Pattern Recognition Letters 30 (2009) 55–65

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

# Simultaneous detection of abrupt cuts and dissolves in videos using support vector machines

Vasileios Chasanis, Aristidis Likas, Nikolaos Galatsanos *

*Department of Computer Science, University of Ioannina, Ioannina 45110, Greece*

ARTICLE INFO

*Article history:*
Received 12 December 2007
Received in revised form 22 July 2008
Available online 6 September 2008

Communicated by Y. Ma

*Keywords:*
Abrupt cut detection
Dissolve detection
Support vector machines

ABSTRACT

Video shot detection is an important contemporary problem since it is the first step towards indexing and content based video retrieval. Traditionally, video shot segmentation approaches rely on thresholding methodologies which are sensitive to the content of the video being processed and do not generalize well the when there is little prior knowledge about the video content. To ameliorate this shortcoming we propose a learning based methodology using a set of features that are specifically designed to capture the differences among hard cuts, gradual transitions and normal sequences of frames at the same time. A support vector machine (SVM) classifier is trained both to locate shot boundaries and characterize transition types. Numerical experiments using a variety of videos demonstrate that our method is capable of accurately discriminating shot transitions in videos with different characteristics.

© 2008 Elsevier B.V. All rights reserved.

## 1. Introduction

In recent years, there has been a significant increase in the availability of high quality digital video as a result of the expansion of broadband services and the availability of large volume digital storage devices. Due to the extended use of videos in several applications such as distance learning, video surveillance, internet–TV and video on demand, as well as the thousands of produced movies and documentaries, a large amount of video information is added to the repositories every year. Consequently, there has been an increase in the need to access this huge amount of information and a great demand for techniques that will provide efficient indexing, browsing and retrieving of video data. The first step towards this direction is to segment the video into smaller "physical" units in order to proceed with indexing and browsing.

The smallest physical segment of a video is the shot and is defined as an unbroken sequence of frames recorded from one camera. After this segmentation has been accomplished, each shot is summarized with one or more frames called key-frames which are selected using spatial and temporal features. Further analysis requires grouping of shots into scenes with similar content. In this paper, we will focus on the first stage of the video segmentation problem which is shot boundary detection. Shot transitions can be classified into two categories. The first one which is the most common is the abrupt cut. An abrupt or hard cut takes place between consecutive frames due to camera switch. In other words, a different or the same camera is used to record a different aspect of the scene. The second category concerns gradual transitions such dissolves, fade-outs followed by fade-ins, wipes and a variety of video effects which stretch over several frames. A dissolve takes place when the initial frames of the second shot are superimposed on the last frames of the first shot. A fade-out is a gradual decrease in the intensity of a frame resulting to a black frame, while fade-in is the opposite i.e., starting from a black image the intensity of the frame gradually increases.

A formal study of the shot boundary detection problem is presented in (Yuan et al., 2007). In (Hanjalic, 2002), the major issues to be considered for the effective solution of the shot boundary detection problem are identified. A comparison of existing methods is presented in (Boreczky and Rowe, 1996; Dailianas et al., 1995; Gargi et al., 2000; Lienhart, 1999). There are several approaches to the shot boundary detection task most of which involve the determination of a predefined or adaptive threshold. A simple way to declare a hard cut is pair-wise pixel comparison (Zhang et al., 1993), where the difference in intensity or color values of corresponding pixels in two successive frames is evaluated and compared against a threshold. This method is very sensitive to object and camera motions, thus many researchers propose the use of a motion independent characteristic, which is the intensity or color, global or local histogram (Nagasaka and Tanaka, 1995; Zhang et al., 1993). The use of second order statistical characteristics of frames is also suggested (Kasturi and Lain, 1991; Zhang et al., 1993). More specifically, the likelihood ratio test is used to compare corresponding blocks of successive frames. Shot transitions are identified when the number of changed blocks is above a predefined threshold. To overcome the difficulties that

* Corresponding author. Tel.: +30 26510 98861; fax: +30 26510 98860.
*E-mail addresses:* vchasani@cs.uoi.gr (V. Chasanis), arly@cs.uoi.gr (A. Likas), galatsanos@cs.uoi.gr (N. Galatsanos).

0167-8655/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2008.08.015

arise from the use of global thresholds several adaptive thresholding methods are reported (Volkmer et al., 2004; Yeo and Liu, 1995; Yusoff et al., 2000). In (Zabih et al., 1999), an algorithm is presented based on the analysis of entering and exiting edges between consecutive frames. This approach works well on abrupt changes, but fails in the detection of gradual changes. In (Cernekova et al., 2006), mutual information and joint-entropy between frames are used for the detection of cuts, fade-ins and fade-outs. An original approach to partitioning of a video into shots based on a foveated representation of the video is proposed in (Boccignone et al., 2005).

A quite interesting approach is presented in (Yuan et al., 2007) where the detection of shot boundaries is based on a graph partitioning problem. More specifically a weighted graph is constructed where each frame is treated as a node and the edges represent the similarity between corresponding frames. Then the min-max criterion is used to partition this graph and the scores for all feasible cuts are calculated. As it concerns the gradual transitions, multi-resolution graphs are constructed which are further partitioned using the same criterion. Finally, support vector machines with active learning are implemented to declare boundaries and non-boundaries. A support vector machine classifier with color and motion features is also employed in (Dalatsi et al., 2001). In that work, the first minutes of a video have been used for training and the rest for testing. In (Feng et al., 2005), the authors propose as features of SVMs, wavelet coefficient vectors within sliding windows.

A variety of methods have been proposed for gradual transitions detection, but still are inadequate to solve this problem due to the complicated nature of such transitions. In (Zhang et al., 1993), a twin-comparison technique is proposed for hard cuts and gradual transitions detection by applying different thresholds based on differences in color histograms between successive frames. In (Ngo et al., 2001), a spatio-temporal approach was presented for the detection of a variety of transitions. There is also research specifically aimed towards the dissolve detection problem. In (Lienhart, 2001), the problem of dissolve detection is treated as a pattern recognition problem. Another direction, which is followed in (Fernando et al., 1999; Hanjalic, 2002; Lelescu and Schonfeld, 2003), is to model the transitions types by presupposing probability distributions for the feature difference metrics and perform a posteriori shot change estimation. It is worth mentioning that the organization of the TREC video shot detection task (NIST) provides a standard performance evaluation and comparison benchmark.

In summary, the main drawback of most previous algorithms is that they are threshold dependent. As a result, if there is no prior knowledge about the visual content of a video that we wish to segment into shots, it is rather difficult to select an appropriate threshold.

In order to overcome this difficulty we propose in this paper a supervised learning methodology for the shot detection problem. In other words, the herein proposed approach does not use thresholds and can actually detect shot boundaries of videos with totally different visual characteristics. Another advantage of the proposed approach, apart from the fact that we do not use any thresholds, is that we can detect hard cuts and gradual transitions at the same time in contrast with existing approaches. For example, in (Dalatsi et al., 2001) the authors propose a support vector machine classifier only for abrupt cut detection. In (Yuan et al., 2007), features for abrupt cuts and dissolves are constructed separately and two different SVM models are trained. In our approach, we define a set of features designed to discriminate hard cuts from gradual transitions. These features are obtained from color histograms and describe the variation between adjacent frames and the contextual information at the same time. Due to the fact that the gradual transitions spread over several frames, the frame-to-frame differences are not sufficient to characterize them. Thus, we also use the differences between non-adjacent frames in the definition of the proposed features.

These features are used as inputs to a support vector machine (SVM) classifier algorithm. A set of nine different videos with over 70 K frames from TV series, documentaries and movies is used to train and test the SVM classifier. The resulting classifier achieves content independent correct detection rates greater than 94%.

The rest of this paper is organized as follows: In Sections 2 and 3, the features proposed in this paper are described. In Section 4, the SVM method employed for this application is briefly presented. In Section 5, we present numerical experiments and compare our method with four existing methods. Finally, in Section 6, we present our conclusions and suggestions for future research.

## 2. Feature selection

### 2.1. Color histogram and $x^2$ value

Color histograms are the most commonly used features to detect shot boundaries. They are robust to object and camera motion, and provide a good trade-off between accuracy of detection and implementation speed. We have chosen to use normalized RGB histograms. So for each frame a normalized histogram is computed, with 256 bins for each one of the RGB component defined as $H^R$, $H^G$ and $H^B$, respectively. These three histograms are concatenated into a 768 dimension vector representing the final histogram of each frame.

$$H = [H_R H_G H_B]. \tag{1}$$

To define whether two shots are separated with an abrupt cut or a gradual transition we have to look for a difference measure between frames. The simplest method for shot detection is to compute the histograms of two adjacent frames calculate the sum of their binwise differences and compare to a threshold.

In our approach, we use a variation of the $x^2$ value to compare the histograms of two frames in order to enhance the difference between the two histograms. Finally the difference between two images $I_i$, $I_j$ based on their color histograms $H_i$, $H_j$ is given from the following equation:

$$d(I_i, I_j) = \frac{1}{3}\left(\sum_{k=1}^{768} \frac{(H_i(k) - H_j(k))^2}{H_i(k) + H_j(k)}\right), \tag{2}$$

where $k$ denotes the bin index. In the literature, two versions of the $x^2$ value have been used (Nagasaka and Tanaka, 1995; Sethi and Patel, 1995). One uses the square of the histogram values in the denominator as normalization and the other the histogram value of the second frame. We decided not to use squares for normalization but the one used herein Eq. (2), because in this manner the produced feature vectors that will be subsequently discussed, were smoother.

### 2.2. Inter-frame distance

The dissimilarity value given in Eq. (2) can be computed for any pair of frames within the video sequence. We compute the value not only between adjacent frames, but also between frames with time distance $l$, where $l$ is called the inter-frame distance as suggested in (Bescós et al., 2005; Hanjalic, 2002). We compute the dissimilarity value $d(I_i, I_{i+1})$ for three values of the inter-frame distance $l$:

1. $l = 1$. This is used to identify hard cuts between two consecutive frames, so the dissimilarity values are computed for $l = 1$.
2. $l = 2$. Due to the fact that during a gradual transition two consecutive frames may be the same or very similar to each other, the dissimilarity value will tend to zero and, as a result, the

sequence of the dissimilarity values could have the form shown in Fig. 1. The computation for $l = 2$ usually results in a smoother curve, which is more useful for our further analysis. A typical example of a sequence of dissimilarity values for $l = 2$ is shown in Fig. 2.

3. $l = 6$. A gradual transition stretches along several frames, while the difference value between consecutive frames is smaller, so we are interested not only in the difference between consecutive frames, but also between frames that are a specific distance apart from each other. As the inter-frame distance increases, the curve becomes smoother as it can be observed in the example of Fig. 3.

Of course the maximum distance between frames for which the inter-frame distance in Eq. (2) is useful is rather small. This distance should be less than the minimum length of all transitions in the video set in order to capture the form of the transition. Thus, the choice of $l = 6$ was made due to the fact that most of the gradual transitions in our set of videos have length between 7 and 40 frames.

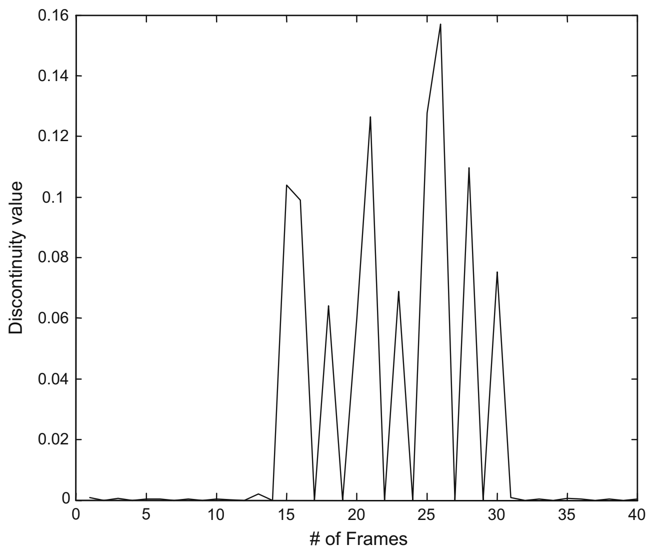

**Fig. 1.** Dissimilarity pattern for $l = 1$.

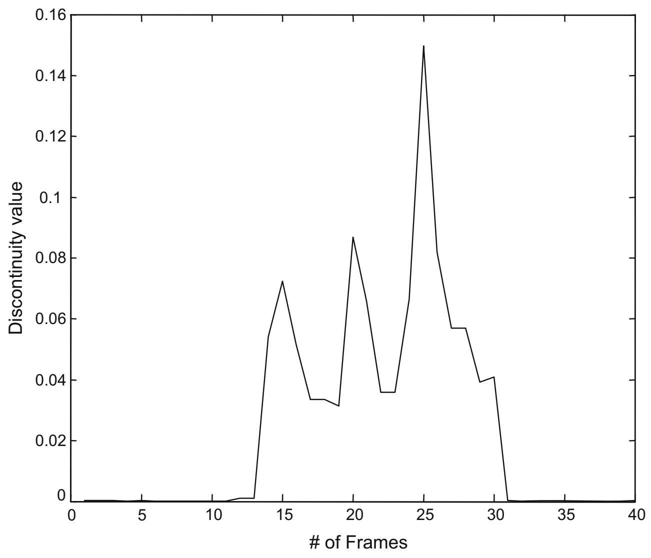

**Fig. 2.** Dissimilarity pattern for $l = 2$.

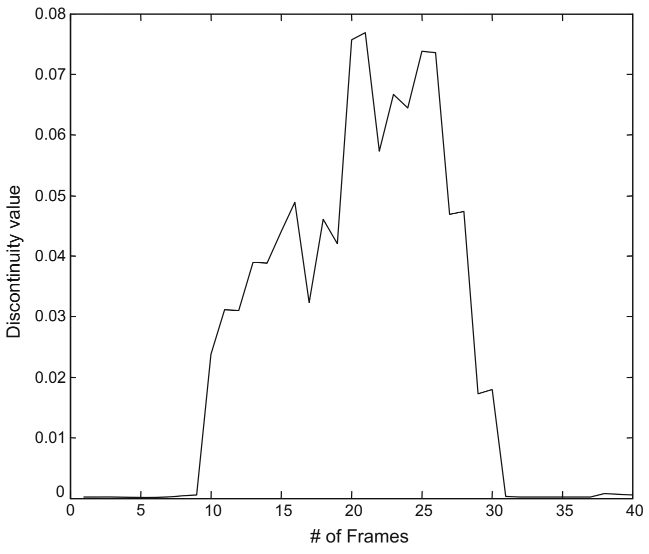

**Fig. 3.** Dissimilarity pattern for $l = 6$.

## 3. Feature vector selection for shot boundary classification

The dissimilarity values defined in Section 2 are not going to be compared with any threshold, but they will be used to form feature vectors based on which an SVM classifier will be constructed.

### 3.1. Definition of feature vectors

The feature vectors selected are the normalized dissimilarity values calculated in a temporal window centered at the frame of interest. More specifically, the dissimilarity values that are computed in section 2 form three vectors, one for each one of the three inter-frame distances $l$.

$$\begin{aligned} D^{l=1} &= [d(I_1, I_2), \ldots, d(I_i, I_{i+1}), \ldots, d(I_{N-1}, I_N)], \\ D^{l=2} &= [d(I_1, I_3), \ldots, d(I_i, I_{i+2}), \ldots, d(I_{N-2}, I_N)], \\ D^{l=6} &= [d(I_1, I_7), \ldots, d(I_i, I_{i+6}), \ldots, d(I_{N-6}, I_N)]. \end{aligned} \tag{3}$$

Moreover for each frame, we define a window of length w that is centered at this frame and contains the dissimilarity values. As a result for the $i$th frame the following three vectors are composed:

$$\begin{aligned} W^{l=1}(i, 1:w) &= [D^{l=1}(i - w/2), \ldots, D^{l=1}(i), \ldots, D^{l=1}(i + w/2 - 1)], \\ W^{l=2}(i, 1:w) &= [D^{l=2}(i - w/2), \ldots, D^{l=2}(i), \ldots, D^{l=2}(i + w/2 - 1)], \\ W^{l=6}(i, 1:w) &= [D^{l=6}(i - w/2), \ldots, D^{l=6}(i), \ldots, D^{l=6}(i + w/2 - 1)]. \end{aligned} \tag{4}$$

To obtain the final features we normalize the dissimilarity values in Eq. (4) by dividing each dissimilarity value by the sum of the values in the window. This provides the normalized "magnitude" independent features.

$$\widetilde{W}^{l=k}(i,j) = \frac{W^{l=k}(i,j)}{\sum_{j=1}^{w} W^{l=k}(i,j)}, \quad k = 1, 2, 6 \tag{5}$$

The size of the window used is $w = 40$. In our experiments we also considered windows of length 50 and 60 in order to capture longer transitions. The 120-long vector resulting from the concatenation of the normalized dissimilarities for the three windows given by

$$F(i) = [\widetilde{W}^{l=1}(i), \quad \widetilde{W}^{l=2}(i), \quad \widetilde{W}^{l=6}(i)] \tag{6}$$

is the feature vector corresponding to frame $i$.

In what follows, we show examples of the feature vectors for a hard cut, two dissolves and a "normal" sequence of frames in Figs. 4–7. By observing these features, it is clear that the shape of the $l = 1$ normalized dissimilarity vectors for normal sequences and dissolves can be of similar shape. However, the inclusion of the $l = 2$ and $l = 6$ dissimilarity vectors discriminates the two categories.

## 4. Support vector machine classifier

After the feature definition, an appropriate classifier has to be used in order to categorize each frame in three categories: normal sequences, abrupt cuts and gradual transitions. For this purpose we selected the support vector machine (SVM) classifier (Cortes and Vapnik, 1995) that provides state-of-the-art performance and scales well with the dimension of the feature vector which is relatively large (equal to 120) in our problem.

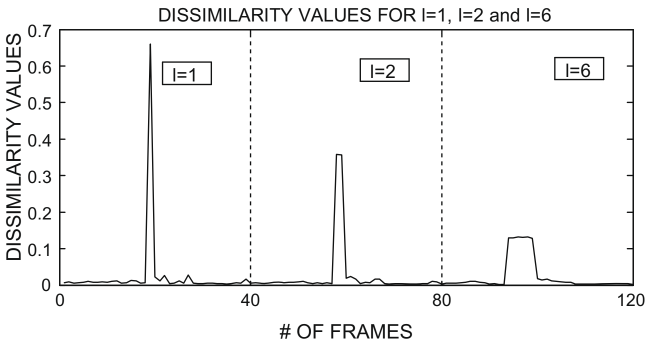

**Fig. 4.** Feature vector for a hard cut.

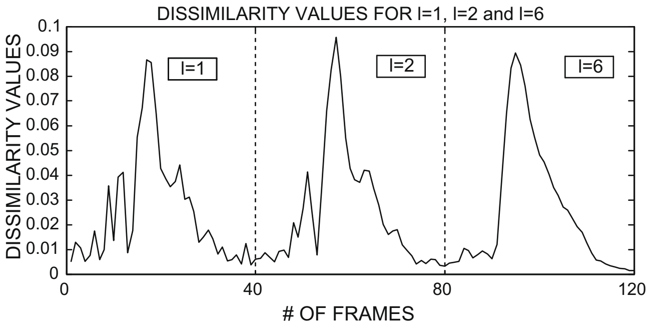

**Fig. 5.** Feature vector for the first dissolve example.

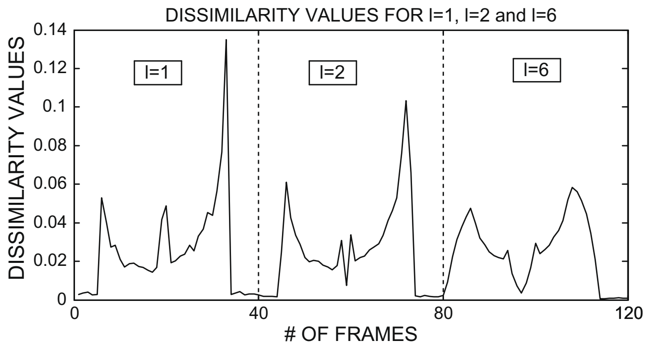

**Fig. 6.** Feature vector for the second dissolve example.

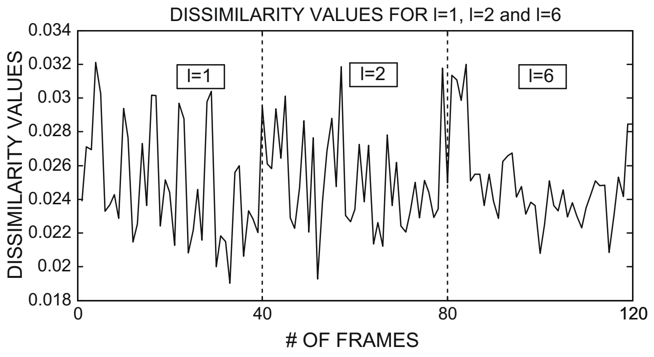

**Fig. 7.** Feature vector for a normal sequence.

The classical SVM classifier finds an optimal hyperplane which separates data points of two classes. More specifically, suppose we are given a training set of l vectors $x_i \in R^n$, $i = l$, $l$ belonging to one of two classes, and a vector $y \in R^l$ with $y_i \in \{1, -1\}$ denoting the class of vector $x_i$. We also assume a mapping function $\varphi(x)$, that maps each training vector to a higher dimensional space, and the corresponding kernel function $K(x,y)$ (Eq. (9)). Then, the SVM classifier (Cortes and Vapnik, 1995) is obtained by solving the following primal problem:

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\ \text{subject to} \quad & y_i(w^T \varphi(x_i) + b) \geqslant 1 - \xi_i, \\ & \xi_i \geqslant 0, \quad i = 1, \ldots, l. \end{aligned} \tag{7}$$

The decision function is

$$\operatorname{sgn}\left( \sum_{i=1}^{l} w_i K(x_i, x) + b \right), \quad \text{where } K(x_i, x_j) = \varphi^T(x_i)\varphi(x_j). \tag{8}$$

A notable characteristic of SVMs is that after training, usually most of the training patterns $x_i$ have $w_i = 0$ in Eq. (8), in other words they do not contribute to the decision function. Those $x_i$ for which $w_i \neq 0$, are retained in the SVM model and called support vectors (SVs). In our approach the commonly used radial basis function (RBF) kernel is employed.

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2), \tag{9}$$

where $\gamma$ denotes the width of the kernel. It must be noted that in order to obtain an efficient SVM classifier the parameters $C$ (Eq. (7)) and $\gamma$ (Eq. (9)) must be carefully selected, usually through cross-validation.

The above algorithm is suitable for binary classification. In our application, we have a three-class problem, thus we used the "one-against-one" approach (Knerr et al., 1990) in which for a $k$-class problem, $k(k - 1)/2$ binary classifiers are constructed and each one is trained to discriminate data from two classes. More specifically, if we assume that class label 0 corresponds to normal sequences, class label 1 to dissolves and class label 2 to hard cuts, three binary classifiers discriminating between pairs of classes (0,1), (1,2) and (0,2) are constructed. The final classification is based on a voting strategy where the decision of each binary classifier is considered as a vote for its proposed class and the class with the maximum number of votes is selected. In the case of a tie the class with the smallest index is selected. This tie braking strategy is well-justified since in our case class 0 corresponds to normals which is the most probable outcome.

## 5. Experiments

In this section, we present numerical experiments of the proposed approach and compare our method with three other methods.

### 5.1. Data

The video sequences used for our data set were taken from TV series, documentaries and educational films. Nine videos (70000 frames) were used; containing 355 hard cuts and 142 dissolves (Table 1), manually annotated.

### 5.2. Performance criteria

To evaluate the performance of our method we used the following criteria (Bimbo, 1999):

$$\mathrm{RECALL} = \frac{N_c}{N_c + N_m},$$
$$\mathrm{PRECISION} = \frac{N_c}{N_c + N_f}, \qquad (10)$$
$$F_1 = \frac{2 * \mathrm{RECALL} * \mathrm{PRECISION}}{\mathrm{RECALL} + \mathrm{PRECISION}},$$

where $N_c$ stands for the number of correct detected shot boundaries, $N_m$ for the number of missed ones and $N_f$ the number of false detections. During our experiments, we calculate the F1 value for the cuts ($F_{1C}$) and the dissolves ($F_{1D}$) separately. Then the final performance measure is given from the following equation:

$$F_1 = \frac{a}{a+b}F_{1C} + \frac{b}{a+b}F_{1D}, \qquad (11)$$

where $a$ is the number of true hard cuts and $b$ the number of true dissolves.

### 5.3. Results and comparison

In our experiments, eight videos are used for training and the ninth for testing, therefore, nine "rounds" of testing were conducted. In order to obtain good values of the parameters $\gamma$ and $C$ (in terms of providing high $F_1$ values), in each "round" we applied three-fold cross-validation using the eight videos of the corresponding training set. A difficulty of the problem under consideration is the generation of an imbalanced training set that contains few positives examples and a huge number of negative ones. In (Schohn and Cohn, 2000), an active learning procedure is proposed to reduce the training time: based on the assumption that the support vectors determine the decision boundary in Eq. (8), they suggest removing the training examples that are far from the SVM's decision hyperplane. In (Yuan et al., 2007), the authors identify the positive examples while reducing the number of negative ones by applying a predefined threshold on their constructed features. In our approach we sample negative examples uniformly, thus we reduce their number to 3% of the total number of examples. More specifically, in our training set there are 440 positive examples (transitions) and 2200 negative examples (no transitions) on average. Finally each model of the training procedure generated on average 1276 support vectors for normal transitions, 101 support vectors for gradual transitions and 152 support vectors for abrupt transitions. The number of examples and support vectors (on average) of the support vector machines classification are summarized in Table 2.

We also tested our method by using larger windows of width $w = 50$ and $w = 60$. In what follows in Tables 3–5, we provide the classification results using different selections of window lengths and inter-frame distances. We notice that the performance improves as the size of the window increases. False boundaries are reduced since larger windows contain more information. The use of larger windows also helps the detection of dissolves that last longer.

In order to reduce the size of our feature vector, we have also consider as feature vectors used to train the SVM classifier, those obtained from the concatenation of features extracted for $l = 2$ and $l = 6$, only. It can be observed (Tables 6–8) that even with the shorter feature vector the proposed algorithm gives very good

**Table 1**
Characteristics of videos used in the experiments

| Video ID | Frames | Cuts | Dissolves | Genre |
|---|---|---|---|---|
| T1 | 6318 | 36 | 23 | Comedy |
| T2 | 9466 | 28 | 16 | Action |
| T3 | 1180 | 4 | 6 | Drama |
| T4 | 1535 | 14 | 8 | Educational |
| T5 | 17982 | 146 | 7 | Action |
| T6 | 1665 | 1 | 19 | Comedy |
| T7 | 14993 | 105 | 11 | Drama |
| T8 | 9840 | 12 | 41 | Documentary |
| T9 | 6355 | 9 | 11 | Documentary |
| Total | 69334 | 355 | 142 | – |

**Table 2**
Training examples and support vectors

| | Positive examples | Negative examples | Support vectors |
|---|---|---|---|
| Cuts | 315 | – | 152 |
| Dissolves | 126 | – | 101 |
| Normal | – | 2200 | 1276 |

**Table 3**
Performance results for $w = 40$, $l = 1$, $l = 2$ and $l = 6$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 351 | 4 | 9 | 98.87 | 97.50 | 98.18 |
| Dissolves | 127 | 15 | 33 | 89.44 | 79.38 | 84.11 |
| Average | – | – | – | 96.18 | 92.32 | 94.21 |

**Table 4**
Performance results for $w = 50$, $l = 1$, $l = 2$ and $l = 6$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 352 | 3 | 8 | 99.15 | 97.78 | 98.46 |
| Dissolves | 130 | 12 | 25 | 91.55 | 83.87 | 87.54 |
| Average | – | – | – | 96.98 | 93.80 | 95.37 |

**Table 5**
Performance results for $w = 60$, $l = 1$, $l = 2$ and $l = 6$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 353 | 2 | 4 | 99.44 | 98.88 | 99.16 |
| Dissolves | 127 | 15 | 25 | 89.44 | 83.55 | 86.39 |
| Average | – | – | – | 96.58 | 94.50 | 95.53 |

**Table 6**
Performance results for $w = 40$, $l = 2$ and $l = 6$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 351 | 4 | 9 | 98.87 | 97.50 | 98.18 |
| Dissolves | 127 | 16 | 30 | 88.73 | 80.77 | 84.56 |
| Average | – | – | – | 95.98 | 92.72 | 94.32 |

**Table 7**
Performance results for $w = 50$, $l = 2$ and $l = 6$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 350 | 5 | 5 | 98.59 | 97.49 | 98.04 |
| Dissolves | 129 | 13 | 21 | 90.85 | 86.00 | 88.36 |
| Average | – | – | – | 96.38 | 94.21 | 95.28 |

**Table 8**
Performance results for $w = 60$, $l = 2$ and $l = 6$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 351 | 4 | 5 | 98.87 | 98.60 | 98.73 |
| Dissolves | 128 | 14 | 26 | 90.14 | 83.12 | 86.49 |
| Average | – | – | – | 96.38 | 94.17 | 95.26 |

**Table 9**
Performance results for $w = 40$, $l = 1$, $l = 2$, $l = 6$ and constant $(C, \gamma)$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 353 | 2 | 9 | 99.44 | 96.98 | 98.19 |
| Dissolves | 128 | 14 | 23 | 90.14 | 84.77 | 87.37 |
| Average | – | – | – | 96.78 | 93.49 | 95.11 |

results that are only slightly inferior to the ones obtained by the longer feature vector.

In order to test the importance of selecting the best values for $(C, \gamma)$, in another experiment we used the SVM classifier with constant values pair $(C, \gamma) = (6, 8)$ for all "rounds" of testing. The obtained results (Tables 9–14) indicate that even without the "optimal selection" of $(C, \gamma)$ the performance of the SVM classifier

**Table 10**
Performance results for $w = 50$, $l = 1$, $l = 2$, $l = 6$ and constant $(C, \gamma)$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 353 | 2 | 7 | 99.44 | 98.06 | 98.74 |
| Dissolves | 128 | 14 | 31 | 90.14 | 80.50 | 85.05 |
| Average | – | – | – | 96.78 | 93.04 | 94.87 |

**Table 11**
Performance results for $w = 60$, $l = 1$, $l = 2$, $l = 6$ and constant $(C, \gamma)$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 353 | 2 | 5 | 99.44 | 98.60 | 99.02 |
| Dissolves | 128 | 14 | 23 | 90.14 | 84.77 | 87.37 |
| Average | – | – | – | 96.78 | 94.65 | 95.70 |

**Table 12**
Performance results for $w = 40$, $l = 2$, $l = 6$ and constant $(C, \gamma)$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 352 | 3 | 13 | 99.15 | 96.44 | 97.78 |
| Dissolves | 127 | 16 | 22 | 88.73 | 85.14 | 86.90 |
| Average | – | – | – | 96.18 | 93.21 | 94.67 |

**Table 13**
Performance results for $w = 50$, $l = 2$, $l = 6$ and constant $(C, \gamma)$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 352 | 3 | 10 | 99.15 | 96.44 | 98.19 |
| Dissolves | 129 | 13 | 20 | 90.85 | 86.58 | 88.66 |
| Average | – | – | – | 96.78 | 94.19 | 95.47 |

**Table 14**
Performance results for $w = 60$, $l = 2$, $l = 6$ and constant $(C, \gamma)$

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 351 | 4 | 7 | 98.87 | 98.04 | 98.46 |
| Dissolves | 127 | 15 | 22 | 89.44 | 85.23 | 87.29 |
| Average | – | – | – | 96.18 | 94.38 | 95.27 |

**Table 15**
Performance results for $w = 40$, $l = 1$, $l = 2$, $l = 6$,HSV histograms, $x^2$ distance

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 353 | 2 | 4 | 99.44 | 98.89 | 99.16 |
| Dissolves | 125 | 17 | 29 | 88.03 | 81.17 | 84.46 |
| Average | – | – | – | 95.19 | 92.84 | 93.97 |

**Table 16**
Performance results for $w = 40$, $l = 1$, $l = 2$, $l = 6$ HSV histograms, Kullback–Liebler distance

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 352 | 3 | 5 | 99.15 | 98.60 | 98.92 |
| Dissolves | 122 | 20 | 31 | 85.92 | 79.74 | 82.73 |
| Average | – | – | – | 94.38 | 92.23 | 93.91 |

remains very good. The choice of $w = 50$ for the window size seems to provide best performance. In addition, it can deal with transitions that spread over many frames.

In another additional experiment we carried out, an HSV normalized histogram is computed for each frame, with eight bins for hue and four bins for each of saturation and value, resulting to $8 \times 4 \times 4$ bins. In Tables 15 and 16, we provide the classification results using the $x^2$ value defined in Eq. (2) and the Kullback–Liebler distance between two histograms, respectively. It can be observed that the method is not sensitive to the choice of the color space and the distance measure between the color.

In order to gain more intuition into the SVM classifier for this problem, in Figs. 9–11, we provide correctly detected feature vectors from dissolves, hard cuts and normal sequences of frames. From these figures, it is clear that the selected features of the three classes of frame sequences exhibit distinguishable characteristics. More specifically, for hard cuts the feature vector contains three "impulses" with decaying height and increasing width. For dissolves it contains three replicas of a pattern that resembles to a rectangle from which a sinusoidal lobe has been subtracted. Furthermore, these patterns become smoother as we move from left to right. Finally, for "normal" sequences of frames the pattern resembles to "white noise" superimposed on a constant DC value.

In Figs. 12–14, we provide a portion of SVs for all the cases for the same "round". These examples are training patterns for which $w_i \neq 0$ in Eq. (8) and are retained in the SVM model. We immedi-

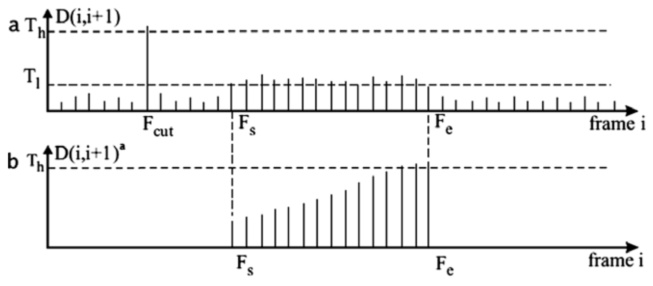

**Fig. 8.** Twin-comparison algorithm (Zhang et al., 1993).

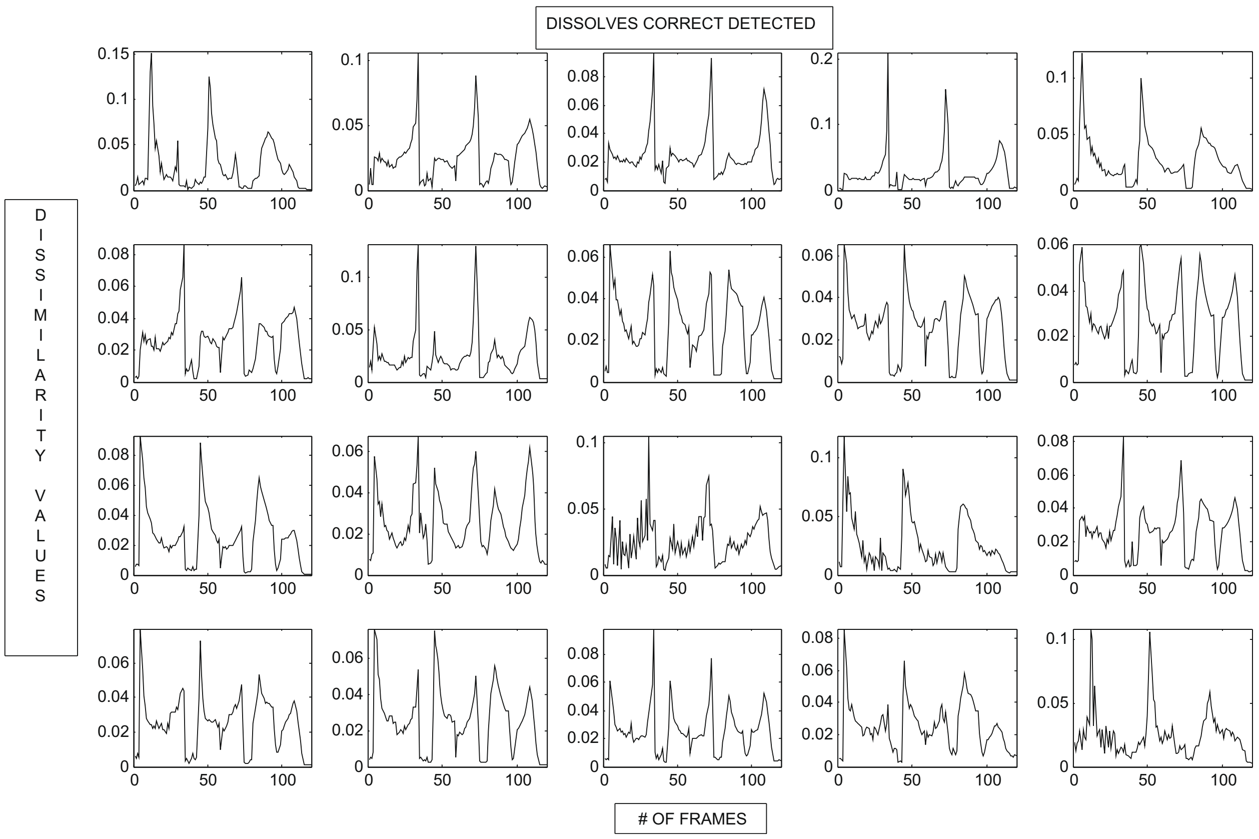

**Fig. 9.** Correctly detected dissolve patterns.

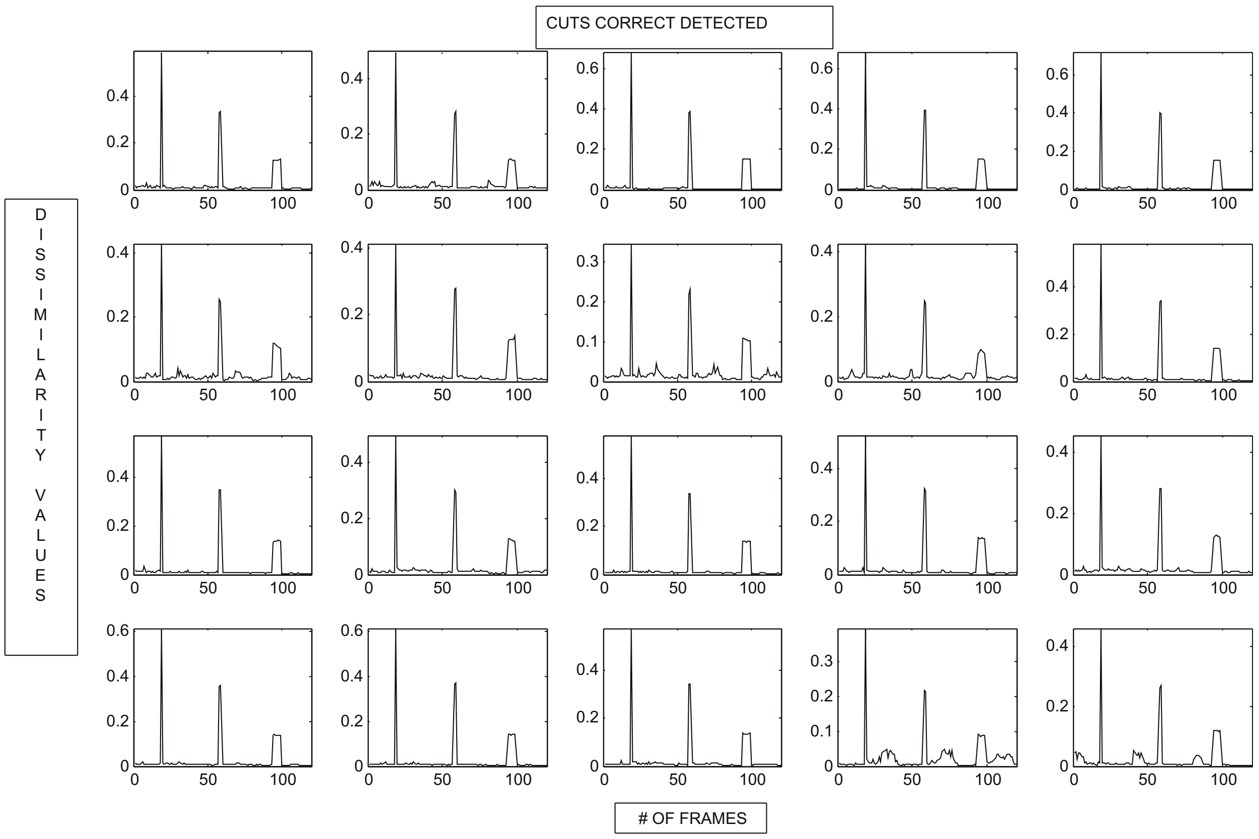

**Fig. 10.** Correctly detected hard cut patterns.

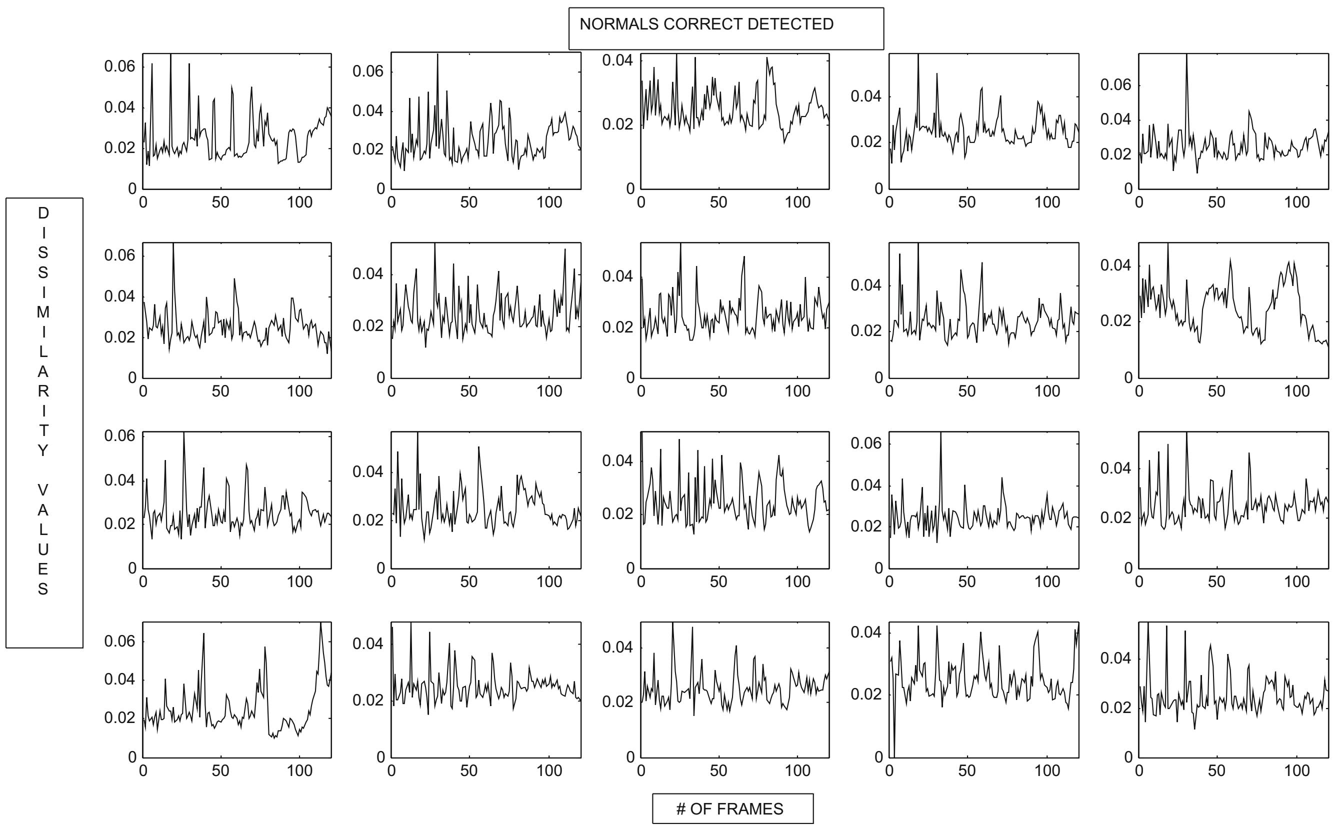

**Fig. 11.** Correctly detected patterns of normal sequences.

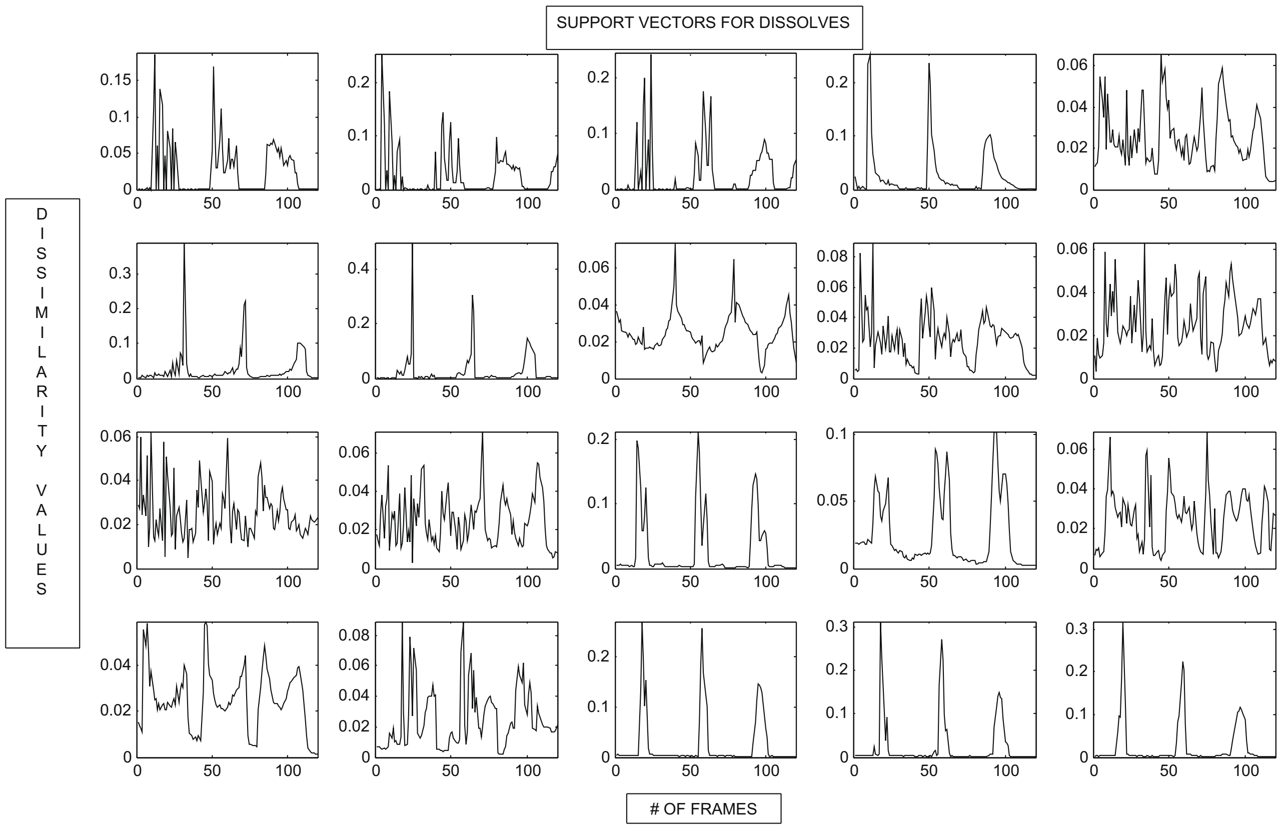

**Fig. 12.** Support vectors for dissolves.

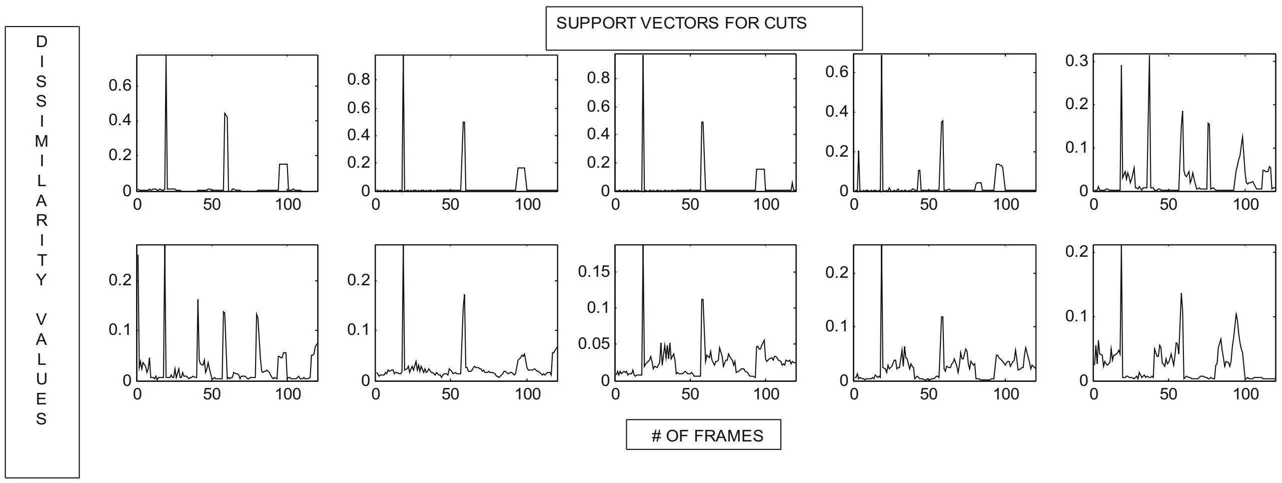

**Fig. 13.** Support vectors for hard cuts.

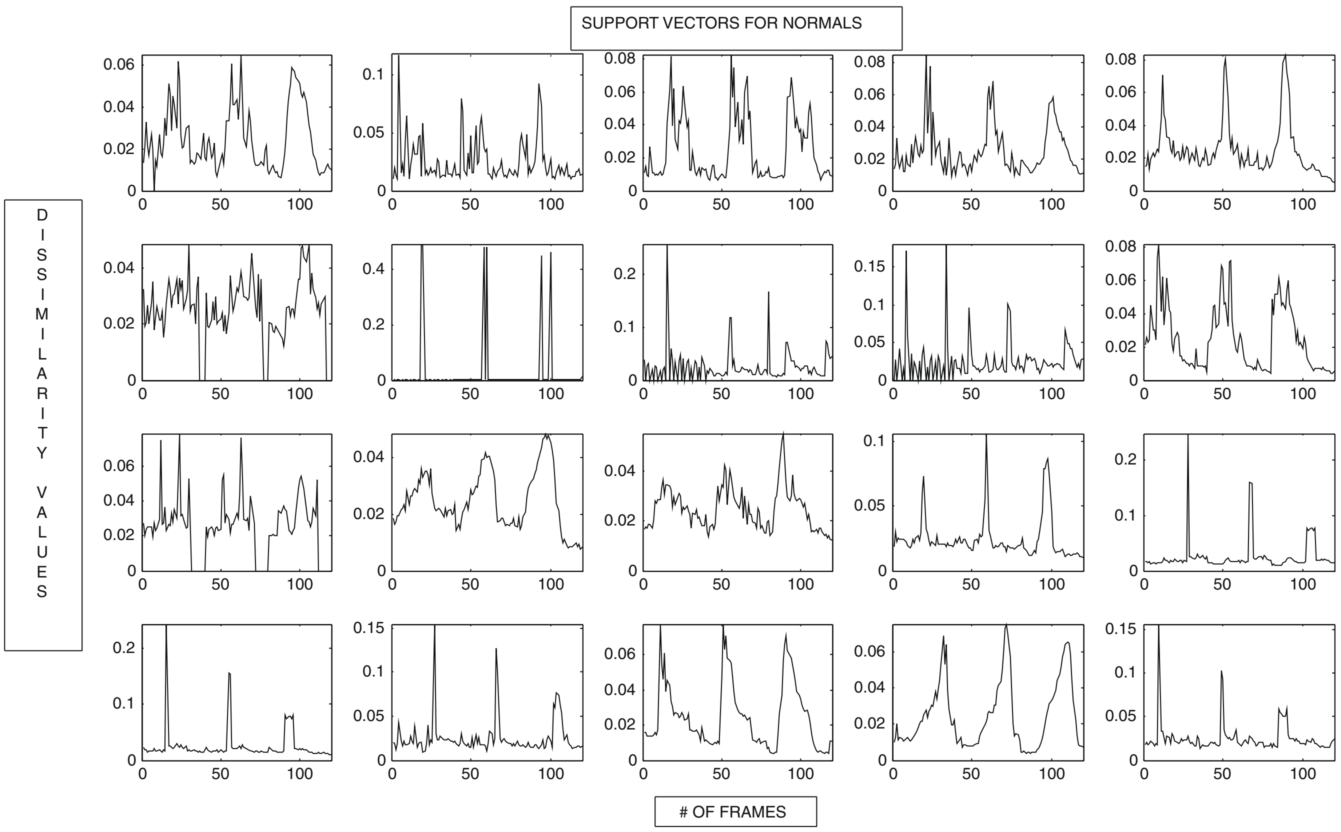

**Fig. 14.** Support vectors for normal sequences.

ately notice, as expected, that the SVs are the ‘‘borderline examples" for all categories. The hardest cases to separate are normal from dissolves, since more SVs of the normal class have characteristics of correctly classified dissolves and fewer have characteristics of hard cuts. Similarly, more SVs for dissolves have characteristics of ‘‘normal" than hard cuts. Furthermore, the SVs for hard cuts are much fewer than their ‘‘normal" and dissolves counterparts. Also one cannot make an assessment whether most of them have characteristics of the normal or dissolve class.

To demonstrate the effectiveness of our algorithm and its advantage over threshold depended methods, we implemented three methods that use thresholds in different ways. More specifically, we implemented pair-wise comparison of successive frames (Zhang et al., 1993), likelihood ratio test (Kasturi and Lain, 1991; Zhang et al., 1993) and the twin-comparison method (Zhang et al., 1993). The first two methods can only detect cuts, while the third can identify both abrupt and gradual transitions. We also compare our method with the method proposed in (Feng et al., 2005).

The pair-wise comparison method (Zhang et al., 1993) compares corresponding pixels of successive frames to determine how many pixels have changed. More specifically we consider that a pixel changes if the difference of its corresponding pixel in the following frame is over a predefined threshold:

**Table 17**
Performance results for Pair-wise comparison method (Zhang et al., 1993)

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 302 | 53 | 54 | 85.07 | 84.83 | 84.95 |
| Dissolves | – | – | – | – | – | – |
| Average | – | – | – | – | – | – |

**Table 18**
Performance results for Likelihood ratio method (Zhang et al., 1993)

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 335 | 5320 | 54 | 94.37 | 86.12 | 90.05 |
| Dissolves | – | – | – | – | – | – |
| Average | – | – | – | – | – | – |

**Table 19**
Performance results for Twin-comparison method (Zhang et al., 1993)

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 317 | 38 | 41 | 89.30 | 88.05 | 88.92 |
| Dissolves | 100 | 42 | 51 | 70.42 | 64.94 | 67.57 |
| Average | – | – | – | 83.90 | 81.80 | 82.82 |

**Table 20**
Performance results for method in (Feng et al., 2005)

| Transition type | $N_c$ | $N_m$ | $N_f$ | RECALL (%) | PRECISION (%) | $F_1$ (%) |
|---|---|---|---|---|---|---|
| Cuts | 348 | 7 | 32 | 97.18 | 91.57 | 94.29 |
| Dissolves | 106 | 36 | 24 | 74.64 | 81.53 | 77.93 |
| Average | – | – | – | 89.77 | 87.78 | 88.67 |

$$DP_i(x,y) = \begin{cases} 1, & \text{if } |P_i(x,y) - P_{i+1}(x,y)| > Th \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

where $P_i(x,y)$ is the intensity of pixel with coordinates $(x,y)$ in frame $i$. A shot boundary is declared if the number of the pixels that have changed is over another predefined threshold. However the appropriate selection of such a threshold is a tedious task. Firstly, this threshold is different for videos that belong to different genres and secondly, even in the same video differences between frames may vary due to different states of illumination and content. Thus, it is difficult to define a global threshold. In (Yusoff et al., 2000), the authors propose the use of a sliding window over the differences. A shot boundary is detected if two conditions are fulfilled: the middle sample of the window (a) is the maximum in the window and (b) is greater than

$$\max(\mu_{\text{left}} + a\sigma_{\text{left}}, \mu_{\text{right}} + a\sigma_{\text{right}}), \tag{13}$$

where $\mu_{\text{left}}$, $\mu_{\text{right}}$ and $\sigma_{\text{left}}$, $\sigma_{\text{right}}$ are the means and standard deviations of the samples, left and right of the middle sample of the window, respectively. The length of the window and the parameter $\alpha$ are set to 21 and 5, respectively. In Table 17, we provide the performance of the pair-wise comparison method.

The second method we implemented uses the likelihood ratio (Kasturi and Lain, 1991; Zhang et al., 1993) as the metric to compute frame differences. More specifically, this metric compares the second order statistics of corresponding regions. Each frame is divided into blocks, which represent the regions, and the likelihood ratio between two consecutive frames $i$, $i+1$ for a specific block $k$ is given from the following equation:

$$\lambda(i,i+1)_\kappa = \frac{\left[\left(\frac{\sigma_i+\sigma_{i+1}}{2}\right) + \left(\frac{\mu_i-\mu_{i+1}}{2}\right)^2\right]^2}{\sigma_i \times \sigma_{i+1}}, \tag{14}$$

where $\mu_i$, $\mu_{i+1}$ and $\sigma_i$, $\sigma_{i+1}$ are the means and standard deviations of block $k$ of frames $i$, $i+1$, respectively.

Then, the likelihood ratio between two consecutive frames $i$, $i+1$ is as follows:

$$L(i,i+1) = \frac{\sum_{k=1}^{K} \lambda(i,i+1)_k}{K}, \tag{15}$$

where $K$ is the number of blocks of the frame. A shot boundary is detected when the likelihood ratio between two frames exceeds a predefined threshold. To improve the performance of the specific method, we do not use a global threshold, but we select the threshold via cross-validation, using the "leave-one-out" method. To identify the threshold and test it on a video of our dataset, we choose the threshold that achieves the best performance over the rest eight videos of our dataset. In Table 18 we provide the performance of the likelihood ratio method.

Finally, the third method we implemented was the twin-comparison algorithm which uses two thresholds for the detection of abrupt and gradual transitions. Each frame is represented with a histogram and the differences between histograms of consecutive frames are calculated. Histograms and their differences are computed using Eqs. (1) and (2). If the difference between two successive frames exceeds a high threshold $T_{\text{high}}$, a cut is detected. A low threshold $T_{\text{low}}$ is used for the detection of gradual transitions. If the difference is above $T_{\text{low}}$ then this frame is characterized as a potential start $F_s$ of the gradual transition. Then, $F_s$ is compared with subsequent frames providing the accumulated differences metric. The end frame $F_e$ of the transition is detected if two conditions are satisfied: (1) the consecutive difference falls below threshold $T_{\text{low}}$ and (2) the accumulated difference exceeds threshold $T_{\text{high}}$. If consecutive difference falls below $T_{\text{low}}$ before the accumulated difference

**Table 21**
Comparative results using RECALL, PRECISION and $F_1$ measures

| Method | Transition type | | | | | |
|---|---|---|---|---|---|---|
| | Cuts | | | Dissolves | | |
| | RECALL (%) | PRECISION (%) | $F_1$ (%) | RECALL (%) | PRECISION (%) | $F_1$ (%) |
| $w = 40$, $l = 1$, $l = 2$ and $l = 6$. | 98.87 | 97.50 | 98.18 | 89.44 | 79.38 | 84.11 |
| $w = 40$, $l = 2$ and $l = 6$. | 98.87 | 97.50 | 98.18 | 88.73 | 80.77 | 84.56 |
| $w = 40$, $l = 1$, $l = 2$ and $l = 6$ and constant $(C,\gamma)$ | 99.44 | 96.98 | 98.19 | 90.14 | 84.77 | 87.37 |
| $w = 40$, $l = 2$ and $l = 6$ and constant $(C,\gamma)$ | 99.15 | 96.44 | 97.78 | 88.73 | 85.14 | 86.90 |
| $w = 40$, $l = 1$, $l = 2$ and $l = 6$ (HSV,$x^2$) | 99.44 | 98.89 | 99.16 | 88.03 | 81.17 | 84.46 |
| $W = 40$, $l = 1$, $l = 2$ and $l = 6$ (HSV,KL) | 99.15 | 98.60 | 98.92 | 85.92 | 79.74 | 82.73 |
| Pair-wise comparison (Zhang et al., 1993) | 85.07 | 84.83 | 84.95 | – | – | – |
| Likelihood ratio (Zhang et al., 1993) | 94.37 | 86.12 | 90.05 | – | – | – |
| Twin-comparison (Zhang et al., 1993) | 89.30 | 88.05 | 88.92 | 70.42 | 64.94 | 67.57 |
| (Feng et al., 2005) | 97.18 | 91.57 | 94.29 | 74.64 | 81.53 | 77.93 |

exceeds $T_{\text{high}}$, then the potential start frame is discarded. In Fig. 8, we illustrate the twin-comparison algorithm.

To compute the two thresholds we follow the method proposed in (Kobla et al., 1999). The threshold $T_{\text{low}}$ is calculated from the following equation:

$$T_{\text{low}} = \mu + a\sigma, \tag{16}$$

where $\mu$ and $\sigma$ are the mean and standard deviation of histogram differences respectively. The value of parameter $\alpha$ is set to 5. To calculate $T_{\text{high}}$ we compute the histogram of the differences values. Threshold $T_{\text{high}}$ is assigned to the index value that corresponds to half of the peak value on the right slope of the peak value of the histogram. $T_{\text{high}}$ must be higher than mean value. In Table 19, we provide the performance of the twin-comparison method.

In (Feng et al., 2005), Blocked Color Histogram is incorporated as feature vector and the temporal multi-resolution characteristics of shot presented by the wavelet transition coefficients are selected as video frame series patterns for a SVM classifier. In Table 20, we present the classification results for this method.

The obtained results indicate that our algorithm outperforms the other three threshold dependent methods and the method proposed in (Feng et al., 2005). In Table 21, we provide the recall, precision and $F_1$ values for our algorithm and the four methods under consideration. For our algorithm we present the results using $w = 40$, for best values $(C, \gamma)$ and constant values pair $(C, \gamma) = (6, 8)$, using all features ($l = 1$, $l = 2$ and $l = 6$) and less features ($l = 2$ and $l = 6$). We also present the results using HSV histograms and two different distance metrics between histograms: (a) $x^2$ value and (b) Kullback–Liebler. The thresholds used in the three threshold dependent methods were calculated in different ways. We used adaptive thresholds in pair-wise comparison algorithm, cross-validation in likelihood ratio method and finally global adaptive threshold in the twin-comparison method. Especially for the dissolve detection our algorithm, provides far better results than the twin-comparison algorithm.

In summary, the proposed system is capable of identifying where a shot boundary occurs and whether the transition is abrupt or gradual. The main advantage of the method is that it can be trained using different types of video and then it can be used to locate shot boundaries in other videos without using any thresholds.

## 6. Conclusions

In this paper, we have proposed a method for shot boundary detection and discrimination between a hard cut and a gradual transition. Features that describe the variation between adjacent frames and the contextual information were derived from color histograms using a temporal window. These feature vectors become inputs to a SVM classifier which categorizes transitions of the video sequence into normal transitions, hard cuts and gradual transitions. This categorization provides an effective segmentation of any video into shots and thus is a valuable aid to further analysis of the video for indexing and browsing. The main advantage is that throughout the whole procedure, no use of any thresholds is made. As a future work, we will try to improve the performance of the method by extracting other types of features from the video sequence.

## Acknowledgements

This research project (PENED) is co-financed by EU – European Social Fund (75%) and the Greek Ministry of Development – GSRT (25%).

## References

Bescós, J., Cisneros, G., Martínez, J.M., Menéndez, J.M., Cabrera, J., 2005. A unified model for techniques on video-shot transition detection. IEEE Trans. Multimedia 7 (2), 293–307.

Bimbo, A.D., 1999. Visual Information Retrieval. Morgan Kaufmann Publishers Inc., San Francisco, California.

Boccignone, G., Chianese, A., Moscato, V., Picariello, A., 2005. Foveated shot detection for video segmentation. IEEE Trans. Circuits Systems Video Technol. 15 (3), 365–377.

Boreczky, J.S., Rowe, L.A., 1996. Comparison of video shot boundary detection techniques. In: Proc. SPIE Storage and Retrieval for Image and Video Databases, vol. 2664, pp. 170–179.

Cernekova, Z., Pitas, I., Nikou, C., 2006. Information theory-based shot cut/fade detection and video summarization. IEEE Trans. Circuits Systems Video Technol. 16 (1), 82–91.

Cortes, C., Vapnik, V., 1995. Support-vector network. Mach. Learn. 20 (3), 273–297.

Dailianas, A., Allen, R.B., England, P., 1995. Comparison of automatic video segmentation algorithms. In: Proc. SPIE Photonics East'95: Integration Issues in Large Commercial Media Delivery Systems, vol. 2615, pp. 2–16.

Dalatsi, C., Krinidis, S., Tsekeridou, S., Pitas, I., 2001. Use of support vector machines based on color and motion features for shot boundary detection. In: Internat. Symp. on Telecommunications.

Feng, H., Fang, W., Liu, S., Fang, Y., 2005. A new general framework for shot boundary detection and key-frame extraction. In: Proc. 7th ACM SIGMM Internat. Workshop Multimedia Inf. Retrieval, pp. 121–126.

Fernando, W.A.C., Canagarajah, C.N., Bull, D.R., 1999. Fade and dissolve detection in uncompressed and compressed video sequences. In: Proc. IEEE Internat. Conf. Image Processing, vol. 3, pp. 299–303.

Gargi, U., Kasturi, R., Strayer, S.H., 2000. Performance characterization of video-shot-detection methods. IEEE Trans. Circuits Systems Video Technol. 10 (1), 1–13.

Hanjalic, A., 2002. Shot-boundary detection: Unraveled and resolved? IEEE Trans. Circuits Systems Video Technol. 12 (2), 90–105.

Kasturi, R., Lain, R., 1991. Dynamic vision. In: Kasturi, R., Lain, R. (Eds.), Computer Vision: Principles. IEEE Computer Society Press, Washington, DC, pp. 469–480.

Knerr, S., Personnaz, L., Dreyfus, G., 1990. Single-layer learning revisited: A stepwise procedure for building and training a neural network. In: Fogelman, J. (Ed.), Neurocomputing Algorithms, Architectures and Applications. Springer-Verlag.

Kobla, V., DeMenthon, D., Doermann, D., 1999. Special effect edit detection using VideoTrails: A comparison with existing techniques. In: Proc. SPIE Conf. on Storage and Retrieval for Image and Video Databases VII, vol. 3656, pp. 302–316.

Lelescu, D., Schonfeld, D., 2003. Statistical sequential analysis for real-time video scene change detection on compressed multimedia bitstream. IEEE Trans. Multimedia 5 (1), 106–117.

Lienhart, R., 1999. Comparison of automatic shot boundary detection algorithms. In: Proc. SPIE Storage and Retrieval for Image and Video Databases VII, San Jose, CA, vol. 3656, pp. 290–301.

Lienhart, R., 2001. Reliable dissolve detection. In: Proc. SPIE Storage and Retrieval for Media Databases, vol. 4315, pp. 219–230.

Nagasaka, A., Tanaka, Y., 1995. Automatic video indexing and full-video search for object appearances. In: Knuth, E., Wegner, L.M. (Eds.), Visual Database Systems II. Elsevier, pp. 113–127.

Ngo, C.W., Pong, T.C., Chin, R.T., 2001. Video partitioning by temporal slice coherence. IEEE Trans. Circuits Systems Video Technol. 11 (8), 941–953.

NIST, Homepage of Trecvid Evaluation. [Online] <http://www-nlpir.nist.gov/projects/trecvid/>.

Schohn, G., Cohn, D., 2000. Less is more: Active learning with support vector machines. In: Proc. 17th Internat. Conf. Machine Learning, pp. 839–836.

Sethi, I.K., Patel, N., 1995. A statistical approach to scene change detection. In: Proc. SPIE on Storage and Retrieval for Image and Video Databases III, vol. 2420, pp. 329–339.

Volkmer, T., Tahaghoghi, S.M.M., Williams, H., 2004. RMIT University at Trecvid 2004. In: Proc. TRECVID 2004 Workshop.

Yeo, B.-L., Liu, B., 1995. Rapid scene analysis on compressed video. IEEE Trans. Circuits Systems Video Technol. 5 (6), 544–553.

Yuan, J., Wang, H., Xiao, L., Zheng, W., Li, J., Lin, F., Zhang, B., 2007. A formal study of shot boundary detection. IEEE Trans. Circuits Systems Video Technol. 17 (2), 168–186.

Yusoff, Y., Chrismas, W., Kittler, J., 2000. Video shot cut detection using adaptive threshold. In: 11th British Machine Vision Conf. (BMVC'00), Bristol, UK.

Zabih, R., Miller, J., Mai, K., 1999. Feature-Based Algorithms for Detecting and Classifying Production Effects. Multimedia Systems 7 (2), 119–128.

Zhang, H.J., Kankanhalli, A., Smoliar, S.W., 1993. Automatic partitioning of full-motion video. Multimedia Systems 1 (1), 10–28.