Proc. 24th International Conference on Multimedia Modeling (MMM 2018). Author's accepted version.
The final publication is available at Springer via https://doi.org/10.1007/978-3-319-73603-7_3

A motion-driven approach for fine-grained temporal segmentation of user-generated videos

Konstantinos Apostolidis, Evlampios Apostolidis, and Vasileios Mezaris

Information Technologies Institute / CERTH
6th km Xarilaou - Thermi, 57001, Thessaloniki, Greece
kapost@iti.gr, apostolid@iti.gr, bmezaris@iti.gr

Abstract. This paper presents an algorithm for the temporal segmentation of user-generated videos into visually coherent parts that correspond to individual video capturing activities. The latter include camera pan and tilt, change in focal length and camera displacement. The proposed approach identifies the aforementioned activities by extracting and evaluating the region-level spatio-temporal distribution of the optical flow over sequences of neighbouring video frames. The performance of the algorithm was evaluated with the help of a newly constructed ground-truth dataset, against several state-of-the-art techniques and variations of them. Extensive evaluation indicates the competitiveness of the proposed approach in terms of detection accuracy, and highlight its suitability for analysing large collections of data in a time-efficient manner.

1 Introduction

The recent advances in video cameras, combined with the widespread use of social networks (e.g. Facebook) and video sharing platforms (e.g. YouTube), led to a tremendous increase in the number of videos captured and shared by amateur users. Such user-generated videos (UGVs) can nowadays be recorded at any time and place with the help of smartphones and a variety of video cameras (such as GoPro action cameras) that can be attached to sticks, body parts or even drones. The ubiquitous use of video capturing devices supported by the convenience of the user to share videos through social networks and video sharing platforms, leads to a wealth of on-line available UGVs.

Analysing such video content, for generating high-level metadata that can be used for indexing and retrieval of it (e.g. concept and event detection), as well as for allowing fine-grained access to it (e.g. finding just the specific parts of videos that show a red sports car) is a requirement in many multimedia applications. The first step of most of such analysis pipelines is the identification of the video's temporal structure. For edited (i.e. professional) videos this typically corresponds to the detection of the video shots (i.e. sequences of frames captured uninterruptedly by a single camera) using a shot segmentation method, e.g. [2]. However, when dealing with UGVs the shot-level fragmentation is too coarse and often fails to reveal useful information about their structure, since UGVs are most commonly captured uninterruptedly, thus being single-shot videos. Motivated by this observation, we developed a motion-driven algorithm to identify visually coherent parts (called sub-shots in the sequel) of a single-shot video, that relate to different actions taking place during the video recording. The proposed approach extracts the optical flow between neighbouring frames and evaluates its spatial distribution over frame sequences, to detect sub-shots. The conducted experimental evaluations illustrate the time-efficiency and superiority of the algorithm against other state-of-the-art sub-shot segmentation techniques.

2 Related Work

Several methods dealing with the temporal segmentation of videos into sub-shots have been introduced, most of which can be grouped in two main classes of methodologies. The techniques of the first class consider a sub-shot as an uninterrupted sequence of frames within a shot that only have a small variation in visual content. Based on this assumption, they try to define sub-shots by assessing the visual similarity of consecutive or neighbouring video frames. A rather straightforward approach that evaluates frames' similarity using colour histograms and the x^2 test was described in [26], while a method that detects sub-shots of a video by assessing the visual dissimilarity of frames lying within a sliding temporal window using 16-bin HSV histograms (denoted as "Eurecom segmentation") was reported in [11]. A different approach [3] estimates the grid-level dissimilarity between pairs of frames and segments a video by observing that the cumulative difference in the visual content of subsequent frames indicates gradual change within a sub-shot; a similar approach was presented in [20]. The method of [25] estimates the brightness, contrast, camera and object motion of each video frame using YUV histograms and optical flow vectors, and defines sub-shot boundaries by analysing the extracted features through a coherence discontinuity detection mechanism on groups of frames within a sliding window.

The methods of the second class segment a video shot into sub-shots based on the rationale that each sub-shot corresponds to a different action of the camera during the video recording. Hence, these approaches aim to detect different types of camera activity over sequences of frames, and define these frame sequences as the different sub-shots of the video. An early, MPEG-2 compatible, algorithm that detects basic camera operations by fitting the motion vectors of the MPEG stream into a 2D affine model, was presented in [18]. Another approach that exploits the same motion vectors and estimates the camera motion via a multi-resolution scheme was proposed in [12]. More recently, the estimation of the affinity between pairs of frames for motion detection and categorization was a core idea for many other techniques. Some of them use the motion vectors of the MPEG-2 stream (e.g. [22]), while others compute the parameters of a 3×3 affine model by extracting and matching local descriptors [8] or feature points [24]. The dominant motion transformation between a pair of frames is then estimated by comparing the computed parameters against pre-defined models. [9] studies several approaches for optical flow field calculation, that include the matching of local descriptors (i.e. SIFT [21], SURF [4]) based on a variety of block matching algorithms, and the use of the PLK algorithm [6]. Contrary to the use of experimentally-defined thresholds for categorizing the detected camera motion, [16] describes a generic approach for motion-based video parsing that estimates the affine motion parameters, either based on motion vectors of the MPEG-2 stream or by applying a frame-to-frame image registration process, factorizes their values via Singular Value Decomposition (SVD) and imports them into three multi-class Support Vector Machines (SVMs) to recognize the camera motion type and direction between successive video frames. A variation of this approach [1], identifies changes in the "camera view" by estimating a simplified three-parameter global camera motion model using the Integral Template Matching algorithm [19]. Then, trained SVMs classify the camera motion of each frame, and neighbouring frames with the same type of camera motion are grouped together forming a sub-shot. Another threshold-less approach [17] aims to identify specific activities in egocentric videos using hierarchical Hidden Markov Models (HMM), while the algorithm of [14] combines the concept of "camera views" and the use of HMM for performing camera motion-based segmentation of UGVs. Finally, a study on different approaches for motion estimation was presented in [5].

Further to the aforementioned two general classes of methodologies, other approaches have been also proposed. [7] extracts several descriptors from the video frames (e.g. colour histograms and motion features) and subdivides each shot into sub-shots by clustering its frames using k-means clustering. [29, 10] utilize data from auxiliary camera sensors (e.g. GPS, gyroscope and accelerometers) to identify the camera motion type for every video sub-shot or a group of events in UGVs, while other approaches define sub-shots by extracting and processing 3D spatio-temporal slices [23] or through statistical analysis [15].

The proposed method is most closely related to [9], in the sense that motion information is described by computing the optical flow field using the PLK algorithm, while it is similar to [8], [24] and [16] in that motion information is again represented using the optical flow field (computed using other techniques, though). However, these previous approaches try to distinguish the type of camera motion via computationally-expensive techniques that involve the estimation of homography and affinity between pairs of frames, or the use of trained classifiers. In contrast, our algorithm efficiently identifies several kinds of video recording activities based on a lightweight process that finds the dominant motion in the four quartiles of the video frame, and compares the frame-level motion distribution against pre-defined motion models.

3 Proposed Method

The proposed algorithm segments a single-shot video into self-contained parts (called sub-shots) which exhibit visual continuity and correspond to individual elementary low-level actions that take place during the recording of the video. These actions include camera panning and tilting; camera movement in the 3D

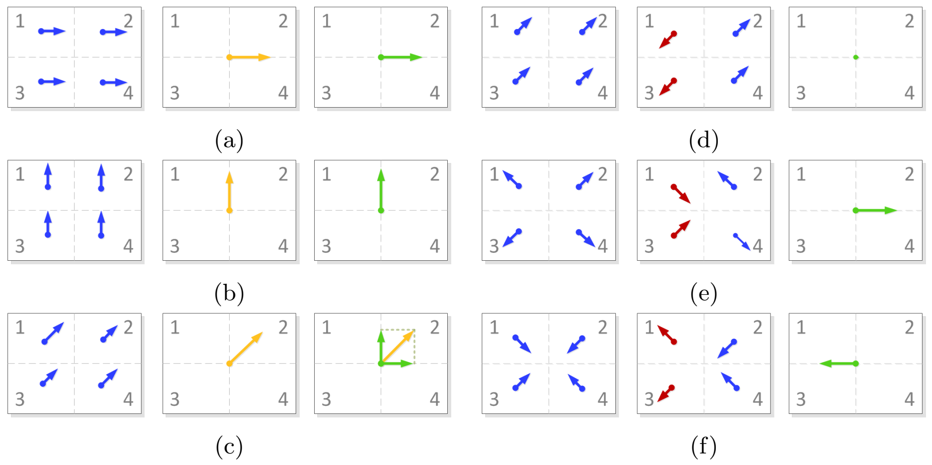

Fig. 1: Motion estimation process for (a) right displacement, (b) upward displacement, (c) diagonal displacement of the camera. Focal distance change estimation process in case of (d) displacement only at horizontal and vertical axes (similar to (c) - thus, no change in the z-axis), (e) forward displacement or zoom in, (f) backward displacement or zoom out.

Euclidean space; camera zoom in/out and minor or no camera movement. The detection of sub-shot boundaries and the identification of the performed action is based on the extraction and spatio-temporal analysis of motion information. The latter is computed using the optical flow between pairs of video frames. In particular each processed pair of frames initially undergoes an image resizing process that maintains the original aspect ratio of the video frames and makes their width equal to w. Following, each frame is spatially segmented into four quartiles. The most prominent corners in each quartile are then detected based on the algorithm of [28], and used for estimating the optical flow at the region-level by utilizing the PLK technique. Based on the extracted optical flow, a mean displacement vector is computed for each quartile, and the four spatially distributed vectors are treated as a region-level representation of the motion activity between the pair of analysed frames (left part of Fig. 1a, 1b, 1c).

For detecting and recognizing any displacement of the camera in the 2D space at the frame-level, the algorithm averages the four computed mean displacement vectors (middle part of Fig. 1a, 1b, 1c) and projects the resulting vector to the horizontal and vertical axis of the Euclidean space (right part of Fig. 1a, 1b, 1c). A horizontal-only camera displacement leads to a single x-axis vector (Fig. 1a), a vertical-only leads to a single y-axis vector (Fig. 1b), while a diagonal displacement results to a pair of x- and y-axis vectors (Fig. 1c). For identifying any camera activity at the depth level (i.e. the z-axis of the 3D space) the developed approach inverts the direction of the mean displacement vectors of the top- and bottom-left regions of the image (left and middle part of Fig. 1d, 1e, 1f), computes the vector sum of all four vectors and projects it on the x-axis (right part of Fig. 1d, 1e, 1f). As depicted in Fig. 1d, in case of camera movement at the horizontal and/or vertical axes only, the vector inversion process leads to a set of counterbalanced mean displacement vectors and thus, the magnitude of the projection is zero. However, in case of camera activity at the depth axis, the four mean displacement vectors do not maintain the same direction, but point either to the corners of the frame (Fig. 1e), forming a projection vector with positive magnitude, which indicates the existence of forward camera movement or camera zoom in operation; or to the centre of the frame (Fig. 1f), forming a projection vector with negative magnitude, that denotes the occurrence of backward camera movement or a camera zoom out operation.

Through the above mentioned process the proposed method computes for each pair of frames three values that represent the spatial displacement in x-, y- and z-axis. However, successive frames of a video, even with the standard frame-rate of 30fps, usually exhibit high visual similarity, which is even more true for videos of greater frame-rates that can be captured using modern smartphones or action cameras (e.g. GoPro cameras, which support video recoding up to 240fps). Guided by this fact, the aforementioned pair-wise motion estimation is not applied on every pair of consecutive video frames, but only on neighbouring frames selected through a sampling strategy with a fixed-step equal to 10% of the video frame-rate. Moreover, for facilitating the upcoming sub-shot segmentation analysis, the computed spatial displacement values, denoted as V_x, V_y and V_z in the sequel, are normalized in $[-1, +1]$ where: V_x (V_y) $= -1$ represents left (downward) displacement of frame pixels equal to 5% of the frame width (height), V_x (V_y) $= +1$ signifies right (upward) displacement of frame pixels equal to 5% of the frame width (height), V_x (V_y) $= 0$ denotes no displacement of frame pixels, $V_z = -1$ ($+1$) indicates increment (decrement) of the focal distance that causes inward (outward) spatial displacement of frame pixels equal to 5% of the frame's diagonal, and $V_z = 0$ indicates no change of the focal distance.

The normalized spatial displacement vectors V_x, V_y and V_z are then post-processed, as described in Algorithm 1, to detect the different sub-shots. Specifically, the values of each vector are initially subjected to low pass filtering in the frequency domain (sample rate equals video frame-rate; cut-off frequency empirically set as 1.0Hz), which excludes sharp peaks related to wrong estimation of the PLK algorithm or quick changes in the light conditions (top row of Fig. 2). Each of the filtered vectors V'_x, V'_y and V'_z is then processed for finding its intersection points with the corresponding axis, and the identified intersection points are stored in vectors I_x, I_y and I_z respectively (Fig. 2c). These intersection points are candidate sub-shot boundaries, since the video frames between a pair of consecutive intersection points exhibit a contiguous and single-directed camera movement, thus being a potential sub-shot according to the proposed approach. However, since most UGVs are captured by amateurs without the use of any professional equipment that ensures the stabilization of the camera, the developed algorithm filters-out fragments depicting minor motion by computing the total displacement over each fragment as the sum of the absolute values of the filtered displacement values V'_x, V'_y and V'_z of each pair of frames in the

Algorithm 1 Pseudo code of the proposed technique

```
Input: $V_x$, $V_y$, $V_z$: axes displacement vectors
Output: $O'$: set of sub-shot boundaries
 1: function ProcessVector($V$)
 2:     Low-pass filter $V$. Store in $V'$.
 3:     Detect intersection points in $V'$. Store in $I$.
 4:     Measure the total displacement between intersection points in $I$. Store in $D$.
 5:     Select fragments with displacement $D > t$ as sub-shots. Store in $B$.
 6: end function
 7: $B_x \leftarrow$ ProcessVector($V_x$)
 8: $B_y \leftarrow$ ProcessVector($V_y$)
 9: $B_z \leftarrow$ ProcessVector($V_z$)
10: Add in $O$ the $B_x$ and $B_y$ fragments.
11: Extend $O$ by adding $B_z$ fragments that do not coincide with $B_x$ and $B_y$ fragments.
    Mark remaining parts of the video as fragments with no or minor movement.
12: Discard fragments less than 1 sec. Store in $O'$.
```

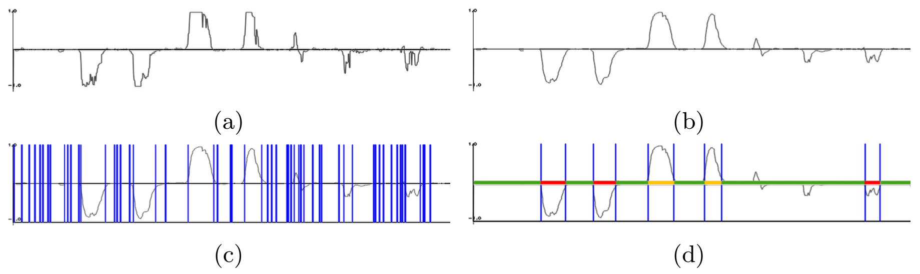

Fig. 2: Application of Algorithm 1 for a single normalized displacement vector: (a) initial values V_x, (b) low-pass filtered values V'_x, (c) detected candidate sub-shot boundaries in I_x, (d) selected sub-shot boundaries in B_x; red parts denote fragments with left displacement, orange parts denote fragments with right displacement and green parts denote fragments with no or minor movement.

fragment. This process results in vectors D_x, D_y and D_z, which store the total displacement score of each defined fragment in the x-, y- and z-axis respectively. The video fragments with total displacement score less than an experimentally-selected threshold t, are discarded. In our evaluations (Section 4) $t = 12$, which leads to the best performance (expressed by F-score in Fig. 3). The determined fragments of each axis are stored in vectors B_x, B_y and B_z (Fig. 2d). A simple fusion process is then applied, that takes the union O of B_x and B_y fragments, extends it by adding B_z fragments that do not temporally coincide (either completely or partially) with B_x and B_y fragments, and marks the remaining parts of the video as fragments with no or minor movement. The final output of the algorithm (O') is formed by discarding fragments with duration $< 1sec.$ through a process that equally dispenses their frames in the previous and the following sub-shot.

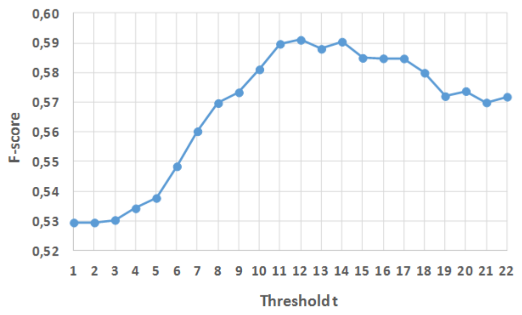

Fig. 3: The algorithm's segmentation effectiveness (expressed as F-score) for different values of threshold t. As shown in the graph, the best performance is achieved for $t = 12$, while any value in the range $[11, 14]$ leads to similar results.

4 Experiments and Results

Driven by the lack of publicly available datasets for evaluating the performance of the developed sub-shot segmentation algorithm[1], we built our own ground-truth dataset. This dataset is publicly available[2] and consists of:

- 15 single-shot videos of total duration 6 minutes, recorded in our facilities; these videos, denoted as "own videos" in the sequel, contain clearly defined fragments that correspond to several video recording activities.
- 5 single-shot amateur videos of total duration 17 minutes, found on YouTube; these videos are denoted as "amateur videos" in the sequel.
- 13 single-shot parts of known movies of total duration 46 minutes; these videos, denoted as "movie excerpts", represent professional video content.

Ground-truth segmentation of the employed dataset was created by human annotation of the sub-shot boundaries for each video, where each boundary indicates the end of a visually and temporally contiguous activity of the video recording device and the start of the next one (e.g. the end of a left camera panning, which is followed by a camera zooming). Overall, our dataset contains 674 sub-shot transitions. The performance of the developed algorithm was compared against other relevant state-of-the-art methods of the literature. Aiming to include in our evaluations several different categories of methods (presented in Section 2), we implemented:

- A straightforward approach (denoted S_HSV in the sequel) which assesses the similarity between subsequent video frames with the help of HSV histograms and x^2 distance, and a variation of it (denoted S_DCT) that represents the

[1] Some works reported in Section 2 use certain datasets (TRECVid 2007 rushes summarization, UT Ego, ADL and GTEA Gaze) which were designed for assessing the efficiency of methods targeting specific types of analysis, such as video rushes segmentation [3] and the identification of everyday activities [30] and thus, ground-truth sub-shot segmentation is not available for them.

[2] http://mklab.iti.gr/project/annotated-dataset-sub-shot-segmentation-evaluation

visual content of the video frames using DCT features and estimates their visual resemblance based on the cosine similarity.

- A method (denoted B_HSV) similar to [26], that selects the first frame of the video F_a as the base frame and compares it sequentially with the following ones using HSV histograms and x^2 distance until some frame F_b is different enough, then frames between F_a and F_b form a sub-shot, and F_b is used as the next base frame in a process that is repeated until all frames of the video have been processed; a variation of this approach (denoted B_DCT) that represents the visual content of the video frames using DCT features and estimates their visual resemblance based on the cosine similarity was also implemented.
- The algorithm of [8] (denoted A_SIFT), which estimates the dominant motion between a pair of frames based on the computed parameters of a 3×3 affine model through the extraction and matching of SIFT descriptors; furthermore, variations of this approach that rely on the use of SURF (denoted A_SURF) and ORB [27] (denoted A_ORB) descriptors were also implemented for assessing the efficiency of faster alternatives to SIFT.
- An implementation of the best performing technique of [9] (denoted A_OF), which computes the optical flow using the PLK algorithm and identifies camera movement by fitting it to a 2×2 affine model containing parameters that represent the camera pan, tilt, zoom and rotation actions.
- Variations of the local-feature-based approaches documented in [9], that rely on the extraction and matching of SIFT, SURF and ORB descriptors (denoted H_SIFT, H_SURF and H_ORB, respectively) or the computation of the optical flow using PLK (denoted H_OF), for estimating the dominant motion based on specific parameters of the homography matrix computed by the RANSAC method [13].

For each one of the tested approaches we counted the number of correct detections (where the detected boundary can lie within a temporal window around the respective ground-truth boundary, equal to twice the video frame-rate), misdetections and false alarms and expressed them in terms of Precision (P), Recall (R) and F-Score (F), similarly to [1, 2]. Time efficiency was evaluated by computing the ratio of processing time over the video's duration (a value below 1 indicates faster-than-real-time processing). All experiments were conducted on a PC with an i7-4770K CPU and 16GB of RAM.

Table 1 reports the evaluation results of each compared approach, both separately on each of the three parts of the dataset, as described above, and on the overall dataset. According to these results, and regarding the different implemented methodologies, approaches that estimate the dominant motion based on a homography matrix seem to be more effective compared to the methods that rely on affine models or the assessment of visual similarity, with the latter one being slightly better compared to the affine-based methods in terms of recall. Among the examined similarity-based techniques, the use of HSV histograms results in better performance in terms of precision; however, the utilization of DCT features leads to remarkably higher recall scores, and thus a better overall

Table 1: Evaluation results for different sub-shot segmentation approaches (P: Precision, R: Recall, F: F-score).

	"Own videos"			"Amateur videos"			"Movie excerpts"			Overall dataset		
Method	P	R	F	P	R	F	P	R	F	P	R	F
S_HSV	0.31	0.28	0.30	0.23	0.09	0.13	0.28	0.44	0.34	0.28	0.36	0.32
S_DCT	0.54	0.88	0.67	0.14	**0.86**	0.25	0.25	**0.84**	0.38	0.22	**0.84**	0.36
B_HSV	0.30	0.09	0.14	**0.55**	0.09	0.16	0.43	0.12	0.18	0.44	0.11	0.18
B_DCT	0.50	0.23	0.32	0.36	0.40	0.38	0.43	0.24	0.31	0.41	0.27	0.32
A_OF	0.41	0.68	0.50	0.20	0.82	0.31	0.30	0.78	0.43	0.27	0.78	0.40
A_SIFT	0.55	0.62	0.59	0.20	0.09	0.12	0.30	0.14	0.19	0.33	0.17	0.23
A_SURF	0.54	0.64	0.58	0.29	0.30	0.29	0.36	0.25	0.30	0.36	0.29	0.33
A_ORB	0.40	0.25	0.30	0.09	0.02	0.03	0.46	0.02	0.05	0.38	0.05	0.08
H_OF	**0.98**	0.62	0.76	0.26	0.67	0.38	0.41	0.58	0.47	0.37	0.60	0.45
H_SIFT	0.90	0.74	0.82	0.27	0.78	0.39	0.35	0.63	0.45	0.34	0.66	0.45
H_SURF	0.88	0.73	0.80	0.26	0.70	0.38	0.36	0.64	0.47	0.36	0.66	0.46
H_ORB	0.85	0.67	0.75	0.18	0.76	0.30	0.30	0.73	0.43	0.28	0.72	0.40
Proposed	0.96	**0.90**	**0.93**	0.42	0.71	**0.53**	**0.48**	0.64	**0.55**	**0.52**	0.70	**0.59**

performance (F-score). Concerning the implemented affine-based techniques, the most efficient is the one that relies on the optical flow, showing the highest recall scores in all different video categories and comparable precision scores with the other related methods. Regarding the suitability of local descriptors for computing an affine model that helps with the identification of the performed movement, SURF are the most effective ones, SIFT perform slightly worse, and ORB exhibit the weakest performance. With respect to the evaluated homography-based approaches, the use of different local descriptors or optical flow resulted in similar efficiency, with ORB being the least competitive descriptor due to lower precision. The last row of Table 1 shows that the proposed algorithm is the best-performing one, achieving the highest F-score on all dataset parts and on the overall dataset. On the first collection of videos, the developed technique also exhibits the highest recall score, with the S_DCT being the second best, while its precision is slightly lower than the one achieved by the H_OF method. However, these two methods have lower precision and recall scores, respectively, resulting to a significantly lower overall performance. On "Amateur videos" the developed technique is again the best performing one, while the B_HSV and the S_DCT methods, that presented competitive precision and recall respectively, achieved significantly lower overall performance. Similar efficiency is observed when analysing single-shot parts of professional movies; the proposed approach is the best in terms of F-score and precision. All the above are reflected in the last three columns of Table 1, which show the superiority of the developed method over the other evaluated techniques in the overall dataset. An indicate example of how the algorithm segments a part of a UGV recorded by a camera that is moving right and then upwards, is presented in Fig. 4.

With respect to the time-efficiency, as shown in Table 2, the more straightforward approaches that segment a video based on the visual resemblance of

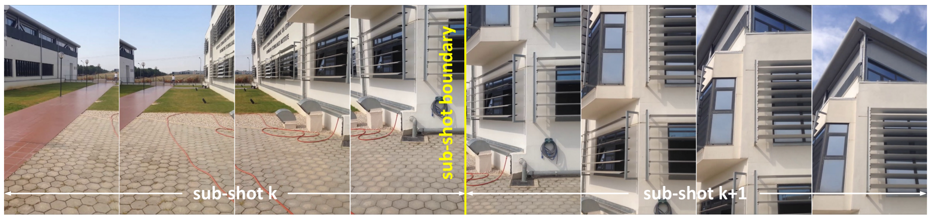

Fig. 4: A sequence of video frames (sampled for space and presentation efficiency) segmented by the proposed algorithm into two sub-shots; one related to a horizontal and one related to an upward camera movement.

video frames are faster that methods computing the parameters of affine models or homography matrices, as expected. Moreover, the use of DCT features outperforms the HSV histograms, while the extraction and matching of complex local descriptors (SIFT and SURF) is more computationally expensive compared to the matching of binary descriptors (ORB) or the extraction of optical flow for computing the affine or homography matrices. As shown, the proposed approach exhibits competitive time performance, being a bit slower than the straightforward similarity-based methods and faster than almost the entire set of the evaluated affine- and homography-based techniques. Its time efficiency permits sub-shot segmentation to be performed nine times faster than real-time analysis, while this performance can be further improved by introducing simple parallelization in the algorithm's execution. In fact, a multi-threaded software implementation of the proposed technique splits the group of analysed frames into four different and non-overlapping parts which are being processed (i.e. for extracting the optical flow among each pair of frames) in parallel on the CPU. The lightweight post processing of the computed displacement vectors for motion detection and recognition is still carried out using a single thread. Experiments on the same dataset showed 267% speed-up compared to the single-thread version, which means that the analysis of a single-shot video with the multi-thread implementation of the algorithm takes only 4.1% of the video's duration.

Table 2: Time-efficiency of the evaluated sub-shot segmentation approaches.

Method	S_HSV	S_DCT	B_HSV	B_DCT	A_OF	A_SIFT	A_SURF	A_ORB	H_OF	H_SIFT	H_SURF	H_ORB	Proposed
Proc. time % of video length	7.1	**2.9**	3.8	6.7	7.8	127.2	56.3	12.7	14.5	132.6	70.2	16.1	11.1

The above findings document that the proposed algorithm combines the time-efficiency of similarity-based approaches that rely on the extraction of lightweight visual descriptors (such as colour histograms and DCT features) with the detection effectiveness of more complex state-of-the-art techniques that estimate the dominant motion with the help of affine transformations and image homography. Moreover, the developed approach ensures the highest accuracy over different types of single-shots videos, while its time-efficiency makes it suitable for application in large collections of videos or real-time analysis of multiple video streams.

5 Conclusions

In this paper we proposed a framework for motion-driven sub-shot segmentation of UGVs and released a new dataset for evaluating sub-shot segmentation algorithms. The developed algorithm detects and recognizes several different types of video recording activities, such as camera pan, tilt, zoom and displacement, by computing the optical flow between neighbouring frames. Experimental evaluations showed that the developed segmentation algorithm outperforms other, more complex methods that rely on the extraction and matching of local descriptors, while it maintains the time-efficiency of more straightforward similarity-based approaches, being several times faster than real-time analysis.

6 Acknowledgements

This work was supported by the EU's Horizon 2020 research and innovation programme under grant agreement H2020-732665 EMMA.

References

1. Abdollahian, G., et al.: Camera motion-based analysis of user generated video. IEEE Trans. on Mult. 12(1), 28–41 (2010)
2. Apostolidis, E., et al.: Fast shot segmentation combining global and local visual descriptors. In: Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing. pp. 6583–6587 (2014), software available at http://mklab.iti.gr/project/video-shot-segm
3. Bai, L., et al.: Automatic summarization of rushes video using bipartite graphs. Mult. Tools and Appl. 49(1), 63–80 (2010)
4. Bay, H., et al.: Surf: Speeded up robust features. Int. Jour. of Comp. Vis. pp. 404–417 (2006)
5. Benois-Pineau, J., et al.: Motion estimation in colour image sequences. In: Advanced Color Image Processing and Analysis, pp. 377–395. Springer (2013)
6. Bouguet, J.Y.: Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. Intel Corporation 5(1-10), 4 (2001)
7. Chu, W.T., et al.: Video copy detection based on bag of trajectory and two-level approximate sequence. In: Proc. of the Comp. Vis., Graph. and Image Processing Conf. (2010)
8. Cooray, S.H., et al.: An interactive and multi-level framework for summarising user generated videos. In: Proc. of the 17th ACM Int. Conf. on Mult. pp. 685–688 (2009)
9. Cooray, S.H., et al.: Identifying an efficient and robust sub-shot segmentation method for home movie summarisation. In: 10th Int. Conf. on Intell. Syst. Design and Appl. pp. 1287–1292 (2010)

10. Cricri, F., et al.: Multimodal event detection in user generated videos. In: IEEE Int. Symposium on Mult. pp. 263–270 (2011)
11. Dumont, E., et al.: Rushes video summarization using a collaborative approach. In: Proc. of the 2nd ACM TRECVID Vid. Summar. Workshop. pp. 90–94 (2008)
12. Durik, M., et al.: Robust motion characterisation for video indexing based on mpeg2 optical flow. In: Int. Workshop on Content-Based Mult. Indexing. pp. 57–64 (2001)
13. Fischler, M.A., et al.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. ACM Communications 24(6), 381–395 (1981)
14. González-Díaz, I., et al.: Temporal segmentation and keyframe selection methods for user-generated video search-based annotation. Expert Syst. Appl. 42(1), 488–502 (2015)
15. Guo, Y., et al.: Selecting video key frames based on relative entropy and the extreme studentized deviate test. Entropy 18(3), 73 (2016)
16. Haller, M., et al.: A generic approach for motion-based video parsing. In: 15th European Signal Processing Conf. pp. 713–717 (2007)
17. Karaman, S., et al.: Hierarchical hidden markov model in detecting activities of daily living in wearable videos for studies of dementia. Mult. Tools and Appl. 69(3), 743–771 (2014)
18. Kim, J.G., et al.: Efficient camera motion characterization for mpeg video indexing. In: Proc. of the IEEE Int. Conf. on Mult. and Expo. vol. 2, pp. 1171–1174 (2000)
19. Lan, D.J., et al.: A novel motion-based representation for video mining. In: Proc. of the Int. Conf. on Mult. and Expo. pp. 469–72 (2003)
20. Liu, Y., et al.: Rushes video summarization using audio-visual information and sequence alignment. In: Proc. of the 2nd ACM TRECVID Vid. Summar. Workshop. pp. 114–118 (2008)
21. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proc. of the 7th IEEE Int. Conf. on Comp. Vis. vol. 2, pp. 1150–1157 (1999)
22. Mei, T., et al.: Near-lossless semantic video summarization and its applications to video analysis. ACM Trans. Mult. Comput. Commun. Appl. 9(3), 16:1–16:23 (2013)
23. Ngo, C.W., et al.: Video summarization and scene detection by graph modeling. IEEE Trans. on Circ. and Syst. for Video Tech. 15(2), 296–305 (2005)
24. Nitta, N., et al.: Content analysis for home videos. ITE Trans. on Media Tech. and Appl. 1(2), 91–100 (2013)
25. Ojutkangas, O., et al.: Location Based Abstraction of User Generated Mobile Videos, pp. 295–306. Springer Berlin Heidelberg (2012)
26. Pan, C.M., et al.: NTU TRECVID-2007 fast rushes summarization system. In: Proc. of the 1st ACM TRECVID Vid. Summar. Workshop. pp. 74–78 (2007)
27. Rublee, E., et al.: Orb: An efficient alternative to sift or surf. In: Proc. of IEEE Int. Conf. on Comp. Vis. pp. 2564–2571 (2011)
28. Shi, J., et al.: Good features to track. In: Proc. of the IEEE Conf. on Comp. Vis. and Patt. Recogn. pp. 593–600 (1994)
29. Wang, G., et al.: Motch: An automatic motion type characterization system for sensor-rich videos. In: Proc. of the 20th ACM Int. Conf. on Mult. pp. 1319–1320 (2012)
30. Xu, J., et al.: Gaze-enabled egocentric video summarization via constrained submodular maximization. In: Proc. of the IEEE Conf. on Comp. Vis. and Patt. Recogn. pp. 2235–2244 (2015)