2009

Adaptive Edge-Oriented Shot Boundary Detection

Don Adjeroh

M. C. Lee

N. Banda

Uma Kandaswamy

Hindawi Publishing Corporation
EURASIP Journal on Image and Video Processing
Volume 2009, Article ID 859371, 13 pages
doi:10.1155/2009/859371

Research Article

Adaptive Edge-Oriented Shot Boundary Detection

Don Adjeroh,[1] M. C. Lee,[2] N. Banda,[1] and Uma Kandaswamy[1]

[1] *Video and Image Processing Laboratory, Lane Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, WV 26505, USA*
[2] *Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin NT, Hong Kong*

Correspondence should be addressed to Don Adjeroh, don@csee.wvu.edu

Received 20 August 2008; Revised 11 April 2009; Accepted 18 May 2009

Recommended by Alberto Del Bimbo

We study the problem of video shot boundary detection using an adaptive edge-oriented framework. Our approach is distinct in its use of multiple multilevel features in the required processing. Adaptation is provided by a careful analysis of these multilevel features, based on shot variability. We consider three levels of adaptation: at the feature extraction stage using locally-adaptive edge maps, at the video sequence level, and at the individual shot level. We show how to provide adaptive parameters for the multilevel edge-based approach, and how to determine adaptive thresholds for the shot boundaries based on the characteristics of the particular shot being indexed. The result is a fast adaptive scheme that provides a slightly better performance in terms of robustness, and a five fold efficiency improvement in shot characterization and classification. The reported work has applications beyond direct video indexing, and could be used in real-time applications, such as in dynamic monitoring and modeling of video data traffic in multimedia communications, and in real-time video surveillance. Experimental results are included.

1. Introduction

Video shot boundary detection (also called video partitioning or video segmentation) is a fundamental step in video indexing and retrieval, and in general video data management. The general objectives are to segment a given video sequence into its constituent shots, and to identify and classify the different shot transitions in the sequence. Different algorithms have been proposed, for instance, based on simple color histograms [1, 2], pixel color differences [3], color ratio histograms [4], edges [5], and motion [6–8]. In this work, we study the problem of video partitioning using an edge-based approach. Unlike ordinary colors, edges are largely invariant under local illumination changes and are much less affected by the possible motion in the video. To ensure robustness, we use both edge-based and color-based features under a multilevel decomposition framework. With the multiple decompositions, we can avoid the time-consuming problem of motion estimation by a careful choice of the decomposition level to operate at. Improvements in video partitioning have been recorded by performing a dynamic classification of the shots as the video is being analyzed, and then adaptively choosing the shot partitioning parameters based on the predicted class of the shot [9]. Automatic shot classification can also serve as an important step in approaching the elusive problem of capturing semantics or meaning in the video sequence (see, e.g., [14]).

We note that the problem of video shot partitioning (or segmentation) is not only relevant to video indexing and video data management. (See [11–14] for discussion on video query, browsing, and video object management). It is also an important issue in other areas of video communication, such as video compression and video traffic modeling [15]. In particular, for problems such as video traffic characterization and modeling, shot-level adaptation becomes mandatory, if the network is to dynamically allocate limited network resources in response to changing video data traffic.

In this work, we introduce adaptation at different stages in the video analysis process—both at the feature extraction stage and at the later stage of frame difference comparison. We propose a new method for fast shot characterization and classification required for such adaptation, using a new set of edge-based features. We introduce a method for automated threshold selection for adaptive scene partitioning schemes. In the next section, we describe recent reported work

that is closely related to our approach. Section 3 presents the multilevel edge-response vectors, the basic features we propose for video partitioning. Shot characterization and adaptation in video partitioning in the context of the edge-based features is described in Section 4. Section 5 presents results on real video sequences. We conclude the paper in Section 6.

2. Related Work

The first step in content-based video data management is shot boundary detection. Simply put, it is the process of partitioning a given video sequence into its constituent shots. The purpose is to determine the beginning (and/or end) of different types of transitions that may occur in the sequence. The problem of video partitioning is compounded by the various changes that might occur in the video, (say due to illumination, motion and/or occlusion), and by the different types of shot transitions (such as fades and dissolves). The inherent variability in video shot characteristics, even for shots from the same sequence introduces further complication. The partitioning algorithm depends on the specific features used, and the similarity evaluation functions adopted. Earlier methods for video shot partitioning are described in [2–4, 16–18]. See [19–21] for a survey.

Most approaches to video partitioning make use of the color (or gray level) information in the video. The limitations of color in video partitioning are the problems of illumination variation and motion-induced false alarm. Edge based methods have thus been proposed to reduce the problem of invariance due to illumination and motion. Zabih et al. [5] made explicit use of edges in video indexing, and showed how the *exiting* and *entering* edges can be used to classify different types of shot breaks. Related methods that exploit edge information for shot detection directly in the compressed domain were proposed in [4, 18, 22, 23]. In [4] color ratio features were proposed as an alternative to color histograms, and were used to identify different types of shot changes without decompressing the video. The motivation was that color ratios capture the color boundaries or color edges in the frames. In [18, 23] methods were proposed to extract edges directly from the DCT coefficients, which can then be used for video partitioning. In [22], Abdel-Mottaleb and Krishnamachari described the edge-based information used as part of the descriptions in MPEG-7. Edge descriptors were given as 4-bin histograms, where each bin is for one of the four directions: vertical, horizontal, left-diagonal, and right-diagonal. Other related compressed domain methods are reported in [9, 13, 16].

More recent approaches to the video partitioning problem have been proposed in [9, 24–27]. Li and Lai [28] described methods for video partitioning using motion estimation, where the motion vectors are extracted using optical flow computations. To account for potential changes in the lighting conditions, the optical flow computations included a parameter to model the local illumination changes during motion estimation. Cooper et al. [25, 29] partitioning techniques that exploit possible self-similarity in the video, by classifying temporal patterns in the video sequence using kernel-based correlation. Li and Lee [26] studied video partitioning, with special emphasis on gradual transitions. Yoo et al. [27] studied both gradual and abrupt shot transitions, and proposed methods based on localized edge blocks. For abrupt shot boundaries, they proposed a correlation-based method, based on which localized edge gradients are then used for detecting gradual shot transitions.

The need for adaptation in the video indexing process was first identified in [30] (see also [9]), where they showed that video shots vary considerably from one shot to the other, even for shots that come from the same video sequence. They thus suggested that the results of an indexing scheme could be improved by treating different shots differently, for instance, by use of a different set of analysis parameters. Since then, there has been an increasing attention to the problem. In [31], detailed experiments were carried out using television news video. It was concluded that the selection of similarity thresholds was a major problem, and hence there is a need for adaptive thresholds to capture the different characteristics of broadcast news video. Vansconcelos and Lippman [10, 32] considered the duration of video shots, and showed that the shot duration can be used to predict the position of a new shot partition, and that the short duration depends critically on the video content. They used a statistical model of the shot duration to propose shot break thresholds. By classifying video shots in terms of the shot complexity and shot duration, and then performing indexing adaptively based on the video shot classes, it was shown in [9, 30] that, indeed, adaptation could be used to improve both the precision and recall simultaneously, without introducing an intolerable amount of extra computation. Dawood and Ghanbari [15] used a similar classification to model MPEG video traffic. The problem of video indexing and retrieval is very closely related to that of image indexing. Surveys on video (and/or image) indexing and retrieval can be found in [19, 21, 33, 34]. Video partitioning or segmentation has been reviewed in [20].

In this paper, we study the use of both color and edges in adaptive video partitioning. Our approach is distinct in its use of multilevel edge-based features in video partitioning, and in the provision of adaptation by a careful analysis of these multilevel features, based on the notion of shot variability. Adaptation is provided at three levels—at the feature extraction stage for the locally-adaptive edge maps, at the video sequence level, and at the individual shot level.

3. Multilevel Edge-Response Vectors

In our approach, we place emphasis on the structural information in the video, as these are generally invariant under various changes in the video, such as illumination changes, translation, and partial occlusion. Thus, in addition to the intensity values, we also make use of the edges in computing the features to be used. In particular, we use multi scale edges, since these can more easily capture localized structures in the video frames.

3.1. Multilevel Image Decomposition. Let $I(x, y)$ be an $M_1 \times M_2$ image, with $x = 1, 2, \ldots, M_1$; $y = 1, 2, \ldots, M_2$. Given $I(x, y)$, we decompose it into different blocks. For each block, we consider its content at different scales, and compute edge-based features at each of these scales. We then use the features to compare two adjacent frames in the video sequence. For simplicity in the discussion, we assume images are square, that is, $M_1 = M_2 = M$. We also assume $M = 2^p$, for some integer p. The ideas can easily be extended for the general rectangular image.

Let b be the number of blocks at a given decomposition level. We choose k, the level of decomposition, such that $b = 1, 4, 16, \ldots, 2^{2k}$, $k = 0, 1, 2, \ldots, \log_2 M$. Let s be the scale, $s = 0, 1, 2, \ldots, S$. Then, given the original image, $I(x, y)$, we can select relevant areas of the image at different scales, s. Let $I^s(i, j)$ be the sub image part selected at scale s, where $i, j = 1, 2, \ldots, (M/2)(1 + (1/2^s))$. At the lowest scale ($s = 0$), we will have the entire image, viz:

$$I(x, y) = I^0(x, y) = I^0(x_0 + x, y_0 + y), \tag{1}$$

where $x = 1, 2, \ldots, M$; $y = 1, 2, \ldots, M$, and x_0, y_0 are starting positions (typically, $x_0 = y_0 = 0$). Let x_0^s, y_0^s be the corresponding starting positions at scale s (these are with respect to x_0, y_0 in $I(x, y)$). Then we have:

$$I^s(x', y') = I^0(x_0^s + i, y_0^s + j), \tag{2}$$

where,

$$\begin{aligned} x' &= x_0^s + i; \quad i = 1, 2, \ldots, \frac{M}{2}\left(1 + \frac{1}{2^s}\right), \\ y' &= y_0^s + j; \quad j = 1, 2, \ldots, \frac{M}{2}\left(1 + \frac{1}{2^s}\right). \end{aligned} \tag{3}$$

At a given scale s, the starting positions are computed as

$$\begin{aligned} x_0^s &= x_0 + \frac{M}{2}\left(1 - \frac{1}{2^s}\right), \\ y_0^s &= y_0 + \frac{M}{2}\left(1 - \frac{1}{2^s}\right). \end{aligned} \tag{4}$$

The size of the image block selected at scale s will therefore be $m_s \times m_s$, where $m_s = (M/2)(1 - 1/2^s)$. For a given decomposition level, we consider each of the $m_s \times m_s$-sized blocks and compute the required image features. If we fix the number of scales to 1 (i.e., $s = 0$) at each level k, (i.e., at each level, we select all the image positions within the block to compute the feature), then the multi scale scheme defaults to a simple multilevel representation of the image. Thus, using $s = 0$ with L maximum number of levels (i.e., $k = 0, 1, 2, \ldots, L - 1$), we will have an N-dimensional feature vector, where

$$N = \sum_{k=0}^{L-1} 2^{2k} = \frac{1}{3}\left(4^L - 1\right). \tag{5}$$

Clearly, the number of features grows quickly with increasing L (e.g., at $L = 4$, $N = 85$; at $L = 5$, $N = 341$). For the multi scale representation, we have more than one scale per level. With S as the number of scales per level (i.e., $s = 0, 1, 2, \ldots, S - 1$), we will have $S \cdot N$ feature values for each particular feature. In the following, we will assume a single scale (i.e., $S = 1$, and $s = 0$). Figure 1 shows schematic diagram of an image at different levels of decomposition, and a tree representation of the individual blocks from each level.

3.2. Edge-Oriented Features. To reduce the possible effect of noise in the video, we first apply Gaussian smoothening on the input image before computing the edge-based features. Given the image $I(x, y)$, the edge gradients are defined as $G_x(x, y) = \partial I/\partial x$, $G_y(x, y) = \partial I/\partial y$. The gradients are obtained using appropriate edge kernels:

$$\begin{aligned} G_x(x, y) &= I(x, y) * H_x(x, y), \\ G_y(x, y) &= I(x, y) * H_y(x, y), \end{aligned} \tag{6}$$

where H_x and H_y are the horizontal and vertical gradient masks, respectively, and $*$ represents convolution. The gradient amplitude is given by

$$G_A(x, y) = \sqrt{G_x^2(x, y) + G_y^2(x, y)}, \tag{7}$$

which can be approximated using the simple absolute sum:

$$G_A(x, y) = |G_x(x, y)| + \left|G_y(x, y)\right|. \tag{8}$$

The phase angle is given by

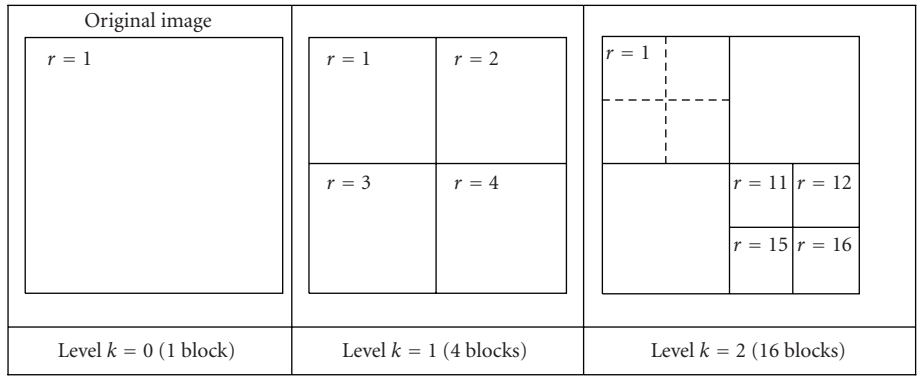
$$G_\phi(x, y) = \tan^{-1}\left(\frac{G_y(x, y)}{G_x(x, y)}\right). \tag{9}$$

These will be calculated once for each frame, but will be used at different levels of decomposition.

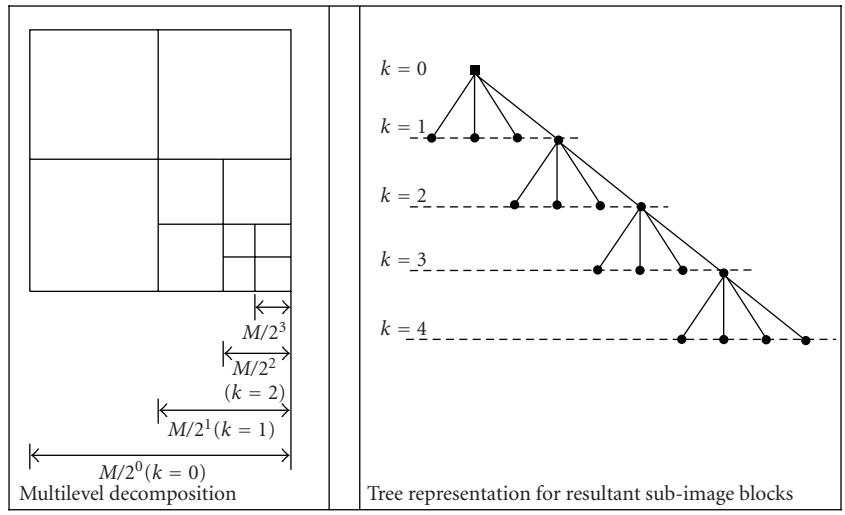
3.2.1. Locally Adaptive-Edge Map. The major motivation for a multilevel approach is that certain variations in an image, such as those due to edges are local in nature, and hence will be better captured by use of local (rather than global) information. For video in particular, this becomes very important. Although some variations (such as panning, tilting, and illumination) in the video could be global with respect to a particular frame, object motion and some other camera operations (such as zooming) are more easily modeled as a local phenomenon. (Note, although zooming could also be global over the video frame, the direction of the motion vectors will vary from one area of the image to the other). We capture global information by using information from the lower levels of decomposition (smaller values of k). With higher levels, we can obtain information about more localized structures in the frame. Such localized structures could be treated differently for improved performance.

We use locally adaptive thresholds to define the edge map at different decomposition levels. Each block is considered using it's own local threshold. For a given block r, at the kth level ($r = 1, 2, 3, \ldots, 2^{2k}$), we define the edge map as follows:

$$E_r^k(x, y) = \begin{cases} 1, & G_{A,r}^k(x, y) > \tau_r^k, \\ 0, & \text{otherwise}, \end{cases} \tag{10}$$



(a)



(b)

Figure 1: Multilevel decomposition (a) an image at three levels of decomposition; (b) tree representation of the decomposition.

where $G^k_{A,r}$ is the corresponding gradient response in the rth block at level k, and τ^k_r is a local threshold. We can choose the threshold simply as

$$\tau^k_r = \frac{\alpha}{\left(m^k_r\right)^2} \sum_{x=1}^{m^k_r} \sum_{y=1}^{m^k_r} G^k_{A,r}(x,y), \tag{11}$$

where $\left(m^k_r\right)^2$ is the size of the rth block at level k, α is a constant. While the above approach to local thresholds is simple and conceptual, it however considers each block independent of the other blocks in the frame. It might be advantageous to consider the local threshold with respect to the global image variations [35]. At a given k, we can write $m^k_r = m_k,\ \forall r$ since the block size would all be the same for any block, r.

Define the overall global image threshold as

$$\tau_g = \tau^o_1 = \frac{\alpha}{M^2} \sum_{x=1}^{M} \sum_{y=1}^{M} G_A(x,y), \tag{12}$$

where α is a constant (which can be determined empirically).

The local threshold for a given block r at each level k is then given by

$$\tau^k_r = \begin{cases} \left(1 - \dfrac{\sigma^k_{e,r}}{\mu^k_{e,r}}\right).\tau_g, & \mu^k_{e,r} < \mu^0_{e,1}, \\ \left(\dfrac{\sigma^k_{e,r}}{\mu^k_{e,r}}\right).\tau_g, & \mu^k_{e,r} \geq \mu^0_{e,1},\ \sigma^k_{e,r} > \mu^k_{e,r}, \\ \left(1 + \dfrac{\sigma^k_{e,r}}{\mu^k_{e,r}}\right).\tau_g, & \mu^k_{e,r} \geq \mu^0_{e,1},\ \sigma^k_{e,r} \leq \mu^k_{e,r}, \end{cases} \tag{13}$$

where $\mu^k_{e,r}$, $\sigma^k_{e,r}$ are, respectively, the edge response mean and standard deviation for block r, at level k.

3.2.2. Edge-Based Features. At a given level k, and for each given block r, we compute the following features.

(i) *Color*, $\mu^k_{c,r}$, $\sigma^k_{c,r}$: color mean and standard deviation using $I(x,y)$.

(ii) *Edge response*, $\mu_{e,r}^k$, $\sigma_{e,r}^k$: edge response mean and standard deviation using $G_A(x, y)$:

$$\mu_r^k = \frac{\alpha}{\left(m_r^k\right)^2} \sum_{x=1}^{m_r^k} \sum_{y=1}^{m_r^k} G_{A,r}^k(x, y);$$
$$\sigma_{e,r}^k = \left[\frac{\alpha}{\left(m_r^k\right)^2 - 1} \sum_{x=1}^{m_r^k} \sum_{y=1}^{m_r^k} \left(G_{A,r}^k(x, y) - \mu_r^k \right)^2 \right]^{1/2} \tag{14}$$

(iii) *Phase angle*, $\mu_{\varphi,r}^k$, $\sigma_{\varphi,r}^k$: mean phase angle and standard deviation using $G_\phi(x, y)$.

(iv) *Edge length*, $\mu_{\lambda,r}^k$: edge length using the edge map, $E_r^k(x, y)$, where $\mu_{\lambda,r}^k = \sum_{x,y} E_r^k(x, y)$.

(v) *Edge response at the edge points*, $\mu_{p,r}^k$, $\sigma_{p,r}^k$: edge response mean and standard deviation computed only at the edge points, as defined by the edge maps.

The edge points are the pixel positions that lie on the edges—as determined by the thresholds above. We call the combined features including the color features *multilevel edge-response vectors* (MERVs).

3.3. Similarity Evaluation Using MERVs. Having extracted the features, the next question is how to find appropriate metrics to compare two video frames using these features. Given two images $I_1(x, y)$ and $I_2(x, y)$, we can compute the distance between them using the general Minkowski distance, or some other metrics. In the following we use the simple city-block distance.

For the edge length, there will be no standard deviation, hence the distance will be

$$d_\lambda(I_1, I_2) = \sum_k \sum_r \left| \mu_{\lambda,r}^{k,1} - \mu_{\lambda,r}^{k,2} \right|. \tag{15}$$

For the other features, we need to consider both the mean and the standard deviation. For example, for the edge response feature, we will have

$$d_e(I_1, I_2) = \sum_k \sum_r \left(\left| \mu_{e,r}^{k,1} - \mu_{e,r}^{k,2} \right| + \left| \sigma_{e,r}^{k,1} - \sigma_{e,r}^{k,2} \right| \right). \tag{16}$$

Similarly, we obtain the corresponding distance $d_c(\cdot)$, $d_\varphi(\cdot)$, and $d_p(\cdot)$ for color, phase angle, and edge response at edge points, respectively. The overall distance between the two images is then determined as a simple weighted-average of the individual distances from the different features:

$$D(I_1, I_2) = w_c d_c(I_1, I_2) + w_e d_e(I_1, I_2) + w_p d_p(I_1, I_2) + w_\varphi d_\varphi(I_1, I_2) + w_\lambda d_\lambda(I_1, I_2), \tag{17}$$

where $w_c + w_e + w_p + w_\varphi + w_\lambda = 1$.

The parameters w_c, w_e, w_φ, w_λ, w_p are respective weights for features based on the color, edge response, phase angle, edge length, and edge response at edge points. By simply varying the weights, we can completely ignore the contribution of any particular feature. For the weights above to be meaningful however, we need to be sure that the range of values for the individual distances will be similar. Thus, we either have to normalize all the features to the same range of values, or we can compute the distance such that the overall distance from each feature is normalized. We take the later approach, and perform normalization at the time of distance computation, based on the model-data feature pairs

$$d_\lambda(I_1, I_2) = \sum_k \sum_r \left(\frac{\left| \mu_{\lambda,r}^{k,1} - \mu_{\lambda,r}^{k,2} \right|}{1 + \mu_{\lambda,r}^{k,1} + \mu_{\lambda,r}^{k,2}} \right),$$
$$d_e(I_1, I_2) = \sum_k \sum_r \left(w_\mu \frac{\left| \mu_{e,r}^{k,1} - \mu_{e,r}^{k,2} \right|}{1 + \mu_{e,r}^{k,1} + \mu_{e,r}^{k,2}} + w_\sigma \frac{\left| \sigma_{e,r}^{k,1} - \sigma_{e,r}^{k,2} \right|}{1 + \sigma_{e,r}^{k,1} + \sigma_{e,r}^{k,2}} \right), \tag{18}$$

where again w_μ and w_σ are weights, with $w_\mu + w_\sigma = 1$. The normalized distances can then be used with the weights in (17) to obtain the overall distance between the frames.

Another important issue is the effect of each individual block in the overall difference. Let $w_{f,r}^k$ be the weight of feature f from the rth block at level k. That is, $f \in \{c, e, \varphi, \lambda, p\}$, where $c, e, \varphi, \lambda, p$ denote respective features based on color, edge response, phase angle, edge length, and edge response at edge points. A simple approach is to adopt a method whereby for a chosen feature f, the contribution from every block at each level is given an equal weight. Effectively, $w_{f,r}^k = 1/N$, $\forall r, k$, where $N = \sum_{k=0}^{L} N_k = \sum_{k=0}^{L} 2^{2k}$. N_k is simply the number of blocks at the kth level. This makes the features from the lower levels of the decomposition to become more important. As the number of decomposition levels L increases, the lower-level features will dominate in the computation of the overall difference, and hence this will become very sensitive to small spatial differences in the frames. This will hence be more susceptible to noise and minute motion variations in the video. For shot classification however, this can be beneficial, since the domination of global movement or features in the video can be avoided.

A better approach could be to divide up the contribution to the overall difference amongst the k levels. The blocks that make up the kth level will then share the contribution allocated to that level. A simple way to do this will be by using an equal distribution of the contribution to all the levels:

$$w_{f,r}^k = \frac{1}{L} \left(\frac{1}{2^{2k}} \right). \tag{19}$$

In all cases, we must have $\sum_r \sum_k w_{f,r}^k = 1$. The effect of the weights using the two cases considered above can be appreciated from Table 1.

Considering the weights at each level, the distance between adjacent frames can be computed:

$$d_e(I_1, I_2) = \sum_k \sum_r \left(\left| \mu_{e,r}^{k,1} - \mu_{e,r}^{k,2} \right| + \left| \sigma_{e,r}^{k,1} - \sigma_{e,r}^{k,2} \right| \right) w_{e,r}^k, \tag{20}$$

or in weighted and normalized form:

$$d_e(I_1, I_2) = \sum_k \sum_r \left(w_\mu \frac{\left| \mu_{e,r}^{k,1} - \mu_{e,r}^{k,2} \right|}{1 + \mu_{e,r}^{k,1} + \mu_{e,r}^{k,2}} + w_\sigma \frac{\left| \sigma_{e,r}^{k,1} - \sigma_{e,r}^{k,2} \right|}{1 + \sigma_{e,r}^{k,1} + \sigma_{e,r}^{k,2}} \right) . w_{e,r}^k. \tag{21}$$

Table 1: Weights for two choices for the contribution from each decomposition level ($L = 4$).

		SCHEME 1		SCHEME 2	
k	N_k	One block $w_{f,r}^k$	All blocks from level k: $\sum_r w_{f,r}^k$	One block $w_{f,r}^k$	All blocks from level k: $\sum_r w_{f,r}^k$
0	1	$\frac{1}{85}$	$\frac{1}{85}$	$\frac{1}{4}$	$\frac{1}{4}$
1	4	$\frac{1}{85}$	$\frac{4}{85}$	$\frac{1}{16}$	$\frac{1}{4}$
2	16	$\frac{1}{85}$	$\frac{16}{85}$	$\frac{1}{64}$	$\frac{1}{4}$
3	64	$\frac{1}{85}$	$\frac{64}{85}$	$\frac{1}{256}$	$\frac{1}{4}$

4. Adaptive Video Partitioning

When the distance $D(\cdot)$ is computed for a series of adjacent video frames, the result will be a sequence of frame differences, *FD-sequence* for short. The actual video partitioning is performed by a further analysis of the FD-sequence. Let $D_i = D(f_i, f_{i+1})$ be the difference between two adjacent frames, f_i and f_{i+1}. The FD-sequence is defined as $\mathrm{FD} = D_1, D_2, \ldots, D_{n-1}$, where n is the number of frames in the video. The FD-sequence is usually characterized by significant peaks at frame positions where a shot change has occurred. With the FD-sequence, the video partitioning problem then becomes that of determining appropriate thresholds to isolate these "significant peaks" from other peaks that might occur in the sequence. The shot threshold is defined as $\tau_s = \tau \cdot \max_i\{D(f_i, f_{i+1})\}$. We declare a shot partition at frame t whenever the distance exceeds the threshold: that is, whenever $D_t = \mathrm{FD}(t) > \tau_S$.

4.1. Adaptation at the Video Sequence Level. The description above assumes that video sequences are homogeneous, and hence can all be considered using the same set of parameters. However, video sequences vary considerably from one sequence to the other. First we consider adapting the video analysis algorithm based on the entire video sequence. That is, for each video sequence, we determine the set of analysis parameters that will produce the best results. This set of parameters is then used to analyze all the frames or shots in the video sequence.

Given the weights on the multilevel features (see (17)), we can parameterize the analysis algorithm in terms of these weights, $w = (w_c, w_e, w_p, w_\varphi, w_\lambda)$ and the threshold, τ. For adaptation at the sequence level, rather than considering all the features for the distance calculation, we consider only the features that are relevant to the video being analyzed. Thus, based on the particular video, we can determine the best (w, τ) pair for segmenting the video.

To check the effect of the weights and the thresholds on different video sequences, we used a combination of the weights at different thresholds. Based on empirical analysis, we chose 32 combinations of the weights (Table 2) and 9 thresholds (Table 3).

We observed that different videos may require different contributions from each feature (i.e., different weights, w) for best results. Also, at a given w, different thresholds could produce different results. (See Table 6, Section 5). Similarly, for a given video sequence, various sets of weights can produce the same (best) results, but at different thresholds. Conceptually, adaptation at the sequence level should be simple. But there are several problems. First, at the sequence level, the video is still being considered at a very coarse granularity. Video shots are known to vary greatly, even for shots in the same video. Hence, different shots in the same video sequence could be very different in content. More importantly, automated mapping of the (w, τ) pair for each given video is a major problem, requiring a two-pass approach. This makes sequence-level adaptation unsuitable for real-time applications, or for network applications, where dynamic modeling of video data traffic is required.

4.2. Shot-Level Adaptation. The above problems can be addressed by considering the individual shots that make up the video. In [9], shots were characterized based on the activity and motion in the shots, and the respective shot duration. Using the characterization, video shots were grouped into nine classes, based on which video partitioning was performed by adaptively choosing different thresholds for each shot class. In the current work, we take a different approach for the problems of video characterization and classification.

4.2.1. Estimating Video Shot Complexity. To make the thresholds sensitive to the different shot classes, we need some methods to make such thresholds locally adaptive. The overall video shot complexity depends on the activity and the motion, while the shot class depends on both the complexity and the duration of the shot. The shot duration has a strong correlation with the amount of motion in the video. The length of the shot is typically inversely proportional to the amount of motion in the video [9]. We can determine the temporal duration as we analyze the shot. We could also determine the motion complexity by computing the motion vectors using motion estimation techniques [36]. However, motion estimation is very computationally intensive.

Table 2: Parameter sets used in video analysis (weights, w). ID's are from 1 to 32.

id	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
w_c	0	0	0	0	1	0	0	0	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0	0	0	0	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{4}$
w_e	0	0	0	1	0	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0	0	0	$\frac{1}{2}$	0	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	0	0	0	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{4}$	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{8}$
w_p	0	0	1	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0	$\frac{1}{2}$	0	0	$\frac{1}{2}$	0	$\frac{1}{3}$	0	$\frac{1}{3}$	$\frac{1}{3}$	0	$\frac{1}{3}$	$\frac{1}{3}$	0	0	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{4}$
w_φ	0	1	0	0	0	$\frac{1}{2}$	0	$\frac{1}{2}$	0	$\frac{1}{2}$	0	0	$\frac{1}{2}$	0	0	$\frac{1}{3}$	$\frac{1}{3}$	0	$\frac{1}{3}$	$\frac{1}{3}$	0	$\frac{1}{3}$	0	$\frac{1}{3}$	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{4}$
w_λ	1	0	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$	0	0	$\frac{1}{2}$	0	0	0	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	0	$\frac{1}{3}$	$\frac{1}{3}$	0	$\frac{1}{3}$	0	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	0	$\frac{1}{5}$	$\frac{1}{8}$

Table 3: Parameter sets used in video analysis (thresholds, $\boldsymbol{\tau}$).

Threshold ID	1	2	3	4	5	6	7	8	9
Threshold τ	0.15	0.18	0.22	0.26	0.30	0.33	0.37	0.41	0.45

Since we do not need accurate motion estimation to classify the shots or for adaptive indexing, an estimate of the amount of motion in the shot is enough. Thus, we can approximate the amount of motion using the differences between adjacent frames (e.g., by analyzing the FD-sequence), rather than direct computation of the motion vectors. A similar observation has been made by Tao and Orchard [37], where they noticed that the residual signal generated after motion-compensated predication is highly correlated with the gradient magnitude: the motion compensated error is larger for pixels with larger gradient magnitude on average. They thus suggested that the gradient (from one frame to the other) could be estimated from the reconstructed image using the motion estimates. In this work, we are interested in the reverse procedure; given the gradient information (as captured by the edge response vectors), we wish to estimate the amount of motion in the shot, without explicit motion estimation.

We can estimate both the image activity and the motion by using the already available multilevel edge response vectors, with appropriate weights. For example, if we use $\forall i, w_i = 1/N$ (e.g., $w_i = 1/85$, for 4 decomposition levels), or if we ignore the global averages altogether, (i.e., the contributions from level $k = 0$), then the lower-level features (which are increasingly localized) can be used to predict the amount of motion. We could also ignore further higher level features, for instance, levels at $k \leq L/2$. We can estimate the activity by using the MERVs from just one frame in a given shot.

The motion and activity will generally result in an overall variability of the shot. The shot complexity depends directly on this *shot variability*. To estimate the shot variability, we use the mean and standard deviation of the frame-difference sequence (the FD-sequence) within the shot. We compute this for each of the MERV features, and use a weighted average to determine the shot variability. Given two time instants, t_1 and t_2, $(t_2 > t_1)$, we compute the shot variability as follows. Let $T = t_2 - t_1$ be the duration. Let FD_λ be the frame difference sequence using a particular multilevel feature, say λ:

$$\begin{aligned}\mu_\lambda(t_1,t_2) &= \frac{1}{t_2-t_1}\sum_{t=t_1}^{t_2}\mathrm{FD}_\lambda(t) = \frac{1}{T}\sum_{t=t_1}^{t_2}\mathrm{FD}_\lambda(t),\\ \sigma_\lambda^2(t_1,t_2) &= \frac{1}{T-1}\sum_{t=t_1}^{t_2}\left(\mathrm{FD}_\lambda(t)-\mu_\lambda(t_1,t_2)\right)^2.\end{aligned} \tag{22}$$

Similarly, we compute for color, edge response, edge-response at edge points, and the phase angle. Then, as with the between-frame distances, we obtain the shot-variability using a weighted combination from all the features:

$$\begin{aligned}\mu(t_1,t_2) &= w_c\mu_c(t_1,t_2) + w_e\mu_e(t_1,t_2) + w_p\mu_p(t_1,t_2)\\ &\quad + w_\varphi\mu_\varphi(t_1,t_2) + w_\lambda\mu_\lambda(t_1,t_2),\\ \sigma(t_1,t_2) &= w_c\sigma_c(t_1,t_2) + w_e\sigma_e(t_1,t_2) + w_p\sigma_p(t_1,t_2)\\ &\quad + w_\varphi\sigma_\varphi(t_1,t_2) + w_\lambda\sigma_\lambda(t_1,t_2).\end{aligned} \tag{23}$$

The weights here may not necessarily be the same as those used for the distances.

In [4, 9], different methods were proposed for computing the motion and image complexities, for instance, using the spectral entropy, and other metrics. With the above approach, one problem will be computing the standard deviation at each frame as the shot is progressing. This problem can be solved by doing the computations at only defined periodic intervals (the periods could also be chosen adaptively). However, one advantage of using the shot variability defined above is that the parameters required can be computed incrementally, using the preceding values. We can do this from the general definition of mean and standard deviation.

Given a data ensemble, $X = x_1, x_2, x_3, \ldots, x_{k-1}, x_k, x_{k+1}, \ldots$, and the mean of the first k items, $\mu_k = \mu(x_1, x_k) = (1/k)\sum_{i=1}^{k} x_i$, we can estimate the mean when the $(k+1)$th item is added:

$$\mu_{k+1} = \frac{k\mu_k + x_{k+1}}{k+1}. \tag{24}$$

Similarly, for the variance (or standard deviation), we have

$$\begin{aligned}(k-1)\sigma_k^2 &= \sum_{i=1}^{k} (x_i - \mu_k)^2 \\ (k+1-1)\sigma_{k+1}^2 &= \sum_{i=1}^{k+1} (x_i - \mu_{k+1})^2 \\ &= \sum_{i=1}^{k} (x_i - \mu_{k+1})^2 + (x_{k+1} - \mu_{k+1})^2.\end{aligned} \tag{25}$$

Solving (25) simultaneously, we obtain the incremental formula

$$\sigma_{k+1}^2 = \frac{(k-1)\sigma_k^2 + k(\mu_k - \mu_{k+1})^2 + (x_{k+1} - \mu_{k+1})^2}{k}. \tag{26}$$

We can use these to incrementally estimate the shot variability using the available FD-sequence. Based on the shot variability, we classify the shots into nine classes, as follows. Given $\mu(t_1, t_2)$ and $\sigma(t_1, t_2)$ for a given shot, we classify each into three classes, namely, low (I), medium (II), and high (III), based on an equi probability classification. Let $\mu_c(t_1, t_2) \in \{I, II, III\}$ be the classification due to $\mu(t_1, t_2)$. Similarly, let $\sigma_c(t_1, t_2) \in \{I, II, III\}$ be the classification due to $\sigma(t_1, t_2)$. Using the classifications from the two dimensions of shot variability, we define a simple mapping function $f(\cdot)$ to determine the overall shot class, viz:

$$S_c = f(\mu_c(t_1, t_2), \sigma_c(t_1, t_2)) = \begin{cases} I, & (\mu_c, \sigma_c) = (I, I), \\ II, & (\mu_c, \sigma_c) = (II, I), \\ III, & (\mu_c, \sigma_c) = (I, II), \\ IV, & (\mu_c, \sigma_c) = (III, I), \\ V, & (\mu_c, \sigma_c) = (I, III), \\ VI, & (\mu_c, \sigma_c) = (II, II), \\ VII, & (\mu_c, \sigma_c) = (III, II), \\ VIII, & (\mu_c, \sigma_c) = (II, III), \\ IX, & (\mu_c, \sigma_c) = (III, III). \end{cases} \tag{27}$$

Table 4 shows the classification results for the test video sequence, based on the above scheme.

4.2.2. Adaptive Shot Thresholds. Having characterized and classified the shots based on the shot variability, the next question is to determine the parameters for video shot partitioning for a given shot. Ideally, given the FD sequence, (and assuming that it was obtained from a distance (and not a similarity) measure), we expect that the threshold for shot changes should decrease with increasing shot length, but increase with increasing shot complexity (or variability). Formally, given a video shot s_j, we classify it into a certain shot class, $c_i \in \{I, II, III, \ldots, IX\}$. The problem of shot-level adaptation then is to determine the parameter set (i.e., the (w, τ) pair) that will produce the best results for all shots, $s_j \in c_i$, $\forall i, j$. Here, best results are defined in terms of information retrieval measures of precision and recall.

We take a pragmatic approach to the problem of determining the parameters. Using a training set of video shots, we use a simple clustering technique to determine the (w, τ) pairs that produce the best results for each shot class in the training set. We then use these pairs for analysis of the test video sequences.

Let $P = (w, \tau)$ be the weight-threshold pair that defines the parameter set for video segmentation. Let V be the number of video sequences used for the training set. We use the edge-response vectors to analyze the video shots in the training set, using all the available weights and thresholds (i.e., 32 weights and 9 thresholds in all, see Tables 2 and 3. Let P_j^c denote the set of (w, τ) pairs that produced correct partitioning results for the class c shots in video sequence j. To select the best (w, τ) pair for a given shot class, c, all we need is the intersection of P_j^c, for all the V sequences:

$$P^c = P_1^c \cap P_2^c \cap P_3^c \cdots \cap P_V^c. \tag{28}$$

When we have $|P^c| > 1$, then any member of P^c can be used as the best parameter set. The major problem is when $P^c = \varnothing$, that is, the intersection is empty, implying that no single parameter set always produced correct results for all the class c shots in the training sequences. Two approaches can be used to address this problem.

For each shot in a given video sequence, we define an array $a_{i,j}$, $i = 1, 2, \ldots, w_{\max}$, $j = 1, 2, \ldots, \tau_{\max}$, such that $a_{i,j} = 1$ if the shot is correctly partitioned with the parameter set (i, j) pair, and $a_{i,j} = 0$ otherwise. We use $w_{\max} = 32$, $\tau_{\max} = 9$ in our implementation. Let $a_{i,j}^c(q)$ denote the cumulative value in the $a_{i,j}$ arrays for all the class c shots in video sequence q. Then, the best parameter set for the class c shots is determined as

$$P^c = \arg\max_{i,j} \left\{ a_{i,j}^c \right\}, \tag{29}$$

where, $a_{i,j}^c = \sum_{q=1}^{V} a_{i,j}^c(q)$.

The above selects the parameter set that produced the best overall result, over all the shots of a given class in the training set. This could be dominated by one video sequence that has many shots of the given type. A variation could be to use the parameter set that produced the best result over the shots of a given class from most of the sequences, although it may not necessarily produce the best results over all shots. Thus,

$$P^c = \arg\max_{q} \{P^c(q)\}, \tag{30}$$

where, $P^c(q) = \text{argmax}_{i,j}\{a_{i,j}^c(q)\}$.

Table 4: Shot classification results based on shot variability.

Class	Antelope	Canyon	Crops	Culture	Journal	LAS	Total
I	20	4	6	28	15	34	107
II	0	1	0	0	0	0	1
III	9	0	5	4	40	7	65
IV	0	2	0	0	1	0	3
V	2	0	1	0	5	2	10
VI	0	4	0	1	2	0	7
VII	0	5	0	0	3	0	8
VIII	1	0	0	0	0	1	2
IX	0	2	1	0	1	0	4
Total	32	18	13	33	67	44	211

Table 5: Effectiveness of MERVs for non-adaptive video partitioning.

Video	Shots	Retrieved	Correct	False	Miss	Pr	Rc
Antelope	32	37	30	7	2	0.81	0.94
Canyon	18	18	18	0	0	1.00	1.00
Crops	13	16	13	3	0	0.81	1.00
Culture	33	21	20	1	13	0.95	0.61
Journal	67	70	65	5	2	0.93	0.97
LAS	44	35	35	0	7	1.00	0.83
Average						0.92	0.89

5. Results

To test the performance of the proposed edge-based adaptive method, we ran some experiments using two sets of video sequences. The first set had 6 sequences taken from standard MPEG-7 sequences, and from available online video sources [31]. For each video sequence, the frame size was fixed at 352×288. The second set had 5 sequences taken from the US National Institute of Standards (NIST) benchmark TRECVID 2001 test sequences. The frame size for sequences in this set was 320×240. The experiments were carried out in a MATLAB Version 7.3.0.267 (R2006b) environment using a personal computer with Intel(R) CPU T2400, running at 1.83 GHz with 1.99 GB RAM. We measure performance in terms of the information retrieval measures of precision and recall. We use the following notation: D = set of all positions of true scene cuts in a test video sequence, B = set of all positions of scene cuts returned by the system, C = subset of B that are *true* scene cuts (i.e., correct detection, or $C = B \cap D$). Then, precision $\text{Pr} = |C|/|B|$, and recall $\text{Rc} = |C|/|D|$.

5.1. Effectiveness of MERVs on Non-Adaptive Partitioning. First, we tested the effectiveness of the proposed edge-response vectors in video partitioning, without consideration for adaptation. This is important, since the results of the adaptive schemes will also be influenced by the inherent robustness of the edge-based features. The results are shown in Table 5. As can be seen, the edge-oriented approach produced about 90% in terms of precision and recall.

5.2. Adaptive Partitioning. Table 6 shows the results for adaptation at the video sequence level. The last two columns show the weight-threshold parameter pairs that were used to produce the indicated results. Where there are more than two entries, it means that the indicated entries all produced the same result. The table shows a significant improvement over the non-adaptive approach. The sequence-level adaptation is a two-pass method. That is, it needs a first pass on the data to determine the analysis parameters, and a second pass to perform the analysis. For some applications, such as real-time video streaming, the two-pass approach may not be applicable. Shot-level adaptation avoids the two-pass problem. Table 7 shows that results for shot-level adaptation, based on shot characterization and classification using the proposed shot variability measure. The results are a little worse than the two-pass method using sequence-level adaptation, but generally better than the static approach.

5.3. Comparative Results. We performed a comparative experiment using other popular techniques. Table 8 shows the results. For color histograms, we used region-based histograms with 16 blocks ($(M_1/4) \times (M_2/4)$ regions) per frame, where M_1 and M_2 are the frame dimensions. Analysis using motion-vector-based methods [28] are based on 20×20 sub blocks. The specific kernel size used for Cooper et al.'s DCT-based method [25] are also indicated in the table, as this varied significantly from sequence to sequence. In all cases, we have reported results using the parameters that gave the best overall result for a given video sequence,

Table 6: Results for proposed sequence-level adaptive partitioning.

Video	Shots	Retrieved	Correct	False	Miss	Pr	Rc	Weight, w	Threshold, τ
Antelope	32	31	30	1	2	0.97	0.94	12	0.26
Canyon	18	18	18	0	0	1.00	1.00	15, 18, 25, 29	0.26
Crops	13	13	13	0	0	1.00	1.00	15	0.37
Culture	33	33	33	0	0	1.00	1.00	9, 23	0.18
Journal	67	71	67	4	0	0.94	1.00	23	0.3
LAS	44	45	44	1	0	0.98	1.00	23, 27, 28	0.18
Average						0.98	0.99		

Table 7: Results for adaptive partitioning using proposed shot variability measure.

Video	Shots	Retrieved	Correct	False	Miss	Pr	Rc
Antelope	32	35	31	4	1	0.89	0.97
Canyon	18	19	18	1	0	0.95	1.00
Crops	13	13	12	0	1	1.00	0.92
Culture	33	27	26	1	7	0.96	0.78
Journal	67	60	59	1	8	0.98	0.88
LAS	44	44	44	0	0	1.00	1.00
Average						0.96	0.93

or for the test video set used. Apart from the results in [9], none of the other methods used adaptive partitioning. Thus, we can compare the performance of the static (non-adaptive) method using the proposed MERVs as features with the results from the other schemes. The table shows that MERV features are very competitive, having a comparable performance with the correlation-based method [27], the best performing technique of the other schemes tested. While simple color histogram did well on some video sequences, it produced poor performance on CROPS and CANYON video sets. This is mainly because these two sequences have both indoor and outdoor scenes involving significant variation in illumination. Obviously color features are easily affected by this variation, and hence the precision of the color-histogram based method was quite low for these sequences. The same explains the poor performance of the motion-vector-based method. Illumination variation between frames often leads to poor motion detection and thus a significant error in the motion vectors (even with the special parameter for illumination handling used in [28]). Overall, the results from the adaptive schemes are generally better than those from the non-adaptive schemes. This can be explained by the fact that the adaptive schemes spend time to analyze each shot first, before deciding on the analysis parameters. Thus, they are able to adapt better to the changing nature of shot characteristics as we move along in the video sequence.

We then tested the methods on another set of video sequences, this time using five sequences from the NIST benchmark TRECVID 2001 video sequences. The sequences and annotations by NIST, such as positions of true scene cuts are available via the NIST TRECVID website (http://www-nlpir.nist.gov/projects/trecvid/revised.html). (We could not get access to more recent sequences used in the TRECVID series. The most recent versions are available only for competitors in the TRECVID challenge. All the same, we believe that the 2001 data still provides another independent data set suitable for testing the algorithms). The results on the TRECVID sequences are shown in Table 9. The overall result is not too different from those of Table 8. Both the proposed method and the correlation method produced better results than the others. Both had about the same average recall, with the proposed method performing slightly better in recall (0.922 versus 0.906).

We also compared the proposed adaptive scheme with the scene-adaptive method proposed in [9]. The major difference was in terms of scene characterization. After characterization, we then used the same MERV features to perform video partitioning. Thus, this essentially compares the performance of the proposed shot variability measure for video shot characterization and classification against that of characterization using explicit motion and activity. Using shot variability for shot characterization and classification is slightly superior to using motion and activity complexity measures [9], with (precision, recall) values of (0.96, 0.93) versus (0.94, 0.91). A more striking difference, however, can be observed by considering the computational requirements for the two approaches. Using shot variability as a shot complexity measure is about 5 times faster than using motion and activity. The shot variability measure does not involve explicit motion estimation and activity characterization, but rather uses the same features (i.e., the FD-sequence) that were used in the analysis. Thus, it is generally more efficient than using motion and activity. Table 10 shows the overall time taken by the different methods in video partitioning. The reported time represents the average feature extraction time per frame required in analyzing a given video sequence.

Table 8: Comparative results with other video partitioning algorithms. The last three schemes are propsoed in this work.

Video	Other Proposed Techniques									Methods Proposed in this Work					
	Color histograms		Motion-vector likelihoods [28]		Correlation-based [27]		Kernel-correlation [25]			Non-adaptive MERVs		Sequence-level adaptation		Shot-level adaptation	
	Pr	Rc	Pr	Rc	Pr	Rc	Pr	Rc	BestKernel	Pr	Rc	Pr	Rc	Pr	Rc
Antelope	0.91	0.97	0.50	0.71	0.91	0.94	0.68	0.84	5×5	0.81	0.94	0.97	0.94	0.89	0.97
Canyon	0.53	1.00	0.78	0.60	1.00	0.94	0.74	1.00	6×6	1.00	1.00	1.00	1.00	0.95	1.00
Crops	0.31	1.00	0.35	0.84	1.00	1.00	0.71	1.00	4×4	0.81	1.00	1.00	1.00	1.00	0.92
Culture	0.89	0.97	0.32	0.73	0.89	1.00	0.76	0.84	5×5	0.95	0.61	1.00	1.00	0.96	0.78
Journal	0.98	0.72	0.36	0.76	0.67	0.94	0.78	0.54	5×5	0.93	0.97	1.00	0.97	0.98	0.88
LAS	0.97	0.82	1.00	0.53	0.84	0.84	0.78	0.89	4×4	1.00	0.83	1.00	0.99	1.00	1.00
Average	0.77	0.91	0.55	0.70	0.89	0.94	0.74	0.85		0.92	0.89	1.00	0.98	0.96	0.93

Table 9: Comparative results with other video partitioning algorithms on TRECVID 2001 dataset.

Sequence	Number of shots	Color histograms		Motion-vector likelihoods [28]		Correlation-based [27]		Kernel-correlation [25] (5×5 kernel)		Kernel-correlation [25] (6×6 kernel)		Proposed method	
		Pr	Rc	Pr	Rc	Pr	Rc	Pr	Rc	Pr	Rc	Pr	Rc
anni005	38	0.64	0.83	0.46	0.53	0.87	0.89	0.71	0.64	0.72	0.64	0.87	0.91
anni006	41	0.78	0.70	0.57	0.56	0.84	0.88	0.71	0.68	0.79	0.69	0.82	0.89
anni009	38	0.84	0.71	0.59	0.62	0.86	0.94	0.81	0.78	0.83	0.81	0.87	0.93
BOR08	197	0.78	0.78	0.24	0.64	0.85	0.88	0.60	0.83	0.69	0.81	0.86	0.91
NAD53	83	0.69	0.62	0.46	0.73	0.79	0.94	0.69	0.84	0.75	0.80	0.81	0.97
	Average	0.75	0.73	0.46	0.62	0.84	0.91	0.71	0.75	0.76	0.75	0.85	0.92

Table 10: CPU time taken for feature extraction for different methods.

Method	CPU Time (sec)
Color histogram	0.0180
Motion-vector likelihoods [28]	0.3991
Correlation-based [27]	0.0270
Kernel-correlation [25] (with 5×5 kernel)	0.9515
Proposed adaptive scheme	0.1210

6. Discussion and Conclusion

Although video partitioning is an actively researched area, recent publications [9, 20, 24–28] show that the problem is far from being completely resolved. The major contributions of this paper are on two aspects of the video partitioning problem. The first is the proposed new set of features (the multilevel edge response vectors (MERVs)) for video partitioning. The edge-based nature of the features makes them particularly suitable in handling significant illumination variations in the video, while the multilevel decomposition framework makes it possible to adapt the features to the nature of the video frames being considered. The second contribution is on adaptive video partitioning. While adaptive video partitioning was first described in [9, 30], here we propose a new and more efficient method for performing the scene characterization required for scene partitioning, and a method for automated determination of thresholds based on the video shot classes. The proposed method is online—performing scene characterization and classification as the frames in a given shot are being observed, rather than waiting until the end of a given shot (as was done in [9]). This feature, coupled with the improved efficiency in shot characterization makes the approach particularly suitable for fast and online characterization of the video, which is important in both video retrieval, and in video traffic modeling for adaptive network resource allocation [15]. We mention that, while we have provided adaptation based on the MERV features proposed, the general idea of adaptation in video analysis is independent of the specific features being used. For any given feature, the idea of adaptation can be applied by a careful study of the feature in question, and then adapting the analysis parameters using this feature based on the nature of the shot being analyzed.

In conclusion, we have studied the problem of video segmentation, using an adaptive edge-oriented framework. Adaptation is provided by an analysis of the video shot characteristics using the frame difference sequence. In particular, we defined the shot variability measure, based on which the video shots are characterized and then classified. To provide adaptation in the analysis, we determine the best set of parameters for each given shot class, and then analyze the shots that belong to the given class using only these parameter sets. An algorithm for determining the best parameters for each given shot class is presented. We described adaptation at three levels: at the feature extraction stage for the locally-adaptive edge maps, at the video sequence level, and at the individual shot level.

Experimental results show that the proposed multilevel edge-based features provide a performance of about 90% in terms of average precision and recall. In comparison with traditional approaches, the adaptive schemes provide a better performance over non-adaptive approaches, using the same multilevel edge-based features—with video sequence level adaptation producing about 99% performance. Further, the use of shot variability as a measure of shot complexity resulted in a slightly superior performance (about 2% improvement in precision) over a previously proposed method of explicit motion estimation and shot activity analysis. However, in terms of efficiency, using the shot variability led to a five fold improvement in efficiency. The reported work has applications beyond video indexing and retrieval. In particular, given the significant reduction in computations, the approach becomes attractive for real-time applications, such as in dynamic monitoring, characterization and modeling of video data traffic, and in real-time video surveillance.

References

[1] M. J. Swain and D. H. Ballard, "Color indexing," *International Journal of Computer Vision*, vol. 7, no. 1, pp. 11–32, 1991.

[2] A. Nagasaka and Y. Tanaka, "Automatic video indexing and full-video search for object appearances," in *Visual Database Systems II*, E. Knuth and L. M. Wegner, Eds., pp. 113–127, Elsevier, 1992.

[3] H. Zhang, A. Kankanhalli, and S. W. Smoliar, "Automatic partitioning of full-motion video," *Multimedia Systems*, vol. 1, no. 1, pp. 10–28, 1993.

[4] D. A. Adjeroh and M. C. Lee, "Robust and efficient transform domain video sequence analysis: an approach from the generalized color ratio model," *Journal of Visual Communication and Image Representation*, vol. 8, no. 2, pp. 182–207, 1997.

[5] R. Zabih, J. Miller, and K. Mai, "A feature-based algorithm for detecting and classifying production effects," *Multimedia Systems*, vol. 7, no. 2, pp. 119–128, 1999.

[6] J. D. Courtney, "Automatic video indexing via object motion analysis," *Pattern Recognition*, vol. 30, no. 4, pp. 607–625, 1997.

[7] P. Bouthemy, M. Gelgon, and F. Ganansia, "A unified approach to shot change detection and camera motion characterization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 7, pp. 1030–1044, 1999.

[8] S. Dagtas, W. Al-Khatib, A. Ghafoor, and R. L. Kashyap, "Models for motion-based video indexing and retrieval," *IEEE Transactions on Image Processing*, vol. 9, no. 1, pp. 88–101, 2000.

[9] D. A. Adjeroh and M. C. Lee, "Scene-adaptive transform domain video partitioning," *IEEE Transactions on Multimedia*, vol. 6, no. 1, pp. 58–69, 2004.

[10] N. Vasconcelos and A. Lippman, "Statistical models of video structure for content analysis and characterization," *IEEE Transactions on Image Processing*, vol. 9, no. 1, pp. 3–19, 2000.

[11] V. S. Subrahmanian, *Principles of Multimedia Database Systems*, Morgan Kaufmann, San Mateo, Calif, USA, 1998.

[12] T. C. T. Kuo and A. L. P. Chen, "Content-based query processing for video databases," *IEEE Transactions on Multimedia*, vol. 2, no. 1, pp. 1–13, 2000.

[13] C. Taskiran, J.-Y. Chen, A. Albiol, L. Torres, C. A. Bouman, and E. J. Delp, "ViBE: a compressed video database structured for active browsing and search," *IEEE Transactions on Multimedia*, vol. 6, no. 1, pp. 103–118, 2004.

[14] S.-C. S. Cheung and A. Zakhor, "Fast similarity search and clustering of video sequences on the world-wide-web," *IEEE Transactions on Multimedia*, vol. 7, no. 3, pp. 524–537, 2005.

[15] A. H. M. Dawood and M. Ghanbari, "Content-based MPEG video traffic modeling," *IEEE Transactions on Multimedia*, vol. 1, no. 1, pp. 77–87, 1999.

[16] F. Arman, A. Hsu, and M.-Y. Chiu, "Image processing on encoded video sequences," *Multimedia Systems*, vol. 1, no. 5, pp. 211–219, 1994.

[17] A. Hampapur, R. Jain, and T. E. Weymouth, "Production model based digital video segmentation," *Multimedia Tools and Applications*, vol. 1, no. 1, pp. 9–46, 1995.

[18] S.-W. Lee, Y.-M. Kim, and S. W. Choi, "Fast scene change detection using direct feature extraction from MPEG compressed videos," *IEEE Transactions on Multimedia*, vol. 2, no. 4, pp. 240–254, 2000.

[19] G. Ahanger and T. D. C. Little, "A survey of technologies for parsing and indexing digital video," *Journal of Visual Communication and Image Representation*, vol. 7, no. 1, pp. 28–43, 1996.

[20] A. Hanjalic, "Shot-boundary detection: unraveled and resolved?" *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 2, pp. 90–105, 2002.

[21] M. K. Mandal, F. Idris, and S. Panchanathan, "A critical evaluation of image and video indexing techniques in the compressed domain," *Image and Vision Computing*, vol. 17, no. 7, pp. 513–529, 1999.

[22] M. Abdel-Mottaleb and S. Krishnamachari, "Multimedia descriptions based on MPEG-7: extraction and applications," *IEEE Transactions on Multimedia*, vol. 6, no. 3, pp. 459–468, 2004.

[23] B. Shen and I. K. Sethi, "Direct feature extraction from compressed images," in *Storage and Retrieval for Still Image and Video Databases IV*, vol. 2670 of *Proceedings of SPIE*, pp. 404–414, San Jose, Calif, USA, February 1996.

[24] J. Bescos, G. Cisneros, J. M. Martinez, J. M. Menendez, and J. Cabrera, "A unified model for techniques on video-shot transition detection," *IEEE Transactions on Multimedia*, vol. 7, no. 2, pp. 293–307, 2005.

[25] M. Cooper, T. Liu, and E. Rieffel, "Video segmentation via temporal pattern classification," *IEEE Transactions on Multimedia*, vol. 9, no. 3, pp. 610–618, 2007.

[26] S. Li and M.-C. Lee, "Effective detection of various wipe transitions," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 6, pp. 663–673, 2007.

[27] H.-W. Yoo, H.-J. Ryoo, and D.-S. Jang, "Gradual shot boundary detection using localized edge blocks," *Multimedia Tools and Applications*, vol. 28, no. 3, pp. 283–300, 2006.

[28] W.-K. Li and S.-H. Lai, "Integrated video shot segmentation algorithm," in *Storage and Retrieval for Media Databases 2003*, M. M. Yeung, R. W. Lienhart, and C.-S. Li, Eds., vol. 5021 of *Proceedings of SPIE*, pp. 264–271, Santa Clara, Calif, USA, January 2003.

[29] M. Cooper, J. Foote, J. Adcock, and S. Casi, "Shot boundary detection via similarity analysis," in *Proceedings of the TRECVID Workshop*, November 2003.

[30] D. A. Adjeroh and M. C. Lee, "Adaptive transform domain video shot analysis," in *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, Ontario, Canada, June 1997.

[31] C. O'Toole, A. Smeaton, N. Murphy, and S. Marlow, "Evaluation of automatic shot boundary detection on a large video

test suite," in *Proceedings of Conference on Challenge of Image Retrieval*, Newcastle Upon Tyne, UK, February 1999.

[32] N. Vasconcelos and A. Lippman, "A Bayesian video modeling framework for shot segmentation and content characterization," in *Proceedings of Workshop on Content-Based Access to Image and Video Libraries*, San Juan, Puerto Rico, USA, 1997.

[33] Y. Rui, T. S. Huang, and S.-F. Chang, "Image retrieval: current techniques, promising directions, and open issues," *Journal of Visual Communication and Image Representation*, vol. 10, no. 1, pp. 39–62, 1999.

[34] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1349–1380, 2000.

[35] A. S. Al-Fahoum and A. M. Reza, "Combined edge crispiness and statistical differencing for deblocking JPEG compressed images," *IEEE Transactions on Image Processing*, vol. 10, no. 9, pp. 1288–1298, 2001.

[36] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: a review and a new contribution," *Proceedings of the IEEE*, vol. 83, no. 6, pp. 858–876, 1995.

[37] B. Tao and M. T. Orchard, "Gradient-based residual variance modeling and its applications to motion-compensated video coding," *IEEE Transactions on Image Processing*, vol. 10, no. 1, pp. 24–35, 2001.