Shot Detection and Motion Analysis for Automatic MPEG-7 Annotation of Sports Videos

Giovanni Tardini, Costantino Grana, Rossano Marchi, Rita Cucchiara

Department of Information Engineering, University of Modena and Reggio Emilia,
Via Vignolese 905/b, 41100 Modena, Italy, {surname.name}@unimore.it

Abstract. In this paper we describe general algorithms that are devised for MPEG-7 automatic annotation of Formula 1 videos, and in particular for camera-car shots detection. We employed a shot detection algorithm suitable for cuts and linear transitions detection, which is able to precisely detect both the transition's center and length. Statistical features based on MPEG motion compensation vectors are then employed to provide motion characterization, using a subset of the motion types defined in MPEG-7, and shot type classification. Results on shot detection and classification are provided.

1 Introduction

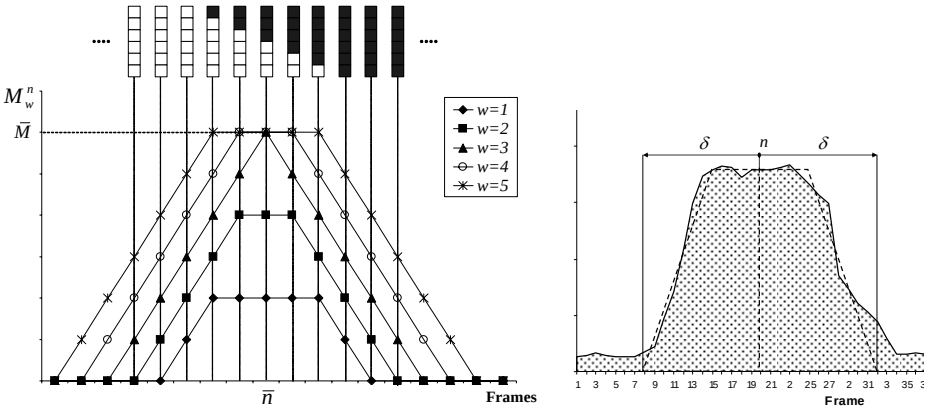
Video annotation is one of the primary processes in the life cycle of multimedia digital libraries. Automatic annotation must provide a description of the meaningful parts of the video in a standard way to be suitable used in further accessing and content based retrieval processes. In this framework, the MPEG-7 standard is becoming very popular.

Annotation must cope with an available ontology of concepts endowed in the video that are often defined by the users of digital libraries. In sports video the ontology can be easily defined being the rules of the play and the appearance of the video predictable and with periodical occurrence.

In this paper we describe general algorithms that are devised for MPEG-7 automatic annotation of Formula 1 videos, and in particular for *camera-car* shots detection. This is a challenging task for many reasons: strong camera motion as well as objects motion is present, and color features are not always important cues for shot classification, since they discriminate between cars and teams but not between events.

Despite of the very specific application, the algorithms here proposed are very general and can be applied in many contexts where shots with a specific motion type (e.g. zooms) must be detected.

In this work we followed the standard approach for edited video annotation that consists in an initial shot detection step and then in shot classification according with certain visual features. We employed a shot detection algorithm suitable both for cuts and linear transitions detection, which, unlike other approaches in literature, is able to



(a) (b)

Fig. 1. (a) Values of M_w^n for an ideal linear transition with $L = 5$ at varying *w*. (b) Values of M_w^n in a real case with a chosen *w* value.

precisely detect both the transition's center and length. Then, motion features are extracted to provide a proper classification and motion characterization of shots.

2 Shot detection

The first step in edited video analysis and characterization is shot detection. In recent years most techniques concentrated on the compressed domain [1]. These works, to obtain faster analysis, employ only information directly available from the MPEG streams, but comparative studies have demonstrated that they perform much worse on gradual transitions [2]. This is especially true in presence of scene with strong motion. Therefore, latest research on video segmentation is particularly focused on gradual transitions.

In [3] a linear transition model is exploited, but the author doesn't deal with the choice of the length of the window. A more refined approach is proposed in [4], where authors deal with long transitions, while in [5] the author expose a comparative study of most of the metrics used in shot detection approaches, both in compressed and uncompressed domain and then proposes an algorithm to detect both abrupt and gradual transitions, with an algorithm whose performances are strictly dependent on the search window length. The authors of [6] describe a neural network approach, trained with a dissolve synthesizer. The classifier detects possible dissolves at multiple temporal scales, and merges the results with a winner-take-all strategy. The algorithm works on contrast-based features, as well as color-based features, and has given good result compared to standard approaches based on edge change ratio.

Before describing our algorithm, it's useful to stress some properties of linear transitions. Let's consider two shots in a video sequence, the first one ending at frame *e*, and the second one starting at frame *s*, with $e < s$, between which a transition

occurs. Two hypotheses are made: the first one is that a feature $F(n)$ exists for each frame I_n, with the characteristic of being discriminating and constant for each shot; the second hypothesis is that the transition is linear and L frames long, where $L = s - e + 1$. This model includes abrupt cuts too, as transitions with length $L = 0$.

The transition center is defined as $\bar{n} = (e+s)/2$ and may correspond to a non integer value, that is an inter-frame position. In order to detect the linear transition, we define a function M_w^n which is a difference measure for the feature F. This function is computed with sub-frame granularity and is centered on frame or half-frame n, with $2n \in \mathbb{N}$, and with window size $2w \in \mathbb{N}$.

In Fig. 1 we see an example of an ideal linear transition with L=5, from a white to a black image. If the transition is linear according with the previous hypotheses, the shape of function M is an isosceles trapezoid, centered in $\bar{n}$, that degenerates into a triangle when $2w = L + 1$. It's easy to verify that, given the model and M_w^n, each side extends over $\min(2w, L+1)$ frames, and the minor base is $|2w-(L+1)|$ long. It's also possible to show that:

$$\begin{aligned} M_w^{\bar{n}} &< \bar{M} \quad \text{if } 2w < L+1 \\ M_w^{\bar{n}} &= \bar{M} \quad \text{if } 2w \geq L+1 \end{aligned} \tag{1}$$

where $\bar{M} = \max_{w,n} M_w^n$. We define $\psi_{w,L}^n(i)$ the ideal trapezoidal shaped function with the described properties. In the real case, it's not possible to obtain an ideal trapezoid from the data, so we have to look for the parameters that provide the best matching between data and the ideal $\psi_{w,L}^n(i)$ function. To solve this optimization problem, we employ an algorithm constructed of two iteratively repeated steps: the first one searches for the center position n and transition length L, assuming a fixed window size $2w$, which is then optimized by the second step, exploiting the property of Eq. 1. The two steps are iteratively repeated, progressively decreasing the estimate error.

The first step begins with a small window $\bar{w}$, and the best trapezoid is searched moving the center n, and trying different values for L. The trapezoid extends over $\delta = \min(2w, L+1) + |w-(L+1)/2|$ frames on the left and on the right of the center frame. For each couple of n and L the following measure is computed:

$$\Phi_{\bar{w},L}^n = \sum_{i=n-\delta}^{n+\delta} \min\left(M_{\bar{w}}^i, \psi_{\bar{w},L}^n(i)\right) - \sum_{i=n-\delta}^{n+\delta} \left|M_{\bar{w}}^i - \psi_{\bar{w},L}^n(i)\right| \tag{2}$$

After finding the trapezoid which maximizes $\Phi_{\bar{w},L}^n$, we consider $\bar{n}$ the candidate transition center. Observing Fig. 1, the value of M_w^n in the ideal case linearly grows with the window w, up to the window corresponding to $w = (L+1)/2$ and successively it is stable, leading to a horizontal straight line in the graph.

We employ this property in the second step of the algorithm to give a different estimate of the transition length by finding the smallest window $2w$ that maximizes M_w^n. To provide a more robust technique for the real case, the tilt change of the graph is searched by optimizing the function:

$$Z_w^{\bar{n}} = \sum_{i=0}^{w} \left| M_i^{\bar{n}} - \frac{M_w^{\bar{n}}}{w} i \right| + \sum_{i=w+1}^{W} \left| M_i^{\bar{n}} - M_w^{\bar{n}} \right| \tag{3}$$

where W is the maximum size that a transition can assume. The w value that minimizes $Z_w^{\bar{n}}$ is then used for the next iteration of the first step.

Given the transition length L and its center $\bar{n}$, as detected by the algorithm, we must verify how much the real data fit to the linear transition model. To this aim, we define an error measure as

$$err_{\bar{w}}^{\bar{n}} = \frac{1}{4\bar{w}+1} \sum_{i=-2\bar{w}}^{2\bar{w}} \left| M_{\bar{w}}^{\bar{n}+i} - \psi_{\bar{w},L}^{\bar{n}}(\bar{n}+i) \right| . \quad (4)$$

Here we assume $L = 2w-1$, which causes $\psi_{w,L}^{n}(i)$ to become a triangular shaped function. The error sum is divided by the triangle's base to obtain a measure independent from the transition length. We also introduce the ratio

$$r_{\bar{w}}^{\bar{n}} = \frac{Peak_{\bar{w}}^{\bar{n}}}{err_{\bar{w}}^{\bar{n}}}, \; Peak_{\bar{w}}^{\bar{n}} = M_{\bar{w}}^{\bar{n}} - \min\left(M_{\bar{w}}^{\bar{n}-2\bar{w}}, M_{\bar{w}}^{\bar{n}+2\bar{w}} \right). \quad (5)$$

The Peak value measures the height of the center value with respect to the lower of the two values of M in correspondence to the extremes of the triangle, and provides information on the transition significance, while the ratio provides a normalized estimate of the sequence similarity with a linear transition. These two values are employed to discriminate true and false transitions.

The algorithm could be applied to every frame of the analyzed video sequence, but it is computationally quite expensive. Thus, we employ a fast pre-processing algorithm to discard frames which are very unlikely to be part of a transition. In our experiments, we used the linear discriminant analysis on MPEG features (DC coefficients, number of intra, forward, backward, and interpolated macro-blocks) and set the threshold to obtain a very high recall with lower precision rates.

3 Motion characterization

For content analysis, several approaches were developed for camera motion characterization in the spatial domain, and some used MPEG motion vectors as an alternative to optical flow [7]. Akutsu et al. [8] presented a camera operation recognition method based on the analysis of motion vector fields by matching motion vector fields with predefined models in Hough space for different types of camera operations. A method to analyze the optical flow in a decomposed manner (projected x and y components were used) was proposed by [9].

A robust statistical method with a six-parameter affine motion model was developed by Bouthemy et al. [10] to detect shot change and camera motion. The use of a three-parameter motion model and a least-squares technique for estimation of camera operation was proposed in [11]. In [12], the spatiotemporal image sequence is constructed by arranging each frame close to the other and forming a parallelepiped with time being the third dimension. Camera operations are recognized by texture analysis of the different faces with the 2D discrete Fourier transform.

It has been verified in literature that MPEG motion compensation vectors are suitable for motion analysis applications, and allow a fast analysis of the basic motion properties of frames. In [13], for example a rough image subdivision in quadrants was employed to perform image queries on videos. Similarly, we chose to provide a frame level motion description by dividing the image in 4 quadrants, and for each one we

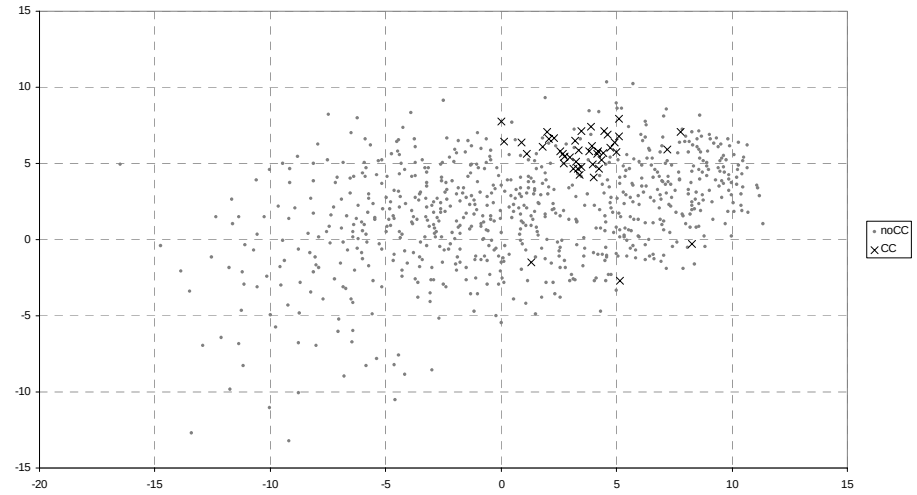


Fig. 2. Sammon's mapping of 755 classified shots

computed three motion features based on the average motion vector: its magnitude, its angle and the deviation of motion vectors from the average:

$$\Delta^i = \sqrt{\sum_{j \in MV^i} \left(\min\left(\left| \alpha_j^i - \bar{\alpha}^i \right|, 2\pi - \left| \alpha_j^i - \bar{\alpha}^i \right| \right) \right)^2} \tag{6}$$

where, with respect to quadrant *i*, MV^i is the set of motion vectors, α_j^i is the direction of the j^{th} motion vector and $\bar{\alpha}^i$ is the direction of their average. Extra care must be taken when using motion vectors. In MPEG video frames, boundary blocks and large smooth areas are most likely to have erroneous motion vectors. Usually, some kind of morphological or median filter should be applied to the motion vector field to remove outliers, before they are used for analysis and indexing. We exploit the Extended Vector Median (EVM), as in [14], to filter the motion vectors in a 5x5 window, that is:

$$mv_{EVM} = \begin{cases} mv_{AVE} = \dfrac{1}{N} \sum_{i=1}^{N} mv_i & \text{if } \sum_{i=1}^{N} \left\| mv_{AVE} - mv_i \right\| \le \sum_{i=1}^{N} \left\| mv_{VM} - mv_i \right\| \\ mv_{VM} & \text{otherwise} \end{cases}, \tag{7}$$

with

$$mv_{VM} \in MV : \sum_{i=1}^{N} \left\| mv_{VM} - mv_i \right\| \le \sum_{i=1}^{N} \left\| mv_j - mv_i \right\|, \forall mv_j \in MV \ . \tag{8}$$

Even if this choice is not of immediate understanding, it has proved to be definitely much more robust than the simple median to situations in which poor motion vectors are available, and the directions are scattered all around. Conversely, it is not misled by single spurious vectors in case of well defined structured motions. With this approach we extract 3 features per quadrant, that is, a total of 12 features per frame.

Motion features thus extracted are exploited to provide both classification and motion characterization of the shots: the classification is based upon the type of

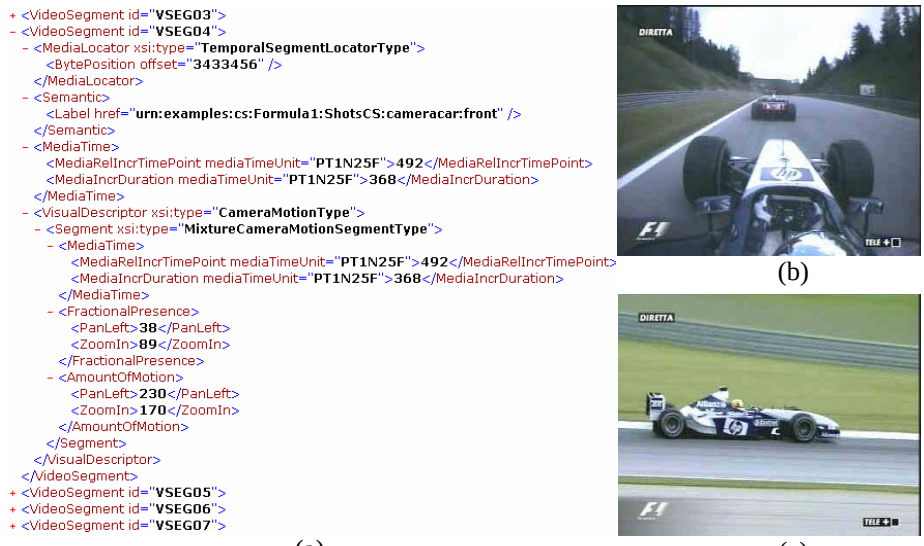


Fig. 3. (a) Example of the MPEG-7 annotation produced by our system. (b) Example of camera car shot. (c) Example of PTZ camera shot

camera from which the shot is taken, i.e. camera-car or PTZ camera, while motion characterization is constituted by statistics on the prominent motion types in the shots.

For the classification task, we just aggregated the 12 motion features of all the frames within the shot, by taking the mean feature vector and the standard deviation of every feature. This quite simple summary provides information on the shot motion main characteristics, but also allows describing if the observed motion is consistent along the whole duration of the shot. In Fig. 2, the 24 dimensional space of an entire Formula 1 video is shown, after projection into two dimensions by means of Sammon's mapping, that is a Non Linear Projection (NLP) procedure for data visualization, which tries to preserve the inter point distances [15]. It is possible to observe that, even if some outliers are present, the Camera Car data clusters together, reassuring on the descriptive power of the motion features chosen. Linear Discriminant Analysis (LDA) was employed to produce a separating hyper plane for the Camera Car shot description.

For the motion characterization task, we considered the five most represented camera movements in Formula 1 videos, which are, employing MPEG-7 terminology, *Fixed* (no motion), *Pan Left*, *Pan Right*, *Zoom In*, and *Zoom Out*. We disregarded *Tilt*, *Rotate* and all the camera position movements (*Track*, *Boom* and *Dolly*) because these represent very rarely observed events and do not appear alone, that is we always observe them in combination with other movements. Since, differently from before, no simple clustering was evident in NLP, we classified the camera movements by k-NN (with k=3) using a set of graphically selected prototypes. Each shot is annotated with the percentages of the different motion types (by taking into account all the frames constituting the shot except the transitions) and their respective amounts of motion, computed with the means of motion vectors' intensities of the quadrants.

4 MPEG-7 Stream Description

The content of the analyzed video clip is described using the MPEG-7 standard. In Fig. 3 an example of the output is shown. The video is divided in a set of instances of *VideoSegmentType* Descriptor Schema (DS), one for each shot detected by our linear transition detection algorithm. The start of each shot (after the transition's end) is located in the file by its position in bytes (*BytePosition* local type) and a reference in time (*MediaTimeType* DS), expressed in number of frames at a specific frame rate (in the example 25 fps). Each shot has a duration equal to the shot's length excluding the transitions at the beginning and at the end of it.

The type of camera used in the shot is referred from the cameras classification scheme as a semantic descriptor (*SemanticType* DS).

The MPEG-7 descriptor schema for camera motion (*CameraMotionType* DS) is briefly defined as follows: for each of the possible camera movements (Pan, Tilt, Zoom, Roll, etc.) both a fractional presence and an amount of motion can be specified. The first is the fraction of the total duration for which a certain motion type is present, and the second describes the average speed of the motion type. We used a single camera motion description for each video segment, which overlaps with the entire segment, but multiple descriptors for each segment could be used as well.

Using MPEG-7 video content description allows our indexes to be compatible with other annotation tools such, as IBM's VideoAnnex [16], and to have a description which can be expanded in a later time without compatibility problems.

5 Results

During the shot detection algorithm development, we used a series of Formula 1 selected sequences as training examples for the choice of both the thresholds and the histogram and spatial metrics linear combination coefficients. The tests here described are evaluated against a ground truth dataset composed of about 125.000 training frames and 160.000 test frames, from a Formula 1 TV videos Digital Library. Here, the pre-processing algorithm selected 1754 possible transitions, including 175 real linear transitions and 539 cuts. Within this dataset, for gradual transitions our algorithm reached a 90% of recall and 82% precision, while for abrupt cut the obtained values are 97% and 90%.

Shot classification was later applied to the obtained segmentation to identify Camera Car instances. Cross-validated LDA provided 88.4% recall and 80.9% precision. Higher recall rates can be set, depending on the penalty assigned to false detections: for example in our case we missed 5 over 43 Camera Car shots, but we could get 4 more correctly classified at the price of 21 more false positives. The users feeling was that it was better to have more false positives than to have to search manually in the whole video a Camera Car shot they knew it was there.

The further motion characterization allowed interesting and often satisfactory query capabilities and similarity searches (e.g. one could query all the shots with more than 60% of panning), but accurate analysis on the test set are currently being performed.

Acknowledgements

The project is funded by the European Network of Excellence DELOS of the VI Framework Program. We thank Ferrari S.p.A. for the video database availability.

References

1. Pei, S.-C., Chou, Y.-Z.: Efficient MPEG Compressed Video Analysis Using Macroblock Type Information. IEEE Trans. Multimedia 1 (1999) 321–333
2. Gargi, U., Kasturi, R., Strayer, S.H.: Performance Characterization of Video-Shot-Change Detection Methods. IEEE Trans. Circuits Syst. Video Technol. 10 (2000) 1–13
3. Yeo, B.-L., Liu, B.: Rapid Scene Analysis on Compressed Video. IEEE Trans. Circuits Syst. Video Technol. 5 (1995) 533–544
4. Heng, W.J., Ngan, K.N.: Long transition analysis for digital video sequences. Circuits Syst. Signal Process. 20 (2001) 113–141
5. Bescos, J.: Real-Time Shot Change Detection Over Online MPEG-2 Video. IEEE Trans. Circuits Syst. Video Technol. 14 (2004) 475–484
6. Lienhart, R., Zaccarin, A.: A System for Reliable Dissolve Detection in Videos. In: Proc. Int. Conf. Image Proc. (2001) 406–409
7. Patel, N.V., Sethi. I.K.: Video shot detection and characterization for video databases. Pattern Recognit. (Special Issue on Multimedia) 30 (1997) 583–592
8. Akutsu, A., Tonomura, Y., Hashimoto, H., Ohba, Y.: Video indexing using motion vectors. In: Proc SPIE (Visual Commun Image Process) 1818 (1992) 1522–1530
9. Xiong, W., Lee, J.C.-M.: Efficient scene change detection and camera motion annotation for video classification. Comput. Vis. Image Underst. 71 (1998) 166–181
10. Bouthemy, P., Gelgon, M., Ganansia, F.: A unified approach to shot change detection and camera motion characterization. IEEE Trans. Circuits Syst. Video Technol. 9 (1999) 1030–1044
11. Milanese, R., Deguillaume, F., Jacot-Descombes, A.: Efficient segmentation and camera motion indexing of compressed video. Real-Time Imaging 5 (1999) 231–241
12. Maeda, J.: Method for extracting camera operations to describe sub-scenes in video sequences. In: Proceedings of IS&T/SPIE conference on digital video compression on personal computers: algorithm and technologies, San Jose, 2187 (1994)56–67
13. Ardizzone, E., La Cascia, M., Avanzato, A., Bruna, A.: Video indexing using MPEG motion compensation vectors. Proc. IEEE Int. Conf. Multimedia Comp. Syst. (1999) 725–729
14. Astola, J., Haavisto, P., Neuvo, Y.: Vector median filters. Proc. IEEE 78 (1990) 678–689
15. Sammon, Jr. J.W.: A nonlinear mapping for data structure analysis. IEEE Trans. Comput. C-18 (1969) 401–409
16. http://www.research.ibm.com/VideoAnnEx/