Costas Cotsaces, Nikos Nikolaidis, and Ioannis Pitas

Video Shot Detection and Condensed Representation

A review

The advances in digital video technology and the ever-increasing availability of computing resources have resulted in the last few years in an explosion of digital video data, especially on the Internet. However, the increasing availability of digital video has not been accompanied by an increase in its accessibility. This is due to the nature of video data, which is unsuitable for traditional forms of data access, indexing, search, and retrieval, which are either text based or based on the query-by-example paradigm. Therefore, techniques have been sought that organize video data into more compact forms or extract semantically meaningful information [1]. Such operations can serve as a first step for a number of different data access tasks, such as browsing, retrieval, genre classification, and event detection. Here we focus not on the high-level video analysis tasks themselves but on the common basic techniques that have been developed to facilitate them. These basic tasks are shot boundary detection and condensed video representation.

Shot boundary detection is the most basic temporal video segmentation task, as it is intrinsically and inextricably linked to the way that video is produced. It is a natural choice for segmenting a video into more manageable parts. Thus, it is very often the first step in algorithms that accomplish other video analysis tasks, one of them being

1053-5888/06/$20.00©2006IEEE

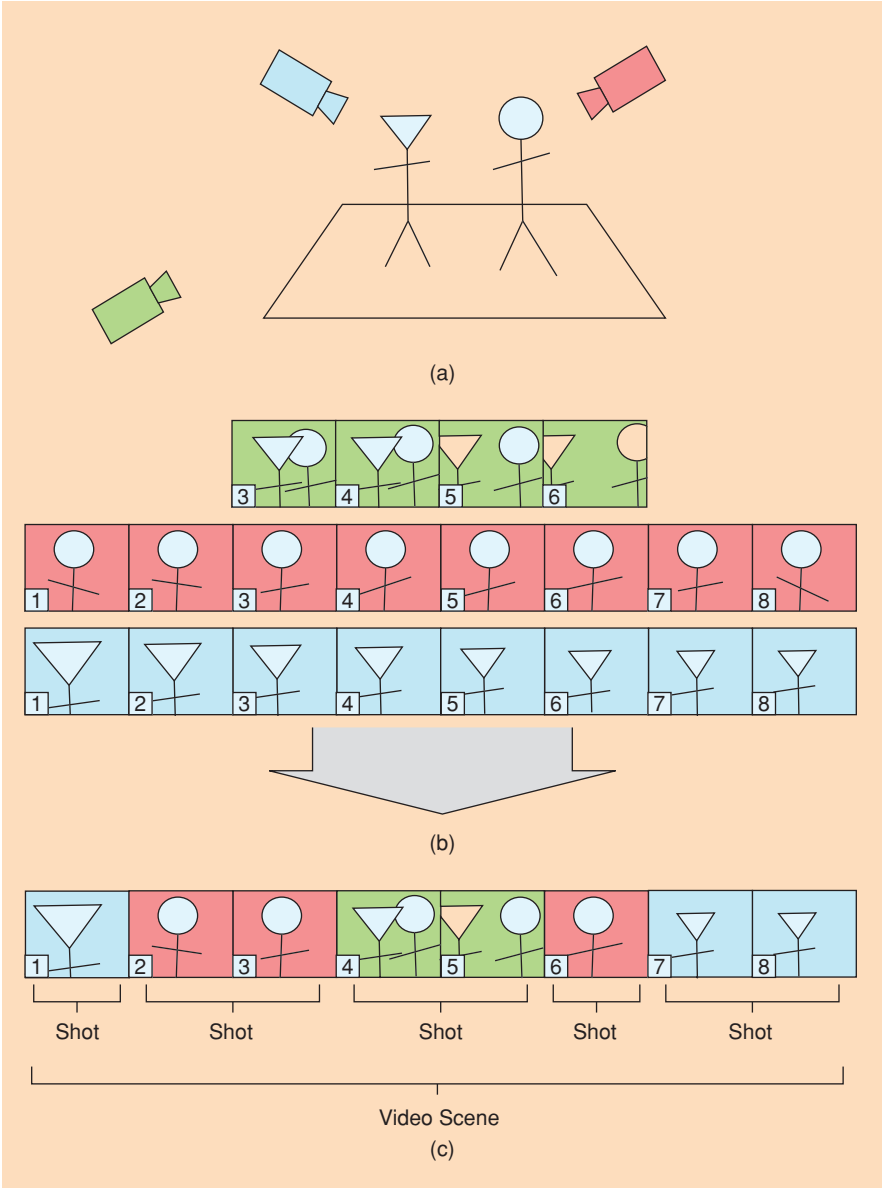
condensed video representation as described below. In the case of video retrieval, a video index is much smaller and thus easier to construct and use if it references whole video shots rather than every video frame. Since scene changes almost always happen on a shot change, shot boundary detection is indispensable as a first step for scene boundary detection. Finally, shot transitions provide convenient jump points for video browsing. Condensed representation is the extraction of a characteristic set of either independent frames or short sequences from a video. This can be used as a substitute for the whole video for the purposes of indexing, comparison, and categorization. It is also especially useful for video browsing. The article by Xiong et al. [25] provides a detailed account of condensed video representation applications. The results of both shot boundary detection and condensed video representation do not need to be immediately directed to the above applications; they may instead be stored as metadata and used when they are needed. This can be achieved by the use of standards, such as MPEG-7 [2], which contain appropriate specifications for the description of shots, scenes, and various types of condensed representation.

Several reviews on shot boundary detection have been published in the last decade [3]–[6], yet none later than 2001. There is little review work on the subject of condensed video representation published in refereed journals.

SHOT BOUNDARY DETECTION

The concept of temporal image sequence (video) segmentation is not a new one, as it dates back to the first days of motion pictures, well before the introduction of computers. Motion picture specialists perceptually segment their works into a hierarchy of partitions. A video (or film) is completely and disjointly segmented into a sequence of scenes, which are subsequently segmented into a sequence of shots. The concept of a scene (also called a story unit) is much older than motion pictures, ultimately originating in the theater. Traditionally, a scene is a continuous sequence that is temporally and spatially cohesive in the real world but not necessarily cohesive in the projection of the real world on film. On the other hand, shots originate with the invention of motion cameras and are defined as the longest continuous sequence that originates from a single camera take, which is what the camera images in an uninterrupted run, as shown in Figure 1.

In general, the automatic segmentation of a video into scenes ranges from very difficult to intractable. On the other hand, video segmentation into shots is both exactly defined and characterized by distinctive features of the video stream itself. This is because video content within a shot tends to be continuous, due to the continuity of both the physical scene and the parameters (motion, zoom, focus) of the camera that images it. Therefore, in principle, the detection of a shot change between two adjacent frames simply requires the computation of an appropriate continuity or similarity metric. However, this simple concept has three major complications.



[FIG1] Schematic sketch of scenes and shots as defined by the video production process: (a) the actual objects (or people) that are imaged, which are spatio-temporally coherent and thus comprise a scene; (b) continuous segments imaged by different cameras (takes); (c) the editing process that creates shots from takes.

The first, and most obvious one, is defining a continuity metric for the video in such a way that it is insensitive to gradual changes in camera parameters, lighting, and physical scene content, easy to compute, and discriminant enough to be useful. The simplest way to do this is to extract one or more scalar or vector features from each frame and define distance functions on the feature domain. Alternatively, the features themselves can be used either for clustering the frames into shots or for detecting shot transition patterns.

The second complication is deciding which values of the continuity metric correspond to a shot change and which do not. This is not trivial, since the feature variation within certain shots can exceed the respective variation across shots. Decision methods for shot boundary detection include fixed thresholds, adaptive thresholds, and statistical detection methods.

The third complication, and the most difficult to handle, is the fact that not all shot changes are abrupt. Using motion picture terminology, changes between shots can belong to the following categories, some of which are illustrated in Figure 2:

[FIG2] Examples of gradual transitions from the TRECVID 2003 corpus: (a) a dissolve between shots with similar color content; (b) a dissolve between shots with dissimilar color content; (c) fade; (d) a wipe of the "door" variety; (e) a wipe of the "grill" variety; and (f) a computer-generated special effect transition between shots.

1) *Cut*: This is the classic abrupt change case, where one frame belongs to the disappearing shot and the next one to the appearing shot.
2) *Dissolve:* In this case, the last few frames of the disappearing shot temporally overlap with the first few frames of the appearing shot. During the overlap, the intensity of the disappearing shot decreases from normal to zero (fade out), while that of the appearing shot increases from zero to normal (fade in).
3) *Fade:* Here, first the disappearing shot fades out into a blank frame, and then the blank frame fades into the appearing shot.
4) *Wipe:* This is actually a set of shot change techniques, where the appearing and disappearing shots coexist in different spatial regions of the intermediate video frames, and the region occupied by the former grows until it entirely replaces the latter.
5) *Other transition types:* There is a multitude of inventive special effects techniques used in motion pictures. They are, in general, very rare and difficult to detect.

COMPONENTS OF SHOT BOUNDARY DETECTION ALGORITHMS

As previously mentioned, shot boundary detection algorithms work by extracting one or more features from a video frame or a subset of it, called a region of interest (ROI). An algorithm can then use different methods to detect shot changes from these features. Since there are many different ways the above components can be combined, we have chosen not to provide a hierarchical decomposition of different classes of algorithms. Instead we present below the different choices that can be made for each component, along with their advantages and disadvantages. These can then be combined more or less freely to design a shot detection algorithm.

FEATURES USED

Almost all shot change detection algorithms reduce the large dimensionality of the video domain by extracting a small number of features from one or more regions of interest in each video frame. Such features include:

1) *Luminance/color:* The simplest feature that can be used to characterize an ROI is its average grayscale luminance. This, however, is susceptible to illumination changes. A more robust choice is to use one or more statistics (e.g., averages) of the values in a suitable color space [7]–[9], like hue saturation value (HSV).
2) *Luminance/color histogram:* A richer feature for an ROI is the grayscale or color histogram. Its advantage is that it is quite discriminant, easy to compute, and mostly insensitive to translational, rotational, and zooming camera motions. For these reasons, it is widely used [10], [11].
3) *Image edges:* An obvious choice for characterizing an ROI is its edge information [9], [12]. The advantage of this feature is that it is sufficiently invariant to illumination changes and several types of motion, and it is related to the human visual

perception of a scene. Its main disadvantage is computational cost, noise sensitivity, and, when not postprocessed, high dimensionality.

4) *Transform coefficients (discrete Fourier transform, discrete cosine transform, wavelet):* These are classic ways to describe the image information in an ROI. The discrete cosine transform (DCT) coefficients also have the advantage of being already present in MPEG-encoded video streams or files.

5) *Motion:* This is sometimes used as a feature for detecting shot transitions, but it is usually coupled with other features, since by itself it can be highly discontinuous within a shot (when motion changes abruptly) and is obviously useless when there is no motion in the video.

SPATIAL FEATURE DOMAIN

The size of the region from which individual features are extracted plays an important role in the overall performance of shot change detection. A small region tends to reduce detection invariance with respect to motion, while a large region might lead to missed transitions between similar shots. In the following, we will describe various possible choices:

1) *Single pixel:* Some algorithms derive a feature for each pixel. This feature can be luminance, edge strength [9], etc. However, such an approach results in a very large feature vector and is very sensitive to motion, unless motion compensation is subsequently performed.

2) *Rectangular block:* Another method is to segment each frame into equal-sized blocks and extract a set of features (e.g., average color or orientation, color histogram) from these blocks [7], [8]. This approach has the advantage of being invariant to small camera and object motion as well as being adequately discriminant for shot boundary detection.

3) *Arbitrarily shaped region:* Feature extraction can also be applied to arbitrarily shaped and sized frame regions derived by spatial segmentation algorithms. This enables the derivation of features based on the most homogeneous regions, thus facilitating a better detection of temporal discontinuities. The main disadvantage is the usually high computational complexity and instability of region segmentation.

4) *Whole frame:* The algorithms that extract features (e.g., histograms) from the whole frame [11], [13], [14] have the advantage of being very robust with respect to motion within a shot, but tend to have poor performance at detecting the change between two similar shots.

FEATURE SIMILARITY METRIC

To evaluate discontinuity between frames based on the selected features, an appropriate similarity/dissimilarity metric needs to be chosen. Assuming that a frame is characterized by K scalar features $F(k), k = 1..K$, the most traditional choice is to use a L_n norm:

$$D_{L_n}(i, j) = \left(\sum_{k=1}^{K} |F_i(k) - F_j(k)|^n\right)^{1/n}.$$

Examples of the above are the histogram difference, where $F(k)$ are the bins of the histogram and $n = 1$, and the image difference where $F(k)$ are the pixels of the (usually subsampled) image and $n = 2$. Another example of a commonly used metric, especially in the case of histograms, is the chi-square (χ^2):

$$D_{\chi^2}(i, j) = \sum_{k=1}^{K} \frac{(F_i(k) - F_j(k))^2}{F_i(k)}.$$

TEMPORAL DOMAIN OF CONTINUITY METRIC

Another important aspect of shot boundary detection algorithms is the temporal window that is used to perform shot change detection. In general, the objective is to select a temporal window that contains a representative amount of video activity. Options include:

1) *Two frames:* The simplest way to detect discontinuity between frames is to look for a high value of the discontinuity metric between two successive frames [8], [13], [15], [26]. However, such an approach can fail to discriminate between shot transitions and changes within the shot when there is significant variation in activity among different parts of the video or when certain shots contain events that cause brief discontinuities (e.g., photographic flashes). It also has difficulty in detecting gradual transitions.

2) *N-frame window:* The most common technique for alleviating the above problems is to detect the discontinuity by using the features of all frames within a suitable temporal window, which is centered on the location of the potential discontinuity [8], [9], [14], [16].

3) *Interval since last shot change:* Perhaps the most obvious method for detecting a shot end is to compute one or more statistics from the last detected shot change up to the current point, and to check if the next frame is consistent with them, as in [7] and [11]. The problem with such approaches is that there is often great variability within shots, meaning that statistics computed for an entire shot may not be representative of its end.

SHOT CHANGE DETECTION METHOD

Having defined a feature (or a set of features) computed on one or more ROIs for each frame (and, optionally, a similarity metric), a shot change detection algorithm needs to detect where these exhibit discontinuity. This can be done in the following ways:

1) *Static thresholding:* This is the most basic decision method, which entails comparing a metric expressing the similarity or dissimilarity of the features computed on adjacent frames against a fixed threshold [11]. This only performs well if video content exhibits similar characteristics over time and only if the threshold is manually adjusted for each video.

2) *Adaptive thresholding:* The obvious solution to the problems of static thresholding is to vary the threshold depending on a statistic (e.g., average) of the feature difference metrics within a temporal window, as in [13] and [16].

3) *Probabilistic detection:* Perhaps the most rigorous way to detect shot changes is to model the pattern of specific types of shot transitions and, presupposing specific probability distributions for the feature difference metrics in each shot, perform optimal a posteriori shot change estimation. This is demonstrated in [7] and [8].

4) *Trained classifier:* A radically different method for detecting shot changes is to formulate the problem as a classification task where frames are separated (through their corresponding features) into two classes, namely "shot change" and "no shot change," and train a classifier (e.g., a neural network) to distinguish between the two classes [14].

PERFORMANCE EVALUATION

The basic measures in detection and retrieval problems in general are recall, precision, and accuracy. Recall quantifies what proportion of the correct entities (shot changes in our case) are detected, while precision quantifies what proportion of the detected entities are correct. Accuracy reflects the temporal correctness of the detected results, and corresponds to the distance between the detected location of a transition and its true location. Therefore, if we denote by D the shot transitions correctly detected by the algorithm, by D_M the number of missed detections (the transitions that should have been detected but were not), and by D_F the number of false detections (the transitions that should not have been detected but were), we have that:

$$\text{Recall} = \frac{D}{D + D_M}$$

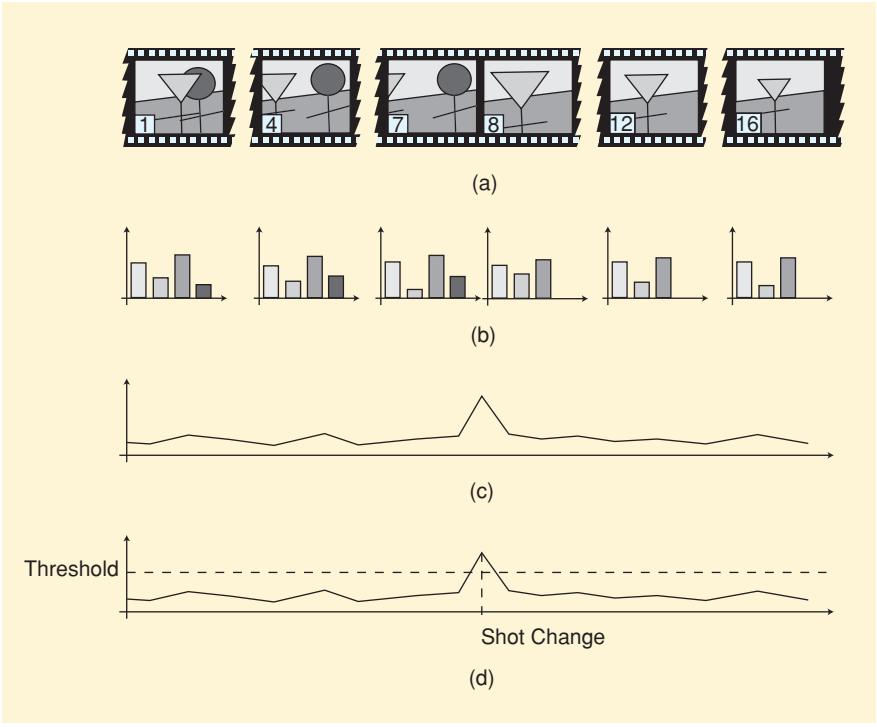
$$\text{Precision} = \frac{D}{D + D_F}.$$

Because of the richness and variety of digital video content, the effective quantitative (and even qualitative) comparison of the results of different video analysis algorithms is meaningless if they are derived from different video corpora. To alleviate this problem, the TREC video retrieval evaluation [17] (*TRECVID*) was organized. The goal of TRECVID is not only to provide a common corpus of video data as a testbed for different algorithms, but also to standardize and oversee their evaluation and provide a forum for the comparison of the results.

SPECIFIC SHOT BOUNDARY DETECTION ALGORITHMS

In the previous section, we analyzed the features, metrics, and decision methods that can be used to design a shot boundary detection algorithm. We illustrate the design of such an algorithm in Figure 3. Specifically, the intensity histogram of the whole frame has been chosen as a feature, the histogram difference as a discontinuity metric, and a constant threshold as the shot change detection method. In the following, we will also give some state-of-the-art examples of complete algorithms.

The shot boundary detection problem is thoroughly analyzed by Hanjalic [8], and a probabilistically-based algorithm is proposed. Each video frame is completely segmented into nonoverlapping blocks, and the average YUV color component values are extracted from each block. Blocks are matched between each pair of adjacent frames according to average difference between the color components between blocks, and the discontinuity value is then defined as the above-average difference for matching blocks. Adjacent frames are compared for detecting abrupt transitions, while for gradual ones, frames that are separated by the minimum shot length are compared. The a priori likelihood functions of the discontinuity metric are obtained by experiments on manually labeled data. Their choice is largely heuristic. The a priori probability of shot boundary detection, conditional on the time that elapsed since the last shot boundary detection, was modeled as a Poisson function; this was observed to produce good results. The shot boundary detection probability is refined with a heuristic derived from the discontinuity value pattern (for cuts and fades) and from in-frame



[FIG3] A typical shot detection algorithm. (a) Samples of the video content. (b) Histogram of the distribution of the intensity values in the displayed frames. (c) Histogram difference between adjacent frames. (d) A shot change is detected when the histogram difference exceeds a threshold.

variances (for dissolves) in a temporal window around the current frame. Effectively, a different detector is implemented for each transition type. In conclusion, although the proposed method claims to offer a rigorous solution to the shot boundary detection problem, it is somewhat heuristic. Yet the probabilistic analysis of the problem is valid and thorough. Additionally, experimental results are very satisfactory, being perfect for cuts and having good recall and precision for dissolves.

The approach developed by Lienhart [14], detects dissolves (and only dissolves) with a trained classifier (i.e., a neural network), operating on either YUV color histograms, magnitude of directional gradients, or edge-based contrast. The classifier detects possible dissolves at multiple temporal scales and merges the results using a winner-take-all strategy. The interesting part is that the classifier is trained using a dissolve synthesizer, which creates artificial dissolves from any available set of video sequences. Although the videos used for experimental verification are nonstandard, performance is shown to be superior when compared to simple edge-based dissolve detection methods.

Cernekova et al. [11] propose performing singular value decomposition (SVD) on the RGB color histograms of each frame to reduce feature vector dimensionality, in order to make the feature space more discriminant. Initially, video segmentation is performed by comparing the angle between the feature vector of each frame and the average of the feature vectors of the current segment. If their difference is higher than a static threshold, a new segment is started. Segments whose feature vectors exhibit large dispersion are considered to depict a gradual transition between two shots, whereas segments with small dispersion are characterized as shots. Experiments were run on a database of four real TV sequences. The main problem with this approach is the static threshold applied on the angle between vectors to detect shot change, which may be problematic in the case of large intrashot content variation and small intershot content variation.

Boccignone et al. [16] approach the shot boundary detection problem from a completely new angle, using the attentional paradigm for human vision. Fundamentally, their algorithm computes for every frame a set (called a trace) of points of focus of attention (FOA) in decreasing order of saliency and then compares nearby frames by evaluating the consistency of their traces. Shot boundaries are found when the above similarity is below a dynamic threshold. Specifically, FOAs are detected by first computing a saliency map from the normalized sum of multiscale intensity contrast, color contrast (blue-yellow and red-green), and orientation contrast. Then, the most salient points in this map are found using a winner-take-all artificial neural network, which also gives a measure of their saliency. The FOAs are accompanied by the color and shape information in their neighborhoods. The consistency of two traces is computed by first finding homologous FOAs between the two traces based on three different types of local consistency: spatial, temporal (i.e., referring to order of saliency), and visual (i.e., referring to neighborhood information). A cut is then detected by comparing the interframe consistency with a probabilistically derived adaptive threshold, while dissolves are detected in a subsequent phase. The results, given for a subset of TRECVID01 and some news and movie segments, are overall satisfactory but not above the state of the art.

Lelescu and Schonfeld [7] present an interesting statistical approach, which additionally benefits from operating on MPEG-compressed videos. They extract the average luminance and chrominance for each block in every I and P frame and then perform principal component analysis (PCA) on the resulting feature vectors. It should be noted that the eigenvectors are computed based only on the M first frames of each shot. The resulting projected vectors are modeled by a Gaussian distribution whose mean and covariance matrixes are estimated from the M first frames of each shot. The originality of the approach is that a change statistic is estimated for each new frame using a maximum likelihood methodology (the generalized likelihood ratio) and, if it exceeds an experimentally determined threshold, a new shot is started. The algorithm has been tested on a number of videos where gradual transitions are common. The test videos used are characterized by the particularly short duration of their shots. This highlights the greatest problem of this algorithm, namely that the extraction of the shot characteristics (eigenspace, mean, and covariance) happens at its very beginning (first M frames). This means that in the case of long and nonhomogeneous shots, like the ones common in motion pictures, the values calculated at the start of the shot might not be characteristic of its end.

The joint probability image (JPI) of two frames is defined by Li et al. [15] as a matrix whose $[i, j]$ element contains the probability that a pixel with color i in the first image has color j in the second image. They also derive the joint probability projection vector (JPPV) as a one-dimensional projection of the JPI, and the joint probability projection centroid (JPPC) as a scalar measure of the JPI's dispersion. Specific types of transition (dissolve, fade, dither) are observed to have specific JPI patterns. To detect shot changes, possible locations are first detected using the JPPC, and then they are refined, validated, and classified into transition types by matching the JPPV with specific transition patterns. Results are given for a relatively small number of abrupt and gradual transitions. The authors admit the obvious problems this method has with large object and/or camera motion.

Another way to attack the most difficult problem in shot boundary detection, which is the detection of gradual transitions, is to use different algorithms for each type of transition. This is the path followed by Nam and Tewfik [9], who designed two algorithms, one for dissolves and fades and one for wipes. The fade/dissolve detector starts by approximating, within a temporal window, the temporal evolution of the intensity of each pixel in the frame with a B-spline. It then computes a change metric as the negative of the normalized sum of the errors of the above approximations. This metric is

then discretized by comparison with a locally adaptive threshold, processed by a morphological filter and used as an indicator of the existence of dissolves and fades. Fades are distinguished from dissolves by additionally checking whether the interframe standard deviation is close to zero. The wipe detector functions by first calculating the difference between one frame and the next, and then multiplying this difference with parts of the frame's two-dimensional (2-D) wavelet transform to enhance directionality. Then, the radon transform of each frame is taken for the four cardinal directions (horizontal, vertical, and the two diagonals) and the same is done for the half-frames (but only for the horizontal and vertical directions). For each frame, the location of the maximum is taken from each of the above projections, representing the location of the strongest edge in each direction. Tracking these maxima over time results in six temporal curves. Wipe candidates are then detected when these curves exhibit a linear increase or decrease, corresponding to the regular movement of the wipe's edge. Finally, the candidates are considered to be verified if they can be efficiently approximated with a B-spline. The results of these methods are close to the state-of-the-art for the sequences tested, but the approach has problems in detecting types of transitions other than the ones it was designed for.

CONDENSED VIDEO REPRESENTATION

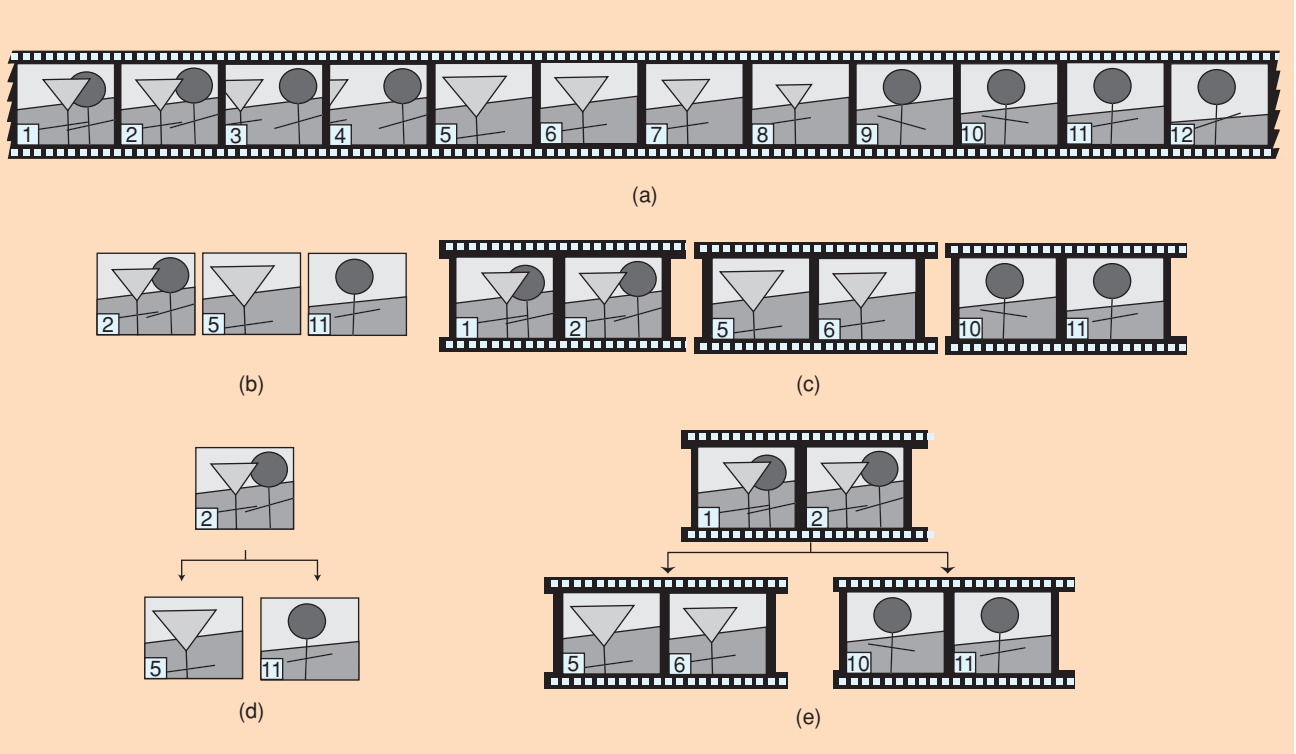
An important functionality when retrieving information in general is the availability of a condensed representation of a larger information unit. This can be the summary of a book, the theme or ground of a musical piece, the thumbnail of an image, or the abstract of a paper. Condensed representation allows us to assess the relevance or value of the information before committing time, effort, or computational and communication resources to process the entire information unit. It can also allow us to extract high-level information when we are not interested in the whole unit, especially during manual organization, classification, and annotation tasks. While for most types of information there is one or more standard forms of condensed representation, this is not the case for video.

In the literature, the term highlighting or storyboarding is used for video representation resulting in distinct images, while skimming is sometimes used for a representation resulting in a shorter video. These classes of methods are illustrated in Figure 4. In addition, we will use the term condensed representation to describe the processes of summarization, highlighting, and skimming in general.

HIGHLIGHTING

Highlighting is the field that has attracted by far the most attention, since it is the simplest and most intuitive. Its purpose is the selection of the video highlights, or those frames that contain the most, or most relevant, information for a given video. Highlighting is sometimes also referred to as key frame extraction.

An advanced, object-based method has been developed by Kim and Hwang [18]. They use the method described in [19]



[FIG4] Illustration of the different types of condensed representation: (a) original video, (b) highlighting, (c) skimming, (d) hierarchical highlighting, and (e) hierarchical skimming. The "film" segments denote continuous video, while images denote video frames.

to extract objects from each frame, and then use these objects to extract video key frames. Object detection and tracking are performed by conducting Canny edge detection on image differences and then finding areas enclosed by edges and tracking them. The objects are labeled either with the magnitude of the first eight Fourier descriptors (i.e., Fourier transform coefficients of the object contour) or with Hu's seven region moments. The distance between objects is defined as the L_1 distance between their descriptors, weighted by experimentally derived constants. Objects in the two frames are paired by the spatial proximity of their centers, and the distance between frames is the maximum of the distance between objects in such pairs. When the number of objects in a frame is different from that of the last frame, then the current frame is automatically labeled as a key frame. Otherwise, a frame is labeled as a key frame if its distance from the previous key frame is greater than a predefined threshold. This object-based method has the advantage of getting close to the physical semantics of the video. However, like other object-based approaches, it has the disadvantage that it is unable to perform well in complex scenes or in scenes with significant background motion. It is not accidental that all the experimental videos used in this method have a static background and a small number of well-defined objects.

Li et al. [20] approach the highlighting problem from the viewpoint of video compression and attempt to minimize the loss of visual information that occurs during highlighting. First they define the summarization rate as the ratio of the size of the condensed video to the size of the original one. They define distortion as the maximum distance between any frame in the highlighting and any frame in the original video. Then they formulate highlighting as one of two problems: either the minimization of distortion with the summarization rate as a constraint [minimum distortion optimal summarization (MDOS)] or the minimization of the summarization rate with distortion as a constraint [minimum rate optimal summarization (MROS)]. They then present a solution for the MROS problem by means of dynamic programming and a solution for the MDOS problem by efficiently searching through relevant solutions to the MROS problem. The only drawback of their approach is that it is too quantitative and does not address the difference between visual and semantic information.

HIERARCHICAL HIGHLIGHTING

A somewhat different task from highlighting is hierarchical highlighting, where the goal is to extract a hierarchy or a tree of highlight frames, so that the user can then interactively request more detail on the part of the video that interests him.

A semiautomatic method for the extraction of a hierarchy of highlights is proposed by Zhu et al. [27]. First, shots are found and a key frame is arbitrarily chosen for each of them. Similarity between shots is defined as the global color histogram and texture similarity between their key frames. Then shot groups are detected by heuristics based on similarity with the previous and following frames at their borders. Shot groups are described as either spatially coherent (i.e., composed of consecutive similar shots) or temporally coherent (e.g., shots forming a dialog), through clustering using a similarity threshold. Videos, groups, shots, and frames then have to be annotated manually with labels selected from a set of ontologies. The similarity between two groups is defined as the average similarity between shots in one group and the correspondingly most similar shots in the other group, using both the above visual features and the semantic labeling. Scenes are detected heuristically by merging groups according to their similarity. The summary is created hierarchically at four levels (video, scene, group, and shot) by semantically clustering the immediately lower level (e.g., clustering shots to create groups, etc). Finally, the highlighting is constructed by selecting the key frames that correspond to those shots and/or groups that contain the most annotation information. The system described is an advanced and complex one, but it suffers from the need for manual annotation and from the arbitrary selection of the key frame of each shot.

The extraction of the hierarchical highlights is performed in two stages by Ferman and Tekalp [21]. Their algorithm receives as input a video that has already been segmented into shots. Each shot is characterized by its α-trimmed average luminance histogram (although, theoretically, other features could be used as well), and each frame is characterized by the normalized distance between its histogram and that of the shot to which it belongs. Then fuzzy c-means clustering is performed on the frames in each shot, based on the above feature. The clustering is refined by two stages of cluster hardening and the pruning of uncertain data, with a recomputation of the fuzzy c-means performed between them. The clustering is repeated with increasing c, until decreasing clustering validity is detected. For each cluster, the most representative frame is selected as a key frame, based either on maximal proximity to the cluster's centroid or on maximal fuzzy cluster membership. In a second stage, temporally adjacent clusters with high similarity are merged to reduce the number of key frames generated according to user preference. The same merging approach can also be used to extract a hierarchy of highlights. Clustering is a typical method for highlight extraction, but the technical interest of the work lies in the refinement procedure of the clustering and the automatic determination of the optimal number of clusters.

SKIMMING

Video skimming is more difficult to perform effectively than video highlighting, because it is more difficult to extract appropriate features from its atomic elements (video segments instead of images), since they contain a much greater amount of information.

Gong and Liu [22] partly solve this problem by constraining the skimming to consist of whole shots of the video. First they down-sample the video temporally as well as spatially.

Each resulting frame is characterized by its color histogram, and SVD is performed on the resulting feature matrix **A** to reduce its dimensionality. For SVD-reduced frame features $\psi_i = [\upsilon_{i1}\ \upsilon_{i2}\ \ldots\ \upsilon_{in}]^T$, the authors define a new metric:

$$\|\psi_i\| = \sqrt{\sum_{j=1}^{\text{rank}(A)} \upsilon_{ij}^2},$$

which is inversely proportional to the cardinality of the feature within the matrix and, thus, to how typical the frame is within the video. Similarly, for a video segment S_i the metric $\text{CON}(S_i) = \sum_{\psi_i \in S_i} \|\psi_i\|$ is defined as a measure of visual content contained in this segment. The next step is clustering the reduced feature vectors. The most common frame is used to seed the first cluster, which is grown until a heuristic limit is reached. The other clusters are grown in such a way that they have $\text{CON}(S_i)$ similar to that of the first cluster. Then the longest shot is extracted from each cluster, and an appropriate number of these shots is selected to form the summary according to user preferences.

HIERARCHICAL SKIMMING

The creation of hierarchical video skimming is approached by Shipman et al. [23] as the creation of a specific type of video on demand, and as part of their Hyper-Hitchcock hypervideo authoring tool. Their work, which they call detail-on-demand video, is based on video segmentation into "takes," which are not real takes but scenes in the case of commercial videos and shots in the case of home videos, and "clips," which are shots in the case of commercial videos and cohesive subsegments of shots in the case of home videos. This way, the authors attempt to sidestep the semantic ambiguity of the scene concept. It should be noted that the segmentation is a prerequisite for their work. The hierarchical skimming method is constructed by selecting, for each level, a number of "clips" and connecting all or some of them to "clips" in the lower levels. The number of levels in the hierarchy is dependant solely on the duration of the video. The "clips" that make up each level are chosen by three methods: by temporal homogeneity, by a set of heuristics that ensures as much as possible that the selected clips belong to different takes, or by an importance metric that is either precomputed or manually assigned.

A different approach is presented by Ngo et al. [24], who create their hierarchical skimming based mostly on traditional concepts such as scenes and shots. Their first step is to segment the video into shots and subshots. Then the shots are organized as a graph with the vertex length equal to similarity of shot key frames, and grouped into clusters using the normalized cut algorithm. The temporal ordering of shots is used to make another directional graph with the clusters as nodes. This is then used to group clusters into continuous scenes. The attention value of a frame is computed from the motion of the MPEG macroblocks in it, with compensation for camera motion. Then the attention value for shots, clusters, and scenes is calculated by averaging the attention value in the immediately lower level of the hierarchy. The hierarchical skimming is produced by first discarding scenes and clusters that have a low number of shots and/or low attention value, and then for each cluster discarding subshots from those shots that have a low attention value. This procedure is continued until the required skimming length is reached.

PERFORMANCE EVALUATION

Because the objective of the above methods is to extract a semantically informative video representation, their objective performance evaluation is a very difficult affair. In practice, performance evaluation is only feasible by visual inspection of the results. In fact, when most authors need to perform experimental evaluation, they use panels of experts who quantify whether a specific video representation contains all salient information of the original video. Of course, there are other measures of the characteristics of each algorithm, such as the size of the resulting representation, but these provide no information on the quality and completeness of the results.

CONCLUSIONS

It is clear that the field of video shot detection has matured and is capable of detecting not only simple cuts but also gradual transitions with considerable recall and precision. This maturity is also evidenced by the establishment of a standard performance evaluation and comparison benchmark in the form of the TREC video shot detection task. The field has also advanced from simple and heuristic feature comparison methods to rigorous probabilistic methods, along with the use of complex models of the shot transition formation process. However, there is still room for algorithm improvement. It is clear that improved solutions for this problem will involve an increasingly rigorous inclusion of the a priori information about the actual physical process of shot formation as well as a thorough theoretical analysis of the editing effects, or a simulation of their generation as in [14].

On the other hand, condensed video representation remains a largely open issue. In part, this is due to its significantly higher complexity, the greater variety of tasks that fall under its heading, and because the evaluation of its results is almost entirely subjective. Furthermore, the desired condensed video representation may vary with the task at hand: a different representation might be needed for archival purposes than for promotional or educational purposes. However, the main reason for the relative failure of most automatic methods for the extraction of representative or significant frames or video segments is that the concept of being "representative" or "significant" is heavily dependent on the semantics of the video, and specifically on the semantic content of each video frame. So it is reasonable to state that the complete solution of the automatic highlighting and skimming tasks depends on the extraction of semantics at the basic (frame and object) level. Still, a significant amount of success has been shown in applications dealing with specific types of video content and when tasks of limited complexity are

attempted. It should also be noted that the lack of a commonly accepted experimental corpus hinders the evaluation of success in the general case.

ACKNOWLEDGMENTS

The C-SPAN video used in this work is provided for research purposes by C-SPAN through the TREC Information-Retrieval Research Collection. C-SPAN video is copyrighted. This work has been partly supported by the FP6 European Union Network of Excellence "Multimedia Understanding Through Semantic Computation and Learning" MUSCLE project (FP6-507752).

AUTHORS

Costas Cotsaces received the diploma of electrical and computer engineering from the Aristotle University of Thessaloniki. He is currently a researcher in the Artificial Intelligence and Information Analysis Laboratory in the Department of Informatics at the same university, where he is also working towards a Ph.D. degree. His research interests lie in the areas of computer vision, image and video processing, pattern recognition, and artificial intelligence. He is a Member of the IEEE and the Technical Chamber of Greece.

Nikos Nikolaidis received the diploma of electrical engineering in 1991 and the Ph.D. degree in electrical engineering in 1997, both from the Aristotle University of Thessaloniki, Greece. From 1992–1996 he served as teaching assistant in the Departments of Electrical Engineering and Informatics at the same university. From 1998–2002, he was postdoctoral researcher and teaching assistant at the Department of Informatics, Aristotle University of Thessaloniki. He is currently a lecturer in the same department. He coauthored the book *3-D Image Processing Algorithms* (Wiley, 2000). He has coauthored five book chapters, 19 journal papers, and 56 conference papers. His research interests include computer graphics, image and video processing and analysis, copyright protection of multimedia, and 3-D image processing. He has been a scholar of the State Scholarship Foundation of Greece.

Ioannis Pitas received the diploma of electrical engineering in 1980 and the Ph.D. degree in electrical engineering in 1985, both from the Aristotle University of Thessaloniki, Greece. Since 1994, he has been a professor at the Department of Informatics, Aristotle University of Thessaloniki. From 1980–1993, he served as scientific assistant, lecturer, assistant professor, and associate professor in the Department of Electrical and Computer Engineering at the same University. He served as a visiting research associate or visiting assistant professor at several universities. He has published 140 journal papers and 350 conference papers and contributed to 18 books in his areas of interest. He has edited or coauthored another five books. He has also been an invited speaker and/or member of the program committee of several scientific conferences and workshops. In the past, he served as associate editor or coeditor of four international journals and as general or technical chair of three international conferences. His current interests are in the areas of digital image and video processing and analysis, multidimensional signal processing, watermarking, and computer vision.

REFERENCES

[1] N. Dimitrova, H.-J. Zhang, B. Shahraray, I. Sezan, T. Huang, and A. Zakhor, "Applications of video-content analysis and retrieval," *IEEE Multimedia*, vol. 9, no. 3, pp. 42–55, July 2002.

[2] P. Salembier and J. Smith, "MPEG-7 multimedia description schemes," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 6, pp. 748–759, June 2001.

[3] R. Lienhart, "Reliable transition detection in videos: A survey and practitioner's guide," *Int. J. Image Graphics*, vol. 1, no. 3, pp. 469–486, Sept. 2001.

[4] I. Koprinska and S. Carrato, "Temporal video segmentation: A survey," *Signal Processing: Image Comm.*, vol. 16, pp. 477–500, Jan. 2001.

[5] R.M. Ford, C. Robson, D. Temple, and M. Gerlach, "Metrics for shot boundary detection in digital video sequences," *ACM Multimedia Syst.*, vol. 8, no. 1, pp. 37–46, Jan. 2000.

[6] U. Gargi, R. Kasturi, and S.H. Strayer, "Performance characterization of video-shot-change detection methods," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 1, pp. 1–13, Feb. 2000.

[7] D. Lelescu and D. Schonfeld, "Statistical sequential analysis for real-time video scene change detection on compressed multimedia bitstream," *IEEE Trans. Multimedia*, vol. 5, no. 1, pp. 106–117, Mar. 2003.

[8] A. Hanjalic, "Shot-boundary detection: Unraveled and resolved?" *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 2, pp. 90–105, Feb. 2002.

[9] J. Nam and A. Tewfik, "Detection of gradual transitions in video sequences using B-spline interpolation," *IEEE Trans. Multimedia*, vol. 7, no. 4, pp. 667–679, Aug. 2005.

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

[11] Z. Cernekova, C. Kotropoulos, and I. Pitas, "Video shot segmentation using singular value decomposition," in *Proc. 2003 IEEE Int. Conf. Multimedia and Expo*, Baltimore, Maryland, July 2003, vol. 2, pp. 301–302.

[12] R. Zabih, J. Miller, and K. Mai, "A feature-based algorithm for detecting and classification production effects," *ACM Multimedia Syst.*, vol. 7, no. 1, pp. 119–128, Jan. 1999.

[13] J. Yu and M.D. Srinath, "An efficient method for scene cut detection," *Pattern Recognit. Lett.*, vol. 22, no. 13, pp. 1379–1391, Nov. 2001.

[14] R. Lienhart, "Reliable dissolve detection," in *Proc. SPIE*, vol. 4315, pp. 219–230, Jan. 2001.

[15] Z.-N. Li, X. Zhong, and M.S. Drew, "Spatialtemporal joint probability images for video segmentation," *Pattern Recognit.*, vol. 35, no. 9, pp. 1847–1867, Sep. 2002.

[16] G. Boccignone, A. Chianese, V. Moscato, and A. Picariello, "Foveated shot detection for video segmentation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 3, pp. 365–377, Mar. 2005.

[17] "TREC video retrieval evaluation" [Online]. Available: http://www-nlpir.nist.gov/projects/trecvid/

[18] C. Kim and J.-N. Hwang, "Object-based video abstraction for video surveillance systems," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp. 1128–1138, Dec. 2002.

[19] C. Kim and J.-N. Hwang, "Fast and automatic video object segmentation and tracking for content-based applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 2, pp. 122–129, Feb. 2002.

[20] Z. Li, G.M. Schuster, and A.K. Katsaggelos, "Minmax optimal video summarization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 10, pp. 1245–1256, Oct. 2005.

[21] A. Ferman and A. Tekalp, "Two-stage hierarchical video summary extraction to match low-level user browsing preferences," *IEEE Trans. Multimedia*, vol. 5, no. 3, pp. 244–256, June 2003.

[22] Y. Gong, "Summarizing audio-visual contents of a video program," *EURASIP J. Appl. Signal Processing*, vol. 2003, pp. 160–169, Feb. 2003.

[23] F. Shipman, A. Girgensohn, and L. Wilcox, "Creating navigable multi-level video summaries," in *Proc. 2003 IEEE Int. Conf. Multimedia and Expo*, Baltimore, Maryland, July 2003, vol. 2, pp. 753–756.

[24] C.-W. Ngo, Y.-F. Ma, and H.-J. Zhang, "Video summarization and scene detection by graph modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 2, pp. 296–305, Feb. 2005.

[25] Z. Xiong, X. Zhou, Q.Tian, Y. Rui, and T.S. Huang, "Semantic Retrieval in Video," *IEEE Signal Processing Mag.*, vol. 23, no. 2, pp. 18–27, 2006.

[26] W.K. Li and S.H. Lai, "Storage and retrieval for media databases," *Proc. SPIE*, vol. 5021, pp. 264-271, Jan. 2003.

[27] X. Zhu, J. Fan, A.K. Elmagarmid, and X. Wu, "Hierarchical video summarization and content dexcription joint semantic and visual similarity," *ACM Multimedia Syst.*, vol. 9, no. 1, 2003.

SP