. Divakaran, K. A. Peker, R. Radhakrishnan, Z. Xiong, and R. Cabasson. Video summarization using mpeg-7 motion activity and audio descriptors. In *Video Mining*, pages 91–121. Springer, 2003.

[9] G. Evangelopoulos, K. Rapantzikos, A. Potamianos, P. Maragos, A. Zlatintsi, and Y. Avrithis. Movie summarization based on audiovisual saliency detection. In *Conf. on 15th IEEE Int. Image Processing*, pages 2528–2531. IEEE, 2008.

[10] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. of Computer Vision*, 59(2):167–181, 2004.

[11] K. Fujimura, K. Honda, and K. Uehara. Automatic video summarization by using color and utterance information. In *IEEE Int. Conf. on Multimedia and Expo*, volume 1, pages 49–52. IEEE, 2002.

[12] M. Furini, F. Geraci, M. Montangero, and M. Pellegrini. Stimo: Still and moving video storyboard for the web scenario. *Multimedia Tools and Applications*, 46(1):47–69, 2010.

[13] S. Jouili, I. Mili, and S. Tabbone. Attributed graph matching using local descriptions. In *Advanced Concepts for Intelligent Vision Systems*, pages 89–99, 2009.

[14] J. Lee, J. Oh, and S. Hwang. Scenario based dynamic video abstractions using graph matching. In *Proc. of the 13th ann. ACM int. conf. on Multimedia*, pages 810–819. ACM, 2005.

[15] B. Li and M. I. Sezan. Event detection and summarization in sports video. In *IEEE Workshop on Content-Based Access of Image and Video Libraries*, pages 132–138. IEEE, 2001.

[16] R. W. Lienhart. Reliable dissolve detection. In *Photonics West Electronic Imaging*, pages 219–230. Int. Society for Optics and Photonics, 2001.

[17] D. Liu, G. Hua, and T. Chen. A hierarchical visual model for video object summarization. *IEEE Trans. on Pat. Analysis and Mac. Int.*, 32(12):2178–2190, 2010.

[18] G. Marchionini and G. Geisler. The open video digital library. *D-Lib Magazine*, 8(12):1082–9873, 2002.

[19] S. Mei, G. Guan, Z. Wang, S. Wan, M. He, and D. D. Feng. Video summarization via minimum sparse reconstruction. *Pat. Rec.*, 48(2):522–533, 2015.

[20] P. Mundur, Y. Rao, and Y. Yesha. Keyframe-based video summarization using delaunay clustering. *Int. J. on Digital Libraries*, 6(2):219–232, 2006.

[21] J. Nam and A. H. Tewfik. Detection of gradual transitions in video sequences using b-spline interpolation. *IEEE Trans. on Multimedia*, 7(4):667–679, 2005.

[22] C.-W. Ngo, Y.-F. Ma, and H.-J. Zhang. Video summarization and scene detection by graph modeling. *IEEE Trans. on Circuits and Systems for Video Technology*, 15(2):296–305, 2005.

[23] H. S. Sawhney and S. Ayer. Compact representations of videos through dominant and multiple motion estimation. *IEEE Trans. on Pat. Analysis and Mac. Int.*, 18(8):814–830, 1996.

[24] H.-C. Shih and C.-L. Huang. Msn: statistical understanding of broadcasted baseball video using multi-level semantic network. *IEEE Trans. on Broadcasting*, 51(4):449–459, 2005.

[25] B. L. Tseng, C.-Y. Lin, and J. R. Smith. Using mpeg-7 and mpeg-21 for personalizing video. *IEEE MultiMedia*, 11(1):42–52, 2004.

[26] F. Wang and C.-W. Ngo. Rushes video summarization by object and event understanding. In *Proc. of the int. workshop on TRECVID video summarization*, pages 25–29. ACM, 2007.

[27] W. Wolf. Key frame selection by motion analysis. In *IEEE Int. Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 1228–1231. IEEE, 1996.

[28] H. J. Zhang, J. Wu, D. Zhong, and S. W. Smoliar. An integrated system for content-based video retrieval and browsing. *Pattern recognition*, 30(4):643–658, 1997.